Teoria e concetti rilevanti

Heap Overflow

Un heap overflow si verifica quando viene scritto più del dovuto nella memoria allocata dinamicamente (heap), andando a sovrascrivere dati adiacenti o strutture di gestione della memoria. In questo caso, la vulnerabilità deriva dall'uso della funzione `gets()`, che non limita la lunghezza dell'input e consente un overflow.

Struttura dei Dati su Heap e Booleani

In questa applicazione, la struttura `utente` è allocata dinamicamente su heap. La funzione `gets()` scrive i dati degli utenti `nome_utente`, `password` ed `email` senza alcun controllo, consentendo la scrittura continua fino alla sovrascrittura di `is_admin`. Essendo un `bool`, `is_admin` assume valori `true` o `false` e quindi, con un overflow appropriato, si può impostare `is_admin` a un valore `true` (qualsiasi valore non nullo).

CWE Coinvolti

- **CWE-122** - Heap-based Buffer Overflow: la vulnerabilità principale dovuta alla possibilità di eccedere i limiti di un buffer allocato su heap.
- **CWE-242** - Use of Inherently Dangerous Function (`gets`): `gets()` è usata senza controllare la lunghezza dell'input, permettendo l'overflow.
- **CWE-787** - Out-of-bounds Write: si verifica quando il buffer viene riempito oltre i limiti previsti, scrivendo in aree di memoria non destinate.

Soluzione dettagliata e passaggi

1. **Analisi del binario**: Dopo aver lanciato il binario, osserviamo il menù principale che consente di registrare utenti e di effettuare il login. Analizzando i permessi, notiamo che `is_admin` è settato solo per l'utente `admin`, mentre ogni altro utente viene registrato come non amministratore.
2. **Esplorazione del codice con `pwndbg`**:
    - Esaminando le funzioni con `info functions`, troviamo `registrazione_utente`.
    - Settando breakpoint su `registrazione_utente` e osservando `malloc`, si vede che alloca 71 byte, sufficienti per memorizzare `nome_utente`, `password`, `email` e `is_admin`.

3. **Comprendere la disposizione della memoria**:
    - La struttura dell'utente è composta da:
        - `nome_utente` di 20 byte.
        - `password` di 20 byte, che inizia a 20 byte dall'inizio.
        - `email` di 30 byte, iniziando a 40 byte dall'inizio.
        - `is_admin` è posizionato dopo `email`, quindi al byte 70.
    - **Obiettivo**: sovrascrivere `is_admin` utilizzando un overflow di `email`, impostandolo a un valore diverso da zero (`true`).
4. **Sovrascrivere `is_admin` con overflow**:
    - Inserendo un `nome_utente` e una `password` di 20 byte ciascuno, `email` inizierà a byte 40 e, utilizzando 31 byte, potrà sovrascrivere `is_admin`.
    - Questo significa che un input di `31 byte` sovrascriverà `is_admin` con un valore non nullo, abilitando l'accesso da amministratore.
5. **Effettuare il login come amministratore**:
    - Dopo aver registrato l'utente con un overflow nell'email, inseriamo il `nome_utente` e `password` usati in fase di registrazione.
    - Verifichiamo che `is_admin` è `true` e riceviamo la flag `HHC{4dm1n_f14g_1534}`.

Codice d'esempio per overflow di `email`

Di seguito, un esempio di input:

yaml
Copy code

```
Nome utente: Utente20Caratteri
Password: Password20Caratteri
Email: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa  # (31 'a' per sovrascrivere `is_admin`)
```

Con questi input, `is_admin` viene impostato a un valore diverso da zero e consente l'accesso amministrativo.