

S7_L4

Unit 2 - CS0424

MATTEO BELTRAMI MARZOLINI
CYBEREAGLES

BUFFER OVERFLOW

TRACCIA

Abbiamo già parlato del buffer overflow, una vulnerabilità che è conseguenza di una mancanza di controllo dei limiti dei buffer che accettano input utente.

Nelle prossime slide vedremo un esempio di codice in C volutamente vulnerabile ai BOF, e come scatenare una situazione di errore particolare chiamata «segmentation fault», ovvero un errore di memoria che si presenta quando un programma cerca inavvertitamente di scrivere su una posizione di memoria dove non gli è permesso scrivere (come può essere ad esempio una posizione di memoria dedicata a funzioni del sistema operativo).

Provate a riprodurre l'errore di segmentazione modificando il programma come di seguito:

- Aumentando la dimensione del vettore a 30;

SVOLGIMENTO

BOF.c

Dopo aver acceso la mia macchina kali, apro il terminale e mi sposto sulla home tramite il comando:

cd /home/Kali/Desktop

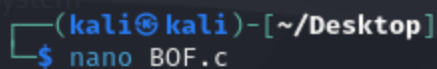


```
(kali@kali)-[~]  
$ cd /home/kali/Desktop
```

Ora creo il mio file in formato c per scrivere dentro l'algoritmo che vulnerabile ai BOF.

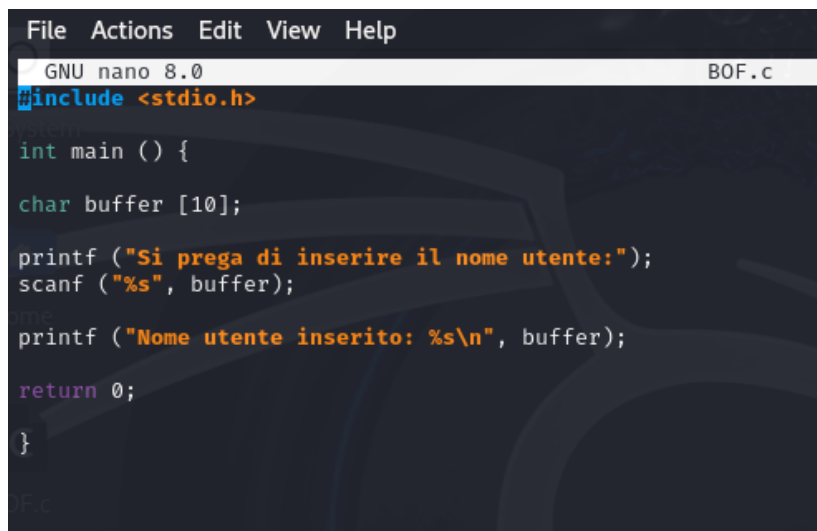
Procedo con il comando:

nano BOF.c



```
(kali@kali)-[~/Desktop]  
$ nano BOF.c
```

e, all'interno, inserisco il mio algoritmo:



```
File Actions Edit View Help  
GNU nano 8.0 BOF.c  
#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;  
}
```

Una volta completato e salvato, compilo il file con il comando:

gcc -g BOF.c -o BOF

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

(kali㉿kali)-[~/Desktop]
$ ./BOF
Si prega di inserire il nome utente:█
```

Successivamente avvio il programma con il comando:

./BOF

Ora provando il programma, come richiesto dalla traccia, inserendo un numero di caratteri uguali a 5, il mio risultato è positivo,

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

ma se inserisco un numero superiore a 10 (in questo caso 30) riceverò un messaggio d'errore: **segmentation fault**

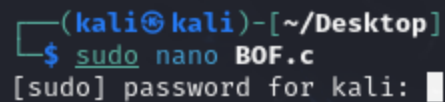
```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Si prega di inserire il nome utente:iohfoaihfoiahoiahofiahfoahfaoh
Nome utente inserito: iohfoaihfoiahoiahofiahfoahfaoh
zsh: segmentation fault ./BOF
```

La traccia mi richiede di aumentare la dimensione del vettore a 30, perchè in questo caso mi esce fuori il messaggio d'errore.

La mia soluzione riguardo questo problema, ovvero di aumentare la dimensione del vettore a 30 è stato risolto aumentando il Buffer.

Procedo rientrando nel file BOF.c per fare le mie modifiche:

sudo nano BOF.c



```
(kali㉿kali)-[~/Desktop]
$ sudo nano BOF.c
[sudo] password for kali: 
```

Entrato all'interno del file procedo alla modifica, sostituendo:

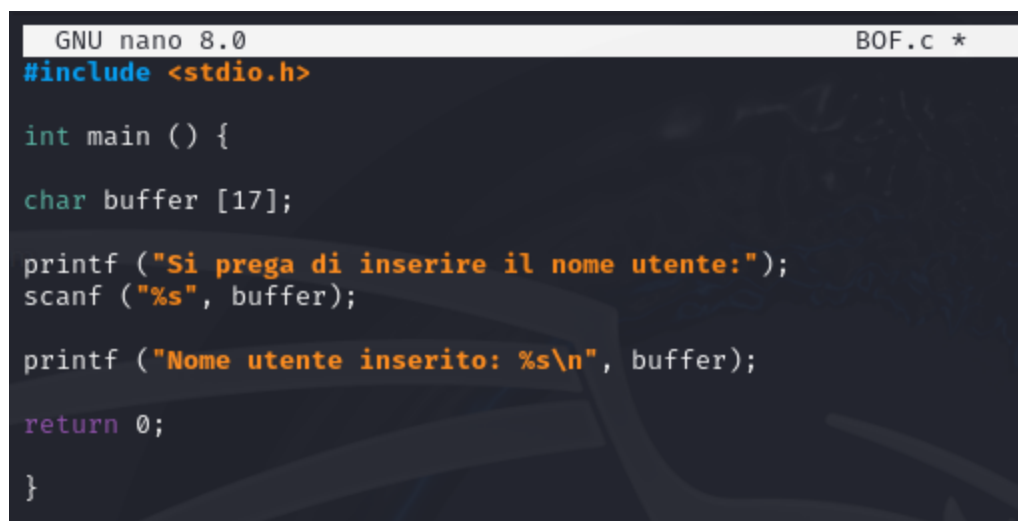
char buffer [10];

con

char buffer [17];

E' stato scelto il valore 17 dopo vari tentativi.

Con il valore uguale a 16 il buffer non bastava per un valore di caratteri uguale a 30, con 17 li supera tranquillamente.



```
GNU nano 8.0 BOF.c *
#include <stdio.h>

int main () {
    char buffer [17];
    printf ("Si prega di inserire il nome utente:");
    scanf ("%s", buffer);
    printf ("Nome utente inserito: %s\n", buffer);
    return 0;
}
```

Infine, compilo nuovamente il file con il comando:

gcc -g BOF.c -o BOF

e avvio nuovamente il programma.

CONCLUSIONE

Come mostra l'immagine sotto, inserendo un numero di caratteri di 30 cifre, non uscirà il messaggio di errore: **segmentation fault**



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

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