[Lab 0] Binary exploitation

Scaricare questo tool:

Analisi statica

```
ghidra
```

```
IDA free
```

Esempi di compilazione di file:

```
gcc -m64 -fno-stack-protector -z execstack -o find_record find_record.c
```

- m64 per farlo in x64
- -fno-stack-protector rimuove la protezione dello stack (disabilito le canaries)

così posso vedere info sul file

```
file nome_file
```

IDA FREE

in IDA per migliorare la visibilità posso fare

```
view -> open subwievs -> generate pseudocode
```

per vedere in che posizione si trova una funzione:

```
seleziona la funzione -> guarda hex -> vedi tipo 0011A0 ecc
IDA FREE
```

Cosa fare

```
seguire dove va a finire l'input dell'utente
```

Analisi dinamica

tools:

```
gdb -> sudo apt install gdb
```

Servono dei plugin per migliorare l'uso di gdb:

```
gdb peda
```

```
gef
```

- più aggiornato, lo trovo su github

Per istallare GEF:

```
 bash -c "$(curl -fsSL https://gef.blah.cat/sh)"
```

Un tool utile per trovare offset ecc è:

```
objdump -D nome_eseguibile
```

- -D per decompilare il binario

Un altro tool è

```
checksec -> sudo apt install checksec
```

Come lo uso

```
checksec --file= nome_binario
```

- mi dice quali opzioni di sicurezza ci sono sul file binario

COME USARE GDB

```
gdb ./nome_binario
```

per settare un break point

```
b main
```

- mette un breakpoint sulla funzione main

per eseguire il programma

```
run parametri_input_del_programma
```

se voglio avanzare di una singola istruzione alla volta faccio:

```
si
```

se voglio eseguire tutto il programma fino al prossimo breakpoint

```
continue

#oppure c
```

Se voglio spostarmi direttamente su una funzione che non è mai chiamata posso spostarmi direttamente così:

```
#sempre dentro gdb

disass secret_function
```

echo 0 | sudo tee /proc/sys/kernel/randomize_va_space

PER SEMPLICITA DISATTIVO LA ASRL

```
echo 0 | sudo tee /proc/sys/kernel/randomize_va_space
```

per riattivarla sul sistema

```
echo 2 | sudo tee /proc/sys/kernel/randomize_va_space
```

per vedere gli indirizzi di un file

```
ldd file_binario
```

Trovare le funzioni di un programma senza gidra ecc:

```
readelf -s nome_file  | grep i func
```

Cosa posso fare per trovare gli indirizzi di due funzioni e la distanza

```
objdump -D find_record | grep "secret_function"
```

prendo l'indirizzo in cui inizia l'esecuzione del file e faccio la differenza hex con quello di secret function

- secret function indirizzo - inidirizzo inizio dell'esecuzione

su gdb faccio

```
set *(void **)($rbp +8) = indirizzo a cui saltare (ovvero quello corrente + il valore della differenza calcolata prima)
```

Vedere address di inizio del file

```
#in gdb

disass file_binario
```