COFELET

- **Scenario**: Un utente malintenzionato vuole ottenere l'accesso come amministratore su un'applicazione che gestisce registrazioni e login, e che possiede una vulnerabilità di heap overflow nella funzione `registrazione_utente`.
- **Task**: Utilizzare l'overflow sull'heap per modificare un valore booleano `is_admin` nella struttura `utente` di un account registrato per ottenere privilegi da amministratore.
- **Goal**: Accedere come amministratore e visualizzare la flag nascosta `HHC{4dm1n_f14g_1534}`, che è accessibile solo dopo aver impostato `is_admin = true`.
- **Condition**: Il giocatore deve sfruttare la vulnerabilità nella funzione `gets()` che consente di superare la dimensione prevista per i campi `nome_utente`, `password` e `email`, permettendo la modifica dei dati sulla struttura `utente`.
- **Scenario Execution Flow (SEF)**:
    4. **Analizzare** il binario per identificare le funzioni, individuando `registrazione_utente` e `login`.
    5. **Controllare** la struttura dei dati allocata da `malloc` in `registrazione_utente`.
    6. **Inserire** input sovradimensionato in `email` per sovrascrivere il campo booleano `is_admin`.
    7. **Effettuare il login** con le credenziali modificate e verificare l'accesso come amministratore.
- **Education Context**: Comprendere e sfruttare un heap overflow per manipolare strutture dati complesse.
- **Gaming Context**: Nessun requisito specifico per l'ambiente di gioco.
- **Knowledge Skills Ability (KSA)**: Capacità di riconoscere vulnerabilità di heap overflow, utilizzo del debugging con `pwndbg` o `GDB`, e capacità di manipolare dati di una struttura.
- **Learning Objects**: Alla fine dell'esperienza, l'utente avrà familiarità con l'heap overflow in strutture dati e l'importanza del controllo sui campi di input.
- **Teaching Content**: Documentazione sulla gestione della memoria in C e basi sulle strutture dati.