

BUFFER OVERFLOW

Ho scritto il codice in .c denominato BOF.c

```
kali@kali: ~/Desktop
File Actions Edit View Help
GNU nano 7.2 BOF.c
#include <stdio.h>

int main() {
    char buffer[10];

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

    return 0;
}
```

Con nome utente di 7 caratteri il programma non riporta alcun problema

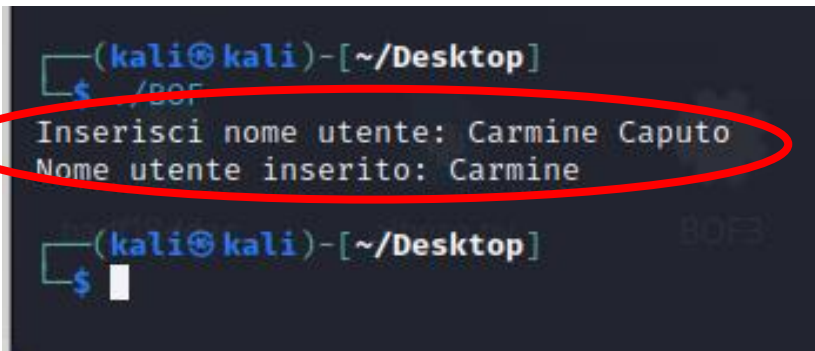
```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ gcc BOF.c -o BOF
(kali@kali)-[~/Desktop]
$ ./BOF
Inserisci nome utente:Carmine
Nome utente inserito: Carmine
(kali@kali)-[~/Desktop]
$
```

Il programma mi da la possibilità di scrivere anche 17 caratteri senza presentare alcun tipo di errore. Dopo aver inserito 18 caratteri tuttavia il programma mi presente l'errore "zsh: segmentation fault ./BOF"

```
(kali@kali)-[~/Desktop]
$ ./BOF
Inserisci nome utente:Carmineeeeeeeeeee
Nome utente inserito: Carmineeeeeeeeeee

(kali@kali)-[~/Desktop]
$ ./BOF
Inserisci nome utente:Carmineeeeeeeeeee
Nome utente inserito: Carmineeeeeeeeeee
zsh: segmentation fault ./BOF
```

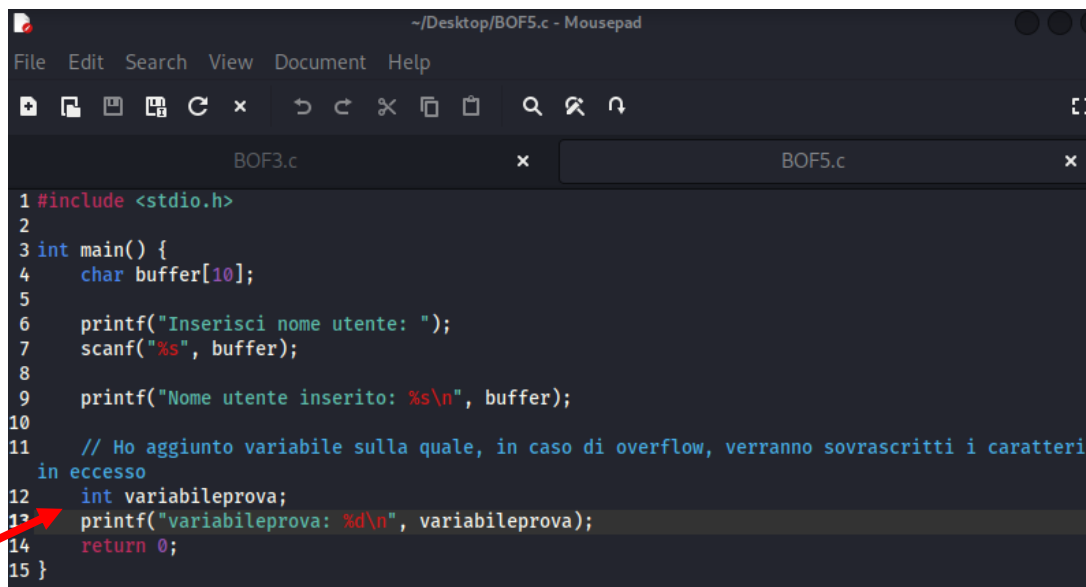
Ho notato inoltre che se inserisco nome e cognome separati da uno spazio, l'output mi restituisce soltanto i caratteri che precedono lo spazio. Questo avviene poiché "scanf" si ferma alla prima spaziatura che incontra!



```
(kali@kali)-[~/Desktop]
$ ./BOF
Inserisci nome utente: Carmine Caputo
Nome utente inserito: Carmine

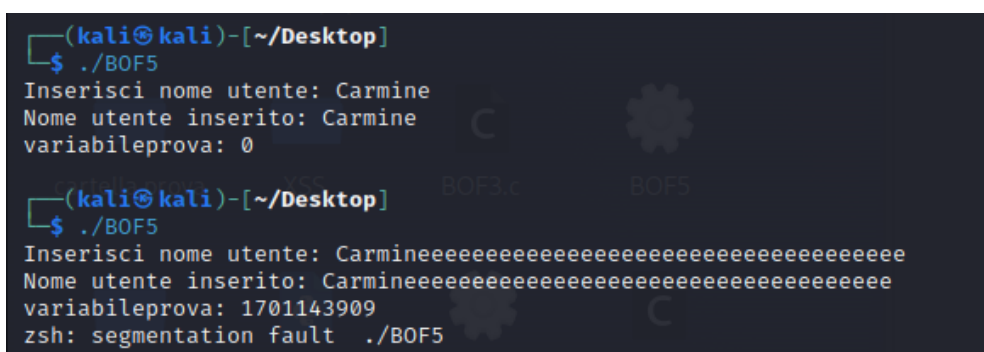
(kali@kali)-[~/Desktop]
$
```

