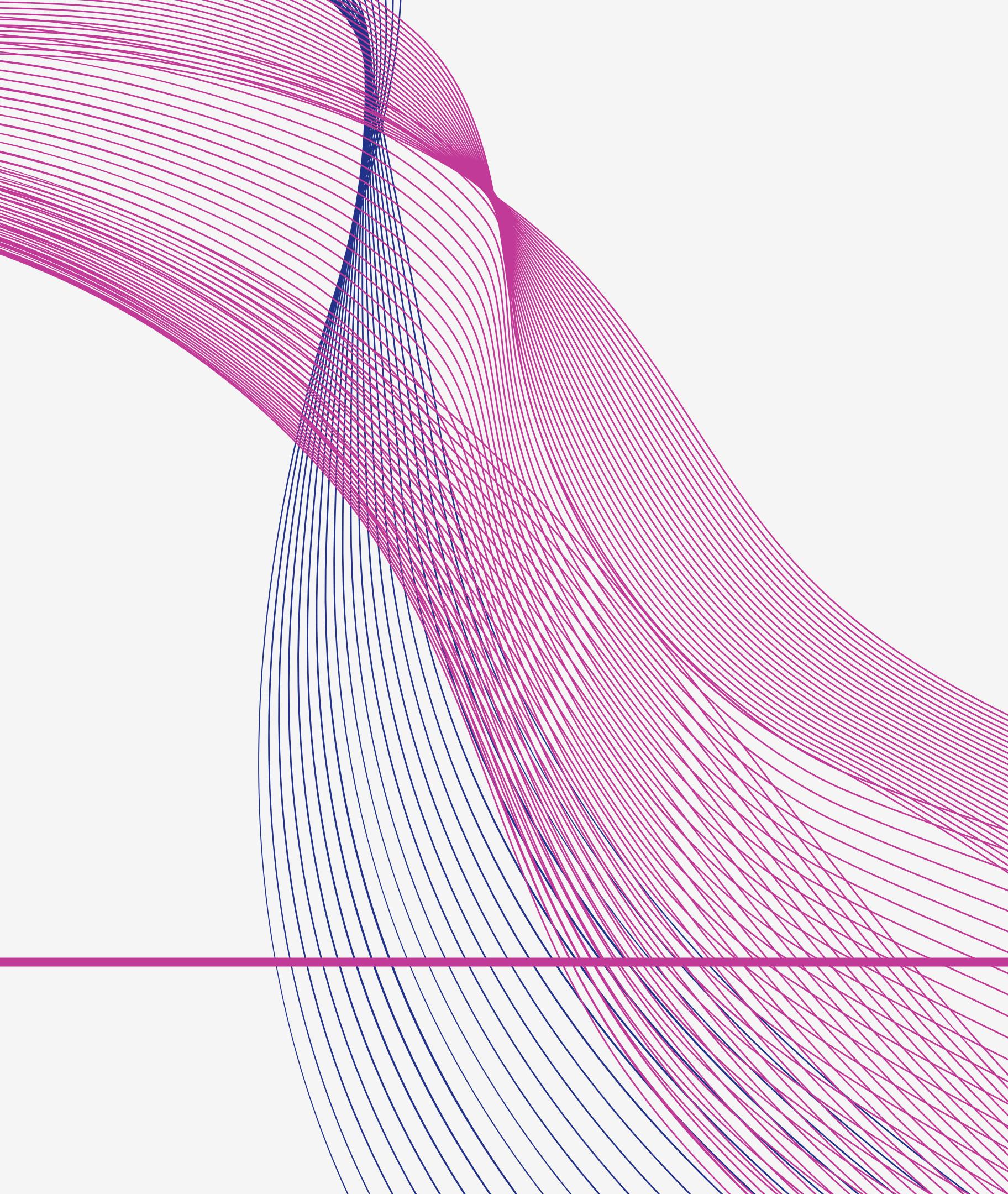


S7/L4

Noemi de Martino





TRACCIA

Dato un codice .c che cerca di riprodurre un errore di **segmentazion fault** a causa del **buffer overflow**, si cerca di riprodurre lo stesso errore aumentando la dimensione del parametro di input.

