Fase 1:

Compara el input con un string (vamos a llamarla contraseña). Si no son iguales. Explota la bomba. Para descubrir la contraseña, se usó gdb. Se colocó un breakpoint en la función **strings\_not\_equal**. Se pidió el estado de los registros (info register) para ver la dirección de memoria guardada en RSI, que es donde está guardada la contraseña. Para ver lo que contenía esa memoria, se usó el comando x/s 0x4c9a58.

(gdb) x/s 0x4c9a58

0x4c9a58: "Los hermanos sean unidos porque esa es la ley primera porque si entre ellos se pelean los devoran los de ajuera"

Fase 2

Al principio de la fase 2, vemos que hay una función **strchr** que busca un espacio " " en el input. Esto nos dice que se esperan dos palabras. Después, encontramos dos call a **strtol**, que convierten el string a un número entero. Entonces, no espera dos palabras, sino dos números.

Pasamos a la función **misterio**, que toma esos dos números y verifica dos condiciones:

1. El resultado de la operación n1 + n2 + 32 debe tener once bits en 1 en binario (11 bits con valor 1).
2. El resultado de n1 XOR n2 debe ser negativo.

De (2), podemos deducir que el bit más significativo de n2 debe ser 1 (para que el número sea negativo). Se propone -1.

Ahora necesitamos encontrar n1 tal que la operación n1 + (-1) + 32 de como resultado un número con 11 bits en 1. Se proponen 4127 y 2080.