Come da traccia ho modificato il codice per vedere dove vengono sovrascritti i caratteri in overflow. Nello specifico ho semplicemente aggiunto una seconda variabile di prova, come da screenshot in basso.



```
~/Desktop/BOF5.c - Mousepad
File Edit Search View Document Help
BOF3.c BOF5.c
1 #include <stdio.h>
2
3 int main() {
4     char buffer[10];
5
6     printf("Inserisci nome utente: ");
7     scanf("%s", buffer);
8
9     printf("Nome utente inserito: %s\n", buffer);
10
11     // Ho aggiunto variabile sulla quale, in caso di overflow, verranno sovrascritti i caratteri
    in eccesso
12     int variabileprova;
13     printf("variabileprova: %d\n", variabileprova);
14     return 0;
15 }
```

- Nel primo caso **ho inserito 7 caratteri (Carmine)** e il programma non mi ha dato alcun errore, stampando correttamente anche la variabile di prova;
- Nel secondo, **avendo inserito più caratteri di quelli consentiti**, i caratteri in eccesso hanno sovrascritto il valore della variabile di prova, con conseguente stampa "zsh: segmentation fault ./BOF5"

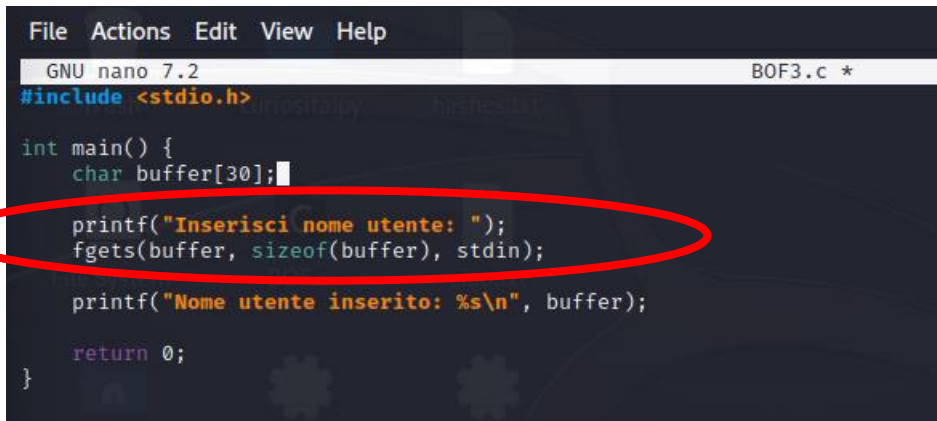


```
(kali@kali)-[~/Desktop]
$ ./BOF5
Inserisci nome utente: Carmine
Nome utente inserito: Carmine
variabileprova: 0

(kali@kali)-[~/Desktop]
$ ./BOF5
Inserisci nome utente: Carmineeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
Nome utente inserito: Carmineeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
variabileprova: 1701143909
zsh: segmentation fault ./BOF5
```