BUFFER OVERFLOW

Il **buffer overflow** è una vulnerabilità comune nei programmi C (e in altri linguaggi di programmazione) che si verifica quando viene scritto oltre i limiti di un buffer, ovvero una zona di memoria allocata per contenere dati. In C, i buffer sono spesso implementati come array di caratteri (stringhe) o altri tipi di dati.

Quando un programma scrive più dati in un buffer di quanto esso possa contenere, i dati in eccesso sovrascrivono le aree di memoria adiacenti al buffer. Questo può causare comportamenti imprevisti e, in casi più gravi, vulnerabilità di sicurezza come l'esecuzione di codice dannoso o l'accesso a zone di memoria critiche.

Causa del Segmentation Fault

Un **segmentation fault** (o **segfault**) è un errore che si verifica quando un programma tenta di accedere a una parte della memoria alla quale non ha il permesso di accedere. Questo può includere tentativi di scrittura su una memoria non allocata per il programma, tentativi di scrittura su una zona di memoria protetta dal sistema operativo o altre situazioni in cui il programma opera in modo non sicuro con la memoria.

Creazione del programma BOF.C

- Creazione di un semplice programma in C chiamato **BOF.c** progettato per accettare una stringa di input.
- Limitazione della dimensione massima dell'input a 10 caratteri per simulare un buffer overflow.

```
.vscode > C BOF.c
1 #include<stdio.h>
2
3 int main () {
4
5     char buffer [10];
6
7     printf("Si prega di inserire il nome utente: ");
8     scanf("%s", buffer);
9
10    printf("Nome utente inserit: %s\n", buffer);
11
12    return 0;
13
14 }
```

Esecuzione BOF.C

Cosa succede durante il Buffer Overflow:

- Quando inseriamo un input di 30 caratteri (ad esempio "123456789012345678901234567890"), gets() copia tutti questi caratteri nel buffer buffer[11].
- Poiché buffer ha spazio solo per 11 byte, i restanti 19 byte dell'input oltre il 10° carattere (che è la fine del buffer) sovrascrivono le aree di memoria adiacenti.

Questo risulta in **segmentation fault**, che è un errore di accesso alla memoria.

Il sistema operativo può rilevare che il programma sta tentando di accedere a una parte di memoria non assegnata a esso o che non è consentita per la scrittura, e interrompe quindi l'esecuzione del programma.

```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Si prega di inserire il nome utente: 1234567894
Nome utente inserit: 1234567894
```

Output di un input con 10 caratteri

```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Si prega di inserire il nome utente: 123456789123456789123456789123456789123456789
Nome utente inserit: 123456789123456789123456789123456789123456789
zsh: segmentation fault  ./BOF
```

Output di un input con caratteri superiori a 10

Modifica del programma **BOF.C**

Modifica del programma **BOF.c** per aumentare la dimensione del buffer a 30 caratteri per consentire un input più lungo:

```
.vscode > C BOF.c
1 #include<stdio.h>
2
3 int main () {
4
5     char buffer [30];
6
7     printf("Si prega di inserire il nome utente: ");
8     scanf("%s", buffer);
9
10    printf("Nome utente inserito: %s\n", buffer);
11
12    return 0;
13 }
```

Esecuzione BOF.C

Output di un input minore di 30 caratteri.

```
(kali㉿kali)-[~/Desktop]  
$ ./BOF
```

Si prega di inserire il nome utente: gattinicoccolosi
Nome utente inserit: gattinicoccolosi

Anche in questo caso l'esecuzione
risulta in un **segmentation fault** con un
input maggiore di 30 caratteri.

```
(kali㉿kali)-[~/Desktop]  
$ ./BOF
```

Si prega di inserire il nome utente: Gattinicoccolosiportanogioia
Nome utente inserit: Gattinicoccolosiportanogioia
zsh: segmentation fault ./BOF