

Oggi abbiamo creato un codice per simulare un buffer overflow.

Abbiamo inserito anche la funzione per fare vedere l'indirizzo di memoria del buffer.

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

int main() {
    char buffer[4];
    char OverFlowa[10];

    printf("Indirizzo di memoria del buffer prima dell'input: %p\n", (void*)buffer);

    char tempBuffer[14]; // Buffer più grande per evitare overflow
    printf("Inserisci una stringa: ");
    fgets(tempBuffer, sizeof(tempBuffer), stdin);

    // Copia solo i primi 4 caratteri nel buffer
    strncpy(buffer, tempBuffer, sizeof(buffer) - 1);
    buffer[sizeof(buffer) - 1] = '\0'; // Assicura la null-terminator

    // Copia il resto della stringa in OverFlowa
    strcpy(OverFlowa, tempBuffer + sizeof(buffer) - 1);

    printf("Contenuto del buffer: %s\n", buffer);
    printf("Contenuto di OverFlowa: %s\n", OverFlowa);

    // Altre operazioni o verifiche

    return 0;
}
```

[ Read 27 lines ]

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute	^C Location	M-U Undo
^X Exit	^R Read File	^_ Replace	^V Paste	^J Justify	^_ Go To Line	M-E Redo

Qua vediamo un esempio di buffer overflow, dove ci indica che caratteri sono andato in overflow.

```
(kali㉿kali)-[~/Desktop]
```

```
$ ./BOF
```

```
Indirizzo di memoria del buffer prima dell'input: 0x7ffd6e39228c
```

```
Inserisci una stringa: Ciao1234567
```

```
Contenuto del buffer: Cia
```

```
Contenuto di OverFlowa: o1234567
```

```
(kali㉿kali)-[~/Desktop]
```

```
$ █
```