```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ pwd
/home/kali

┌──(kali㉿kali)-[~]
└─$ cd Desktop

┌──(kali㉿kali)-[~/Desktop]
└─$ pwd
/home/kali/Desktop

┌──(kali㉿kali)-[~/Desktop]
└─$ nano BOF.c

┌──(kali㉿kali)-[~/Desktop]
└─$ gcc -g BOF.c -o BOF
```

creare il codice in C e compilare il programma

```
File  Actions  Edit  View  Help
  GNU nano 8.4
#include <stdio.h>


int main () {

char buffer [10];


printf ("Si prega di inserire il nome utente:");
scanf ("%s", buffer);

printf ("Nome utente inserito: %s\n", buffer);


return 0;

}
```

inseriamo un utente con pochi caratteri e poi con tanti caratteri , ci accorgiamo come andrà a dare errore

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inserire il nome utente:test1
Nome utente inserito: test1

┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inserire il nome utente:idfugaosihgsaoihadoiahoisdhgaoijghdsaoighaogihgioahg
Nome utente inserito: idfugaosihgsaoihadoiahoisdhgaoijghdsaoighaogihgioahg
zsh: segmentation fault  ./BOF

┌──(kali㉿kali)-[~/Desktop]
└─$ nano BOF.c

┌──(kali㉿kali)-[~/Desktop]
└─$
```

L'errore di segmentazione avviene quando un programma, come abbiamo detto in precedenza, tenta di scrivere contenuti su una porzione di memoria alla quale non ha accesso. Questo è un chiaro esempio di BOF, abbiamo inserito 30 caratteri in un buffer che ne può contenere solamente 10 e di conseguenza alcuni caratteri stanno sovrascrivendo aree di memorie inaccessibili.
Esiste un modo per non dare errore, ovvero aumentare la capienza del buffer

```
File  Actions  Edit  View  Help
  GNU nano 8.4
#include <stdio.h>


int main () {

char buffer [30];


printf ("Si prega di inserire il nome utente:");
scanf ("%s", buffer);

printf ("Nome utente inserito: %s\n", buffer);


return 0;

}
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inserire il nome utente:qqqqqqqqqqqqqqqqq
Nome utente inserito: qqqqqqqqqqqqqqqqq
```

non dà errore perchè inferiore a 30 caratteri

FACOLTATIVO

char buffer[10];: Qui creiamo un array di caratteri di dimensione 10, che può contenere al massimo 9 caratteri più il terminatore nullo \0.

scanf("%9s", buffer);: Il parametro %9s limita la lettura a 9 caratteri al massimo. In questo modo, anche se l'utente inserisce più di 9 caratteri, solo i primi 9 verranno letti. Il buffer sarà comunque terminato correttamente con il carattere nullo \0.

printf("Hai inserito: %s\n", buffer);: Visualizza la stringa letta dal buffer.