Per ovviare ai problemi sopracitati, ho modificato il codice, aumentando la dimensione del vettore a 30 (come da traccia), e sostituendo `scanf("%s, buffer");` con `fgets(buffer, sizeof(buffer), stdin);` dove:

- **Stdin**, usato argomento di fgets, indica a quest'ultimo di leggere l'input dalla tastiera;
- **Fgets** legge l'input digitato dalla tastiera e lo memorizza nell'array buffer;
- **Sizeof(buffer)**, usato come argomento di fgets, indica a quest'ultimo che in questo caso deve leggere un massimo di 29, in quanto l'ultimo è riservato al carattere terminatore di stringa).



```
File Actions Edit View Help
GNU nano 7.2 BOF3.c *
#include <stdio.h>

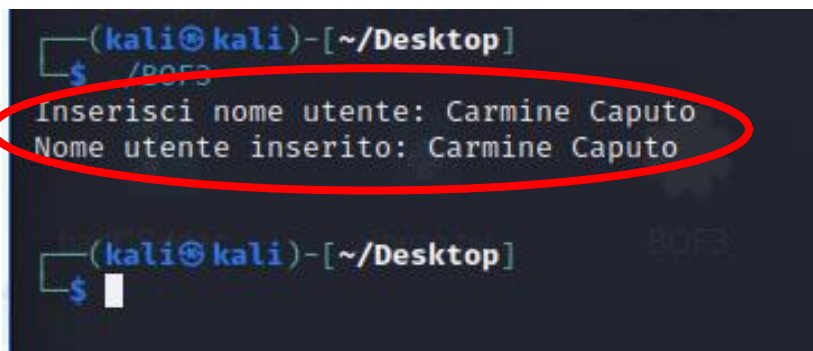
int main() {
    char buffer[30];

    printf("Inserisci nome utente: ");
    fgets(buffer, sizeof(buffer), stdin);

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

    return 0;
}
```

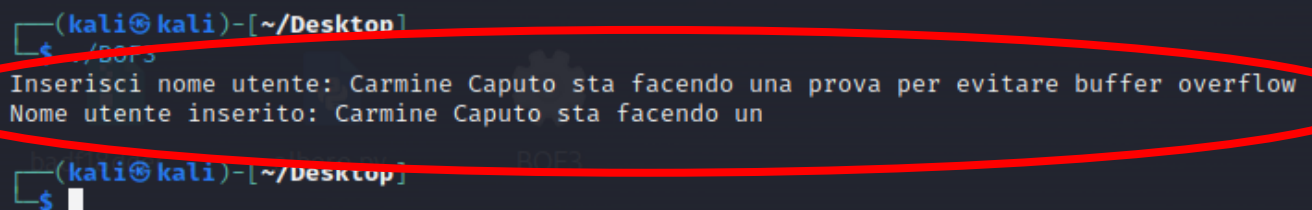
Dopo la modifica è possibile inserire tranquillamente nome e cognome separati da uno spazio



```
(kali@kali)-[~/Desktop]
$ ./BOF3
Inserisci nome utente: Carmine Caputo
Nome utente inserito: Carmine Caputo

(kali@kali)-[~/Desktop]
$
```

Con il codice corretto, ho fatto una prova inserendo più di trenta caratteri. Il programma non restituisce più l'errore `szh: segmentation fault`, poiché prende in considerazione soltanto il numero massimo di caratteri indicati nell'array evitando perciò il buffer overflow.



```
(kali@kali)-[~/Desktop]
$ ./BOF3
Inserisci nome utente: Carmine Caputo sta facendo una prova per evitare buffer overflow
Nome utente inserito: Carmine Caputo sta facendo un

(kali@kali)-[~/Desktop]
$
```

