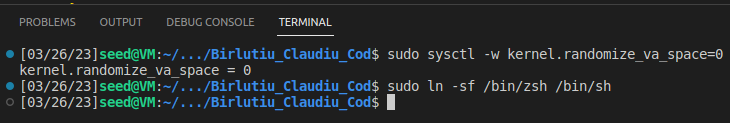
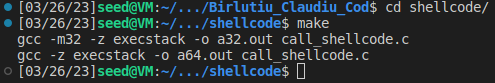
# Raport pentru lucrarea 3: Atacul cu depășire de zonă de memorie (versiunea Set-UID)

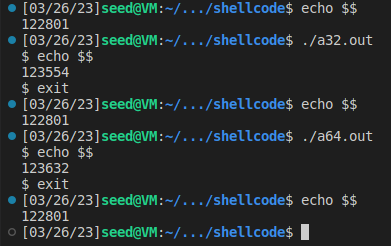
Autor: Birlutiu Claudiu-Andrei

## Sarcina 1: Familiarizarea cu ShellCode

* Pregatirea mediului prin executarea comenzilor din laborator
* am rulat comanda **make** în directorul shellcode și aceasta a creat cele 2 fisiere executabilele ale shellcode-ului (pe 32 de biti și 64 de biti) și de asemenea la crearea lor s-a folosit optiunea de execuție de pe stiva

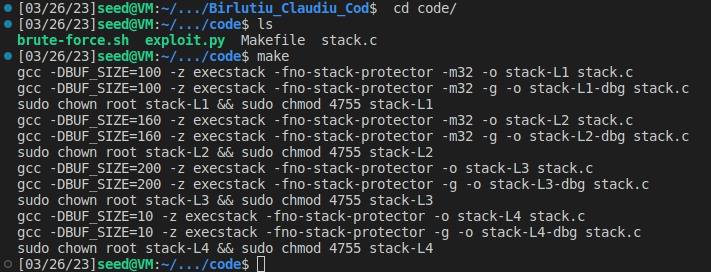


* am lansat în execuție cele 2 executabile și am observat ca în ambele cazuri se deschide un shell nou pe un proces nou (id-uri diferite). Se poate observa în ss-ul de mai jos cum sa deschid cele 2 shell-uri cu id diferit

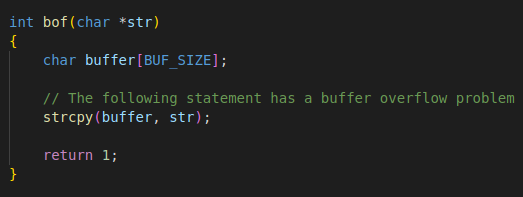


## Sarcina 2: Înțelegerea programului vulnerabil

* Am rulat comanda **make** astfel ca s-au construit mai multe fisiere executabile ale preogramului stack (pe 32 și 64 de biti) cu valori ale dimenisunii buffer-ului diferite și de asemenea crearea programelor pe care se poate face debugging



* codul stack.c manifesta o vulnerailitate în cadrul functiei bof din cauza faptului ca dimensiunea string-ului primit ca parametru și care e copiat cu ajutorul functiei strcpy (care nu verifica dimensiunile maxime destinație – sursa) => lungimea sirului str poate fi mai mare decât BUF\_SIZE = > se vor copia bytes într-o zona nedorita



## Sarcina 3: Lansarea atacului asupra programului pe 32 biți

* În prima faza se va afla distanta dintre pozittia de inceput a tamponului si locul unde este stocata adresa de retur.

## Sarcina 4: Lansarea atacului asupra programului pe 64 biți

## Sarcina 5: Înfrângerea contra-măsurii din dash

## Sarcina 6: Înfrângerea randomizării adreselor

## Sarcina 7a: Protecția StackGuard activată

## Sarcina 7b: Protecția prin stivă ne-executabilă activată