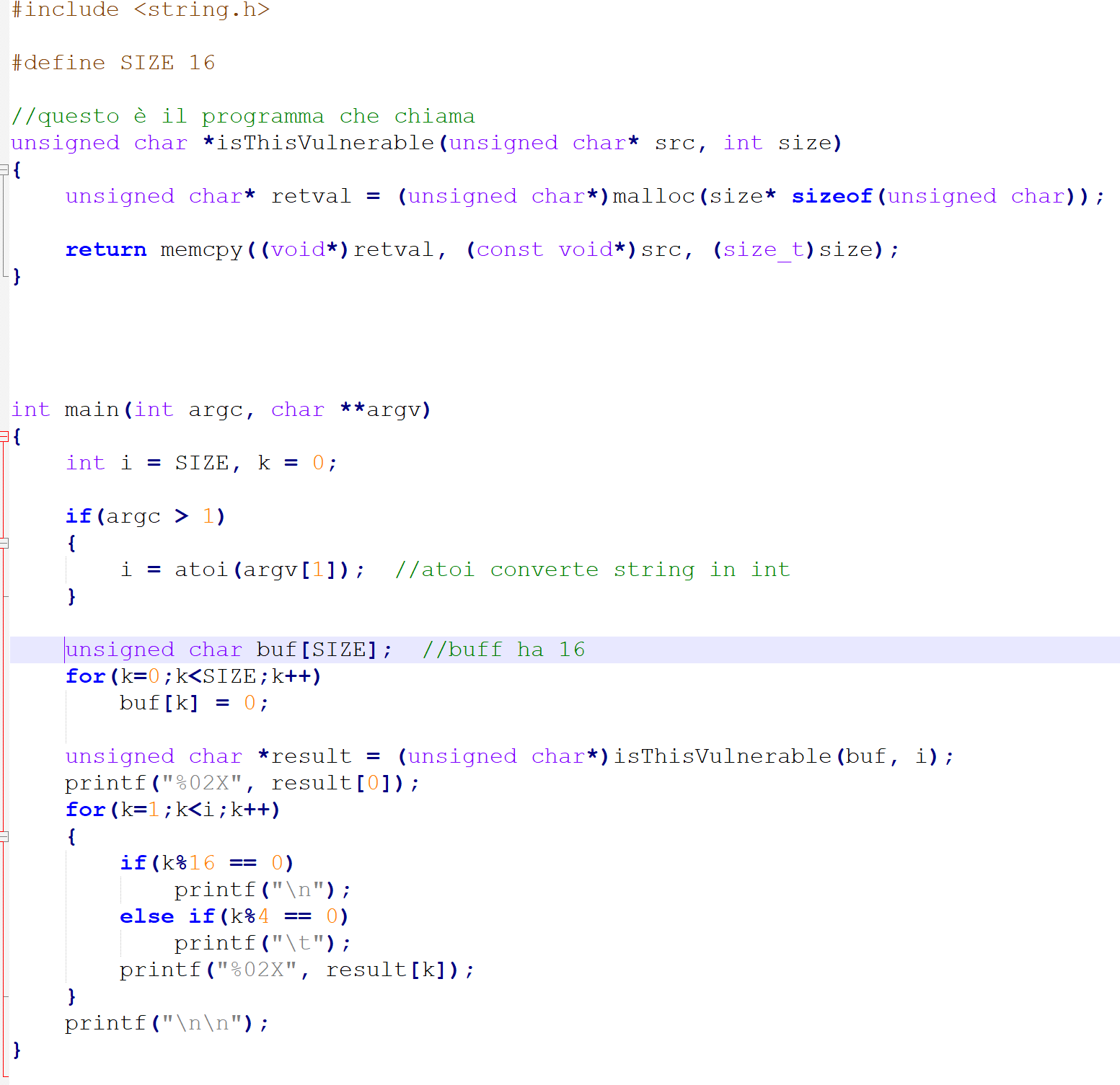
**Soluzione parte 2 esame di figlio di puttana di migliardi**

****

**commenti :**

**memcopy (puntatore della “posizione” del dest, puntatore della posizione del sorgente, numeri di byte da copiare a partire dal posizione del sorgente ). → buffer[20] = sorgente , byte = 50, copia altre 30 cose in piu dopo buffer**

