**Report su Buffer Overflow e Misure di Mitigazione**

**Introduzione al Buffer Overflow**

Nel corso dell'esperimento condotto su un sistema operativo Kali Linux, ho avuto l'opportunità di analizzare una vulnerabilità comune nei programmi scritti in linguaggio C, nota come "Buffer Overflow" (BOF). Il Buffer Overflow si verifica quando i dati vengono scritti in un buffer e superano la capacità di memoria che il buffer può contenere. Questo eccesso di dati può sovrascrivere valori importanti nella memoria e condurre a comportamenti imprevisti del programma, inclusi crash e potenziali vulnerabilità di sicurezza.

**Procedura**

Per illustrare un Buffer Overflow, ho creato un semplice programma in C che chiedeva all'utente di inserire un nome. Il programma era progettato con un buffer di soli 10 caratteri. Quando inserivo un nome che eccedeva questa lunghezza, specificamente 30 caratteri, il programma si interrompeva e restituiva un "segmentation fault", indicando che l'accesso alla memoria oltre i limiti consentiti aveva causato un errore critico.

Immagine che contiene testo, schermata, Carattere, Software multimediale

Descrizione generata automaticamente

Durante l'esperimento, ho seguito questi passaggi per creare e testare il programma:

1. Ho aperto il terminale su Kali Linux.

2. Mi sono spostato nella directory del desktop con il comando `cd /home/Kali/Desktop`.

3. Ho creato il file sorgente usando l'editor nano con il comando `nano BOF.c`.

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

4. Ho inserito il codice sopra nel file nano.

5. Ho salvato e chiuso il file in nano con i comandi:

- `Ctrl` e `x` per uscire.

- `y` e `invio` per salvare il file.

6. Ho compilato il programma con il comando `gcc -g BOF.c -o BOF`.

7. Ho eseguito il programma con `./BOF`.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

**Risultati**

L'esecuzione del programma con un input di 5 caratteri funzionava senza problemi. Tuttavia, inserendo 30 caratteri, il programma crashava, dimostrando così la vulnerabilità al Buffer Overflow. Questo errore non solo interrompe il programma ma potrebbe anche essere sfruttato per eseguire codice arbitrario, una grave minaccia alla sicurezza.

**Modifica e Mitigazione**

Per mitigare questo rischio, ho aumentato la dimensione del buffer da 10 a 30 caratteri e ho ricompilato il programma. Questa modifica ha impedito il crash del programma quando testato con lo stesso input di 30 caratteri.

Immagine che contiene testo, schermata, Software multimediale, Carattere

Descrizione generata automaticamente

**Discussione e Conclusione**

Il Buffer Overflow è un problema comune e pericoloso in programmazione, specialmente in linguaggi come C che non gestiscono automaticamente la memoria. Le conseguenze di tali vulnerabilità possono essere gravi, da errori di sistema a problemi di sicurezza. Per prevenirle, è fondamentale adottare pratiche di programmazione sicure, come:

- Utilizzare funzioni sicure per la gestione delle stringhe che controllano la dimensione dei buffer.

- Evitare l'uso di funzioni insicure come “gets()” o “scanf()” senza specificare un limite alla lunghezza dell'input.

- Adottare linguaggi di programmazione di alto livello che gestiscono la memoria automaticamente dove possibile.