
RELAZIONE TECNICA: GESTIONE DEI PERMESSI IN AMBIENTE LINUX

Studente: Rocco Paolo Caccamo

Corso: Cybersecurity & Ethical Hacking

Modulo: S10L2 - System Hardening & Access Control

Data: 10/02/2026

1. Obiettivo dell'Esercitazione

L'esercitazione ha come oggetto la configurazione e la gestione dei permessi di lettura, scrittura ed esecuzione su sistemi Linux. Lo scopo è applicare il **Principio del Privilegio Minimo** per mettere in sicurezza una risorsa critica, impedendo accessi non autorizzati da parte di utenti o gruppi non privilegiati.

2. Creazione della Risorsa (Setup)

Come prima fase, è stato generato un file denominato `file.txt` contenente dati sensibili simulati. La creazione è avvenuta tramite redirezione dell'output standard all'interno di una nuova directory di lavoro.

Comando eseguito: `echo "Dato_Sensibile_Top_Secret" > file.txt`

```
$ echo "Dato_Sensibile_Top_Secret" > file.txt
```

Fig. 1 - Creazione del file target e popolamento con dati di prova.

3. Analisi dei Permessi Preliminari (Reconnaissance)

Prima di procedere con l'hardening, è stata effettuata una verifica dei permessi assegnati automaticamente dal sistema operativo (umask di default).

L'analisi dell'output di `ls -l` ha evidenziato la seguente configurazione: `-rw-rw-r--`.

- **User:** Lettura e Scrittura (`rw-`).

- **Group:** Lettura e Scrittura (`rw-`).
- **Others:** Lettura (`r--`).

Valutazione del Rischio: La configurazione di default è stata valutata **non sicura**. Il permesso di lettura assegnato a *Others* espone il contenuto del file a qualsiasi utente autenticato nel sistema, violando la confidenzialità. Inoltre, il permesso di scrittura per il *Group* espone il file a rischi di integrità.

```
-rw-rw-r-- 1 kali kali 26 Feb 10 08:29 file.txt
```

Fig. 2 - Verifica dei permessi attuali: evidenziata l'esposizione in lettura pubblica.

4. Hardening e Modifica dei Permessi

Per mitigare i rischi identificati, si è proceduto alla modifica dei permessi utilizzando il comando `chmod`. La configurazione scelta è stata **600** (`rw-----`).

Motivazione delle Scelte:

- **User (6 - Read/Write):** Mantenuti i diritti esclusivi al proprietario per garantire l'operatività e la gestione del file.
- **Group (0 - Nessun accesso):** Rimossi i permessi per prevenire movimenti laterali o modifiche accidentali da parte di utenti appartenenti allo stesso gruppo funzionale.
- **Others (0 - Nessun accesso):** Rimossi totalmente i permessi per garantire che nessun utente esterno possa leggere (`cat`) o enumerare il contenuto del file.

```
(kali㉿kali)-[~]
$ chmod 600 file.txt

-rw-rw-r-- 1 kali kali 59 Jan 29 05:16 custom.txt
-rw----- 1 kali kali 26 Feb 10 08:29 file.txt
```

Fig. 3 - Applicazione della policy di sicurezza e verifica dell'output `rw-----`.

5. Test di Intrusione e Verifica Finale

A conclusione dell'attività, è stato eseguito un test di verifica (Proof of Concept) per confermare l'efficacia delle restrizioni applicate.

Metodologia di Test: Utilizzando il comando `sudo -u nobody bash -c 'echo "Hacked_Real" >> file.txt'`, è stato simulato il comportamento di un attaccante o di un utente non autorizzato che tenta di accedere alla risorsa protetta.

Analisi dei Risultati: Il sistema operativo ha bloccato il tentativo di lettura e scrittura restituendo l'errore "**Permission denied**". Questo conferma che le Access Control Lists (ACL) standard di Linux stanno operando correttamente, isolando la risorsa dall'ambiente circostante.

```
(kali㉿kali)-[~]
└─$ sudo -u nobody bash -c 'echo "Hacked_Real" >> file.txt'
bash: line 1: file.txt: Permission denied
```

Fig. 4 - Tentativo di accesso non autorizzato bloccato dal sistema.

Esito: L'esercizio dimostra come la restrizione dei permessi (da 664 a 600) sia una misura fondamentale di *System Hardening* per garantire la Triade CIA (Confidenzialità, Integrità, Disponibilità) dei dati locali.