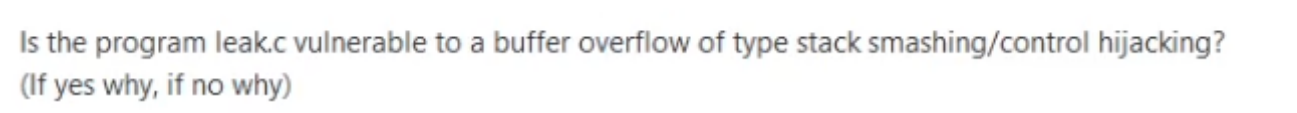
**atoi = converte string in int**

**malloc= prepara un'area di memoria grande quanto size(arbitrario)**

il programma prende argv (la lista dei parametri passati da input) e usa atoi per rendere in formato int . (<--Questo è i )

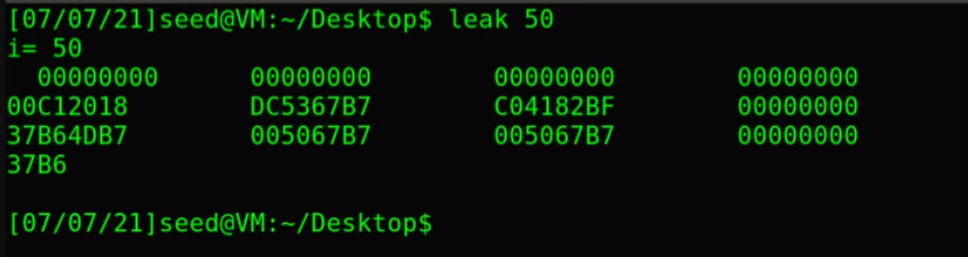
Prepara un buffer da 16 celle [sempre ] e poi chiama la funzione is isThisVulnerable passando buff e i

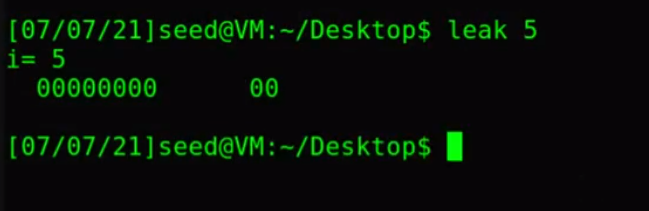
Nel sotto-programma , buff è sorgente , i è size , retval è l’area di memoria destinazione preparata con la funzione malloc grande di dimensione i .



Q1:

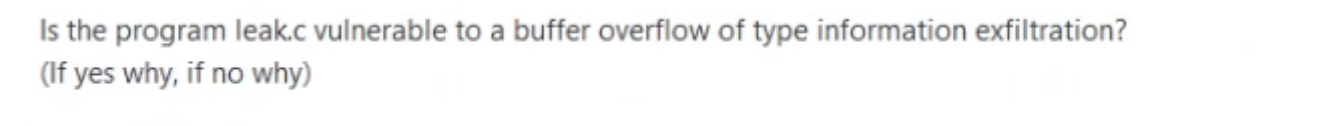
NO, perché nel caso in cui retval è piu grande del size di buff,

eg: retval con dimensione i = 50 , lui copia gli elementi del buffer più altre che stanno nello DOPO lo stack 

