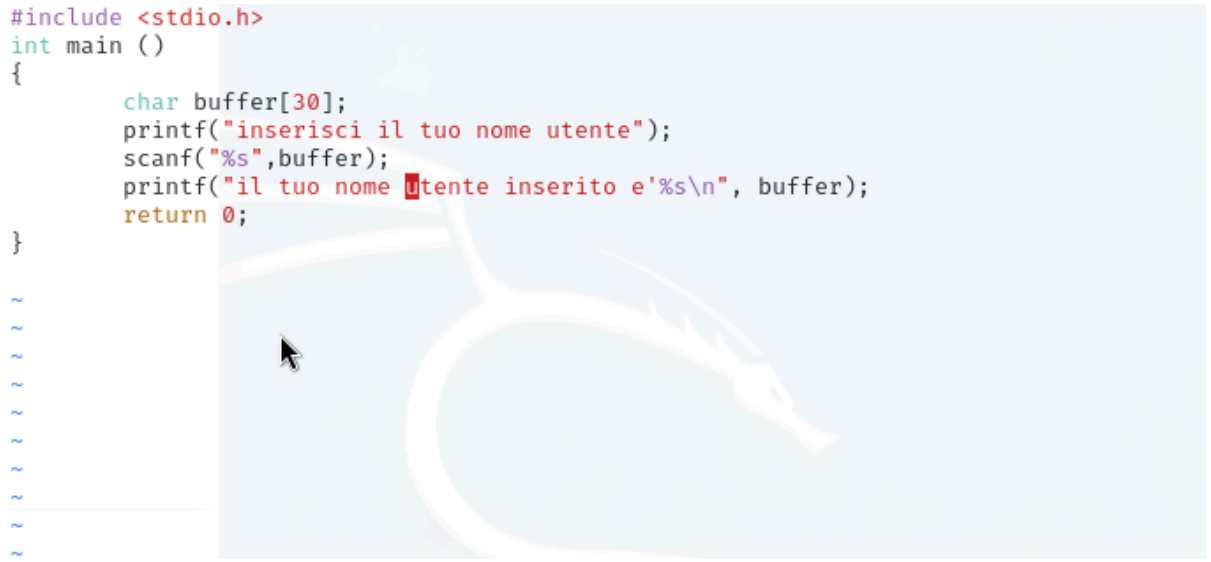
S7L4

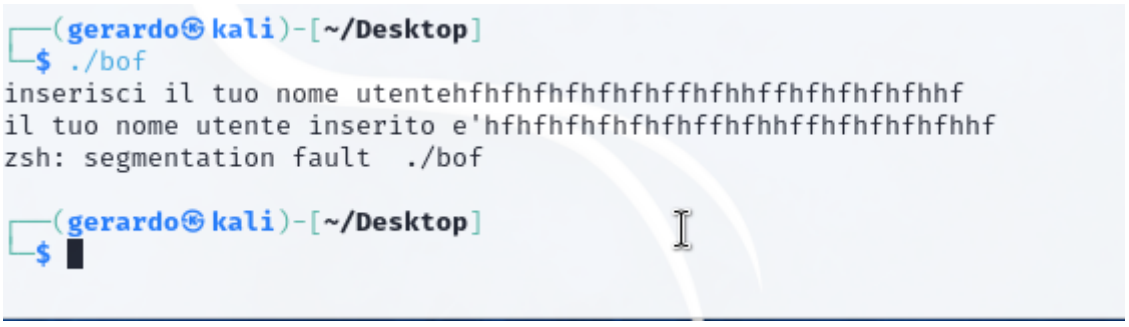
Oggi andremo a vedere come avviene un *buffer overflow* una tecnica che va a sovrascrivere il contenuto degli indirizzi di memoria,utilizzata per caricare ad esempio un malware in vari pezzi di codice.
BUFFER:indirizzo di memoria volatile della RAM

CREARE IL CODICE



```
#include <stdio.h>
int main ()
{
        char buffer[30];
        printf("inserisci il tuo nome utente");
        scanf("%s",buffer);
        printf("il tuo nome utente inserito e'%s\n", buffer);
        return 0;
}
```

BUFFER OVERFLOW



```
┌──(gerardo㉿kali)-[~/Desktop]
└─$ ./bof
inserisci il tuo nome utentehfhfhfhfhfhfhfffhfhhffhfhfhfhfhhf
il tuo nome utente inserito e'hfhfhfhfhfhfhfffhfhhffhfhfhfhfhhf
zsh: segmentation fault  ./bof

┌──(gerardo㉿kali)-[~/Desktop]
└─$
```

zsh:segmentation fault,sta ad indicare che il *buffer overflow* è avvenuto

AUMENTIAMO IL VETTORE

```c
#include <stdio.h>
int main ()
{
        char buffer[30];
        printf("inserisci il tuo nome utente");
        scanf("%s",buffer);
        printf("il tuo nome utente inserito e'%s\n", buffer);
        return 0;
}
```

Se scriviamo più di 30 caratteri vediamo cosa appare

```
inserisci il tuo nome utentegeyfgeyegfygefgyfgefgyegfegfyefgeyfgeyfgeyfgyfgefefyefegy
fgeyfgeyfyeggefeyfgyegfeyfegyfyegfyeeyfefgeyfgefgeyfgyefeygfeyfgefgygfe
il tuo nome utente inserito e'geyfgeyegfygefgyfgefgyegfegfyefgeyfgeyfgeyfgyfgefefyefe
gyfgeyfgeyfyeggefeyfgyegfeyfegyfyegfyeeyfefgeyfgefgeyfgyefeygfeyfgefgygfe
zsh: segmentation fault  ./bof
```

Come possiamo notare il buffer overflow è avvenuto con successo

SOLUZIONE

per risolvere questo problema dobbiamo sanificare l'input,per farlo utilizzeremo la funzione fgets di C.

```c
#include <stdio.h>
#include <string.h>

int main() {
    char buffer[30];

    printf("Inserisci il tuo nome utente (massimo 29 caratteri): ");

    // Legge l'input in modo sicuro utilizzando fgets
    fgets(buffer, sizeof(buffer), stdin);

    // Rimuove il carattere di nuova riga se presente
    size_t length = strlen(buffer);
    if (length > 0 && buffer[length - 1] == '\n') {
        buffer[length - 1] = '\0';
    } else {
        // Se il buffer è pieno, pulisce il buffer in eccesso
        int c;
        while ((c = getchar()) != '\n' && c != EOF);
    }

    // Stampa il nome utente inserito
    printf("Il tuo nome utente inserito è: %s\n", buffer);

    return 0;
}
```