Mi a Buffer túlcsordulás?

A buffer túlcsordulás akkor történik, amikor adatokat írnak egy bufferbe (memóriaterületre) úgy, hogy az adatok meghaladják a rendelkezésre álló buffer méretet. Ez a “túlcsordulás” felülírhatja a memória más részeit, ami biztonsági rést jelent, mert a támadók rosszindulatú kódot helyezhetnek el a programban.

Egyszerű C Példa Tekintsünk egy egyszerű C programot, amely tartalmaz egy buffer túlcsordulásos sebezhetőséget:

```
#include <stdio.h>
#include <string.h>

void veszelyesFuggveny(char *bejovoAdat) {
    char buffer[10];
    strcpy(buffer, bejovoAdat); // veszélyes másolás
}

int main() {
    char nagyAdat[100] = "Túl hosszú szöveg, ami túlcsordulást okoz...";
    veszelyesFuggveny(nagyAdat);
    return 0;
}
```

Ebben a példában a *veszelyesFuggveny()* egy 10 karakter hosszúságú buffer-t használ. Az *strcpy()* függvényt használjuk az adatok másolására a bufferbe, anélkül, hogy ellenőriznénk az adatok hosszát. Ha a *bejovoAdat* több mint 10 karaktert tartalmaz, akkor a buffer túlcsordul, és potenciálisan felülírja a memória más részeit.

Veszélyek

A buffer túlcsordulás veszélyes, mert a támadók kihasználhatják ezt a sebezhetőséget rosszindulatú kód futtatására. Például, ha a túlcsordulás felülírja a visszatérési címet a memóriában, a támadó irányíthatja a programot, hogy saját kódját hajtsa végre.

Védekezési lehetőségek

A buffer túlcsordulás elleni védekezés érdekében fontos, hogy mindig ellenőrizzük az adatok hosszát, mielőtt azokat egy bufferbe másoljuk. A C nyelvben használhatjuk a *strncpy* függvényt a strcpy helyett, amely lehetővé teszi a másolandó karakterek számának korlátozását, vagy használhatunk magasabb szintű nyelveket, amelyek automatikusan kezelik ezt a típusú memória kezelést.

A korábbi egy karakteres jelszókezelő kibővítése

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int checkPassword(const char *jelszo) {
    char buffer[12];
    int jogosultsag = 0; // Alapértelmezett: nincs jogosultság

    strcpy(buffer, jelszo);

    if (strcmp(buffer, "titkos") == 0) {
        jogosultsag = 1; // Jelszó helyes: jogosultság engedélyezve
    }

    return jogosultsag;
}

int main() {
    char jelszo[100];
    printf("Kérem a jelszót: ");
    gets(jelszo); // Veszedelmes függvény, nem ellenőrzi a buffer méretét

    if (checkPassword(jelszo)) {
        printf("Hozzáférés engedélyezve.\n");
    } else {
        printf("Hozzáférés megtagadva.\n");
    }

    return 0;
}
```

A *gets()* függvényt használjuk a jelszó bekérésére, ami veszélyes, mert nem ellenőrzi a rendelkezésre álló buffer méretét. Ha a felhasználó több mint 12 karaktert ad meg, az túlcsordulást okoz a buffer-ben a *checkPassword()* függvényben.

Ez a példa bemutatja, hogyan lehet kihasználni a buffer túlcsordulását, de fontos hangsúlyozni, hogy a valós alkalmazásokban mindig biztonságos programozási technikákat kell alkalmazni. A *gets()* függvényt soha nem szabad már éles környezetben használni, helyette biztonságos alternatívákat, mint a *fgets()*, kell alkalmazni. Ezenkívül a modern fordítók és operációs rendszerek számos védelmi mechanizmust tartalmaznak (pl. stack canaries, ASLR) a buffer túlcsordulás ellen.

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:szamitastechnika:buffer_tulcsordulas

Last update: **2023/11/14 11:22**