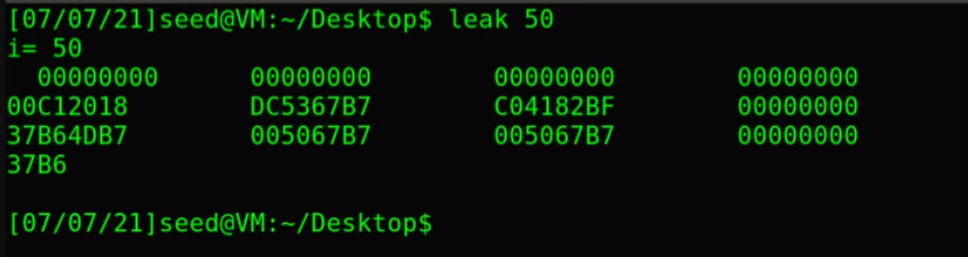


ps : ogni elemento è scritto con 2 byte : 0= 00

Q2

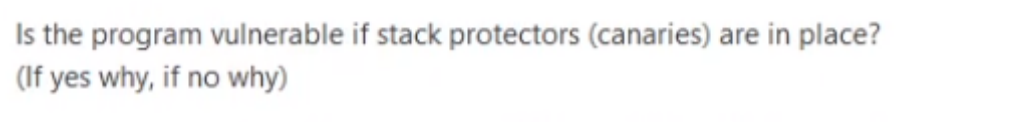


SI, perchè nel caso in cui retval sia piu grande di buff , eg: quando prepara un area di memoria con 50 byte , usa i primi 16 per copiare il buff , **MA I RIMANENTI SONO VISIBILE QUANDO FACCIO PRINT**



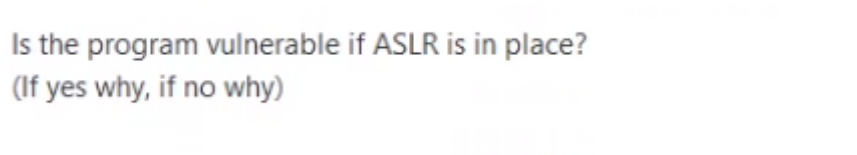
questo è può essere information exfiltration e utilizzabile per attacco.

Q3:

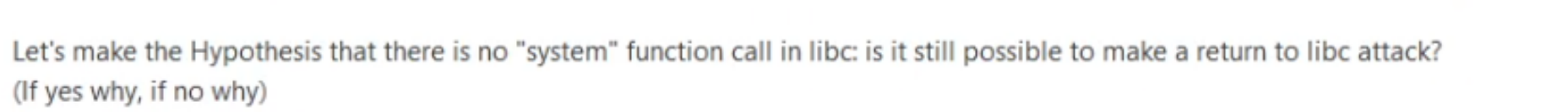


Yes, the program is still vulnerable because ,we don't overwrite the stack

Q4:



si, perchè con ASLR(Address Space Layout Randomization,), quando si va creare Retval, restituisce un puntatore con area di memoria **DIversa a**d ogni esecuzione, quindi non soffre più di il problema di q2.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Q5:

yes, possiamo comunque usare il commando execve per farlo

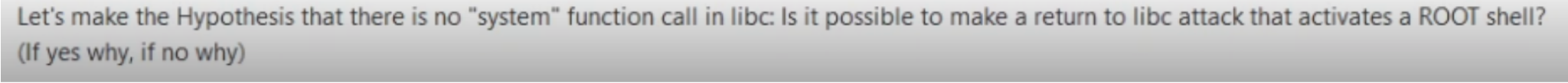
Q6:



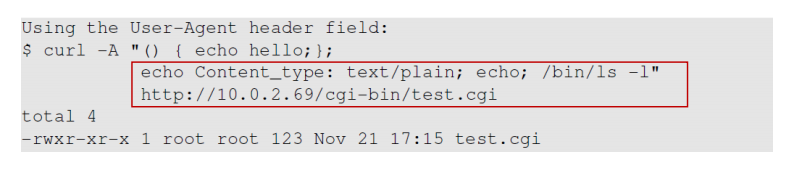


yes, come prima è possibile usare execve per farlo

Q7:

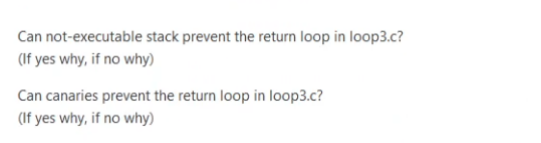
yes, usare ancora una volta il fottuto EXECVE





curl -A “() { echo hello ; };

echo Content\_type : text/plain ; echo; cat/etc/password>/temp/log.txt “



il canaries non previene loop , perchè è un contro misura che va messa prima del return address. se è modificata vuol dire che c’ è stato un buffer over flow che ha scritto “ in sequenza “ fino al return address, e che ha sovrascritto anche canaries . Ma qui laddress di ritorno è modificato di “Preciso” e mirata ! quindi canaries non previene