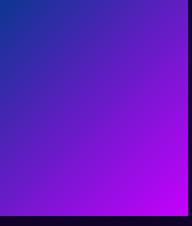


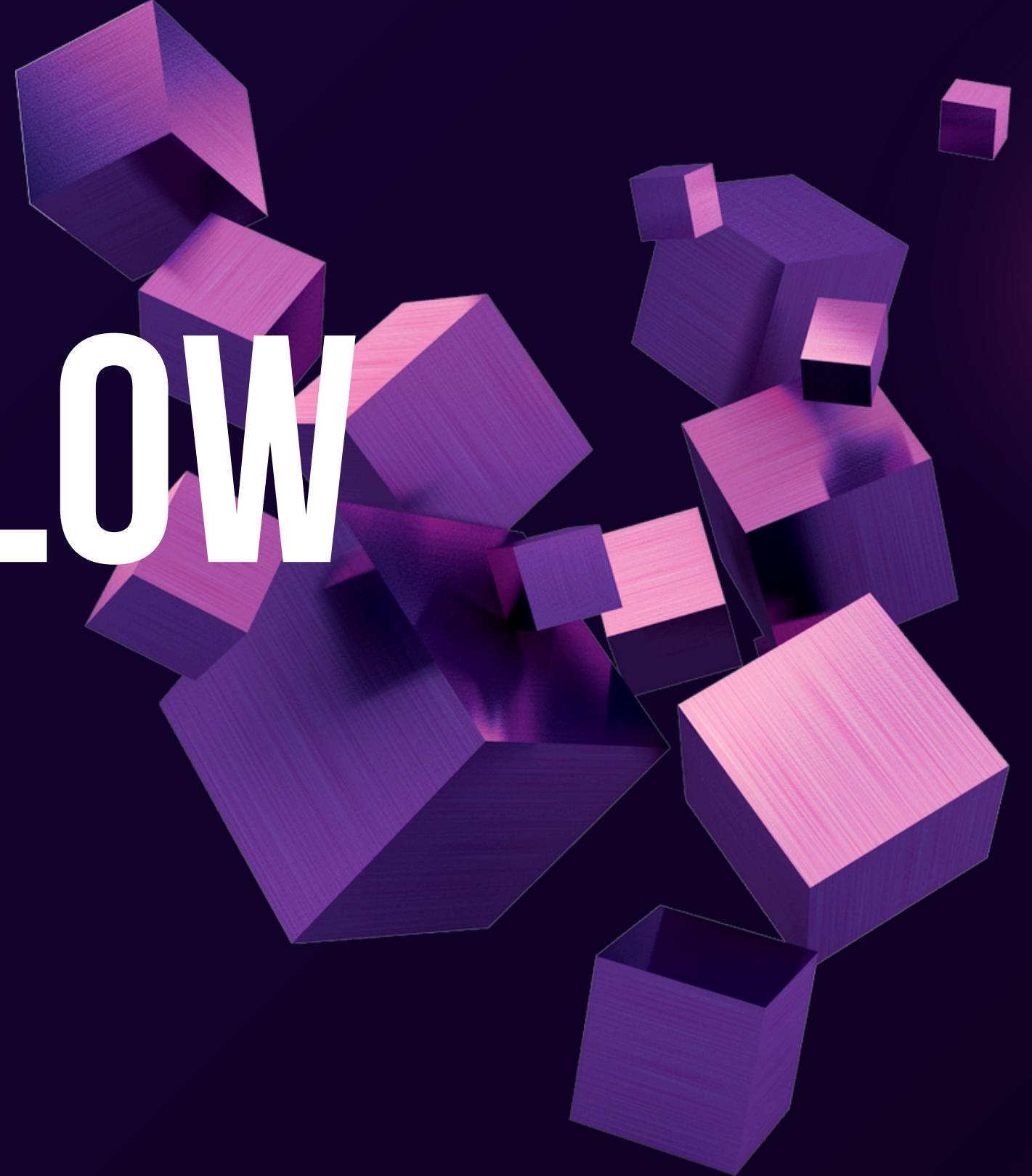


Glulia fiacchi



BUFFER OVERFLOW

S7/L4



TRACCIA

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).

In più viene richiesto di provare a riprodurre l'errore di segmentazione modificando il programma come di seguito:

- Aumentando la dimensione del vettore a 30;

```
→ (kali㉿kali)-[~]
└─$ cd /home/kali/Desktop

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

```
GNU nano 8.0
#include <stdio.h>
int main () {
char buffer [10];
printf ("Quale è il tuo nome:");
scanf ("%s", buffer);
printf ("Nome inserito: %s\n", buffer);
return 0;
}
```

PARTE 1

CREAZIONE DELLA CARTELLA

Avviamo la macchina Kali, e dopodiché creiamo un file chiamato “BOF.c” che conterrà il codice.

In questo caso vedremo stampato “Inserisci il tuo nome:” e dovremmo inserire una frase con massimo 10 caratteri perchè sono quelli specificati nella lunghezza dell’array. Se superiamo questa cifra si verificherà un buffer overflow.

Una volta salvato il file è necessario effettuare il comando “`gcc -g BOF.c -o BOF`” per compilare il file.

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

PARTE 2

ESECUZIONE DEL PROGRAMMA

Una volta creato il file, possiamo lanciare il codice con il comando “./BOF” e ci verrà richiesto di inserire il nostro nome.

E’ stata inserita la parola ”Giulia” ed è stato possibile osservare che non succede nulla perchè la parola rispetta il max consentito di caratteri specificati nel codice.

```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Quale è il tuo nome:■
```

```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Quale è il tuo nome:GIULIA
Nome inserito: GIULIA
```

PARTE 2

ESECUZIONE DEL PROGRAMMA

D'altro canto se si prova ad inserire parole più lunghe come “abcdefghijkl” il programma ci restituisce l'errore «**segmentation fault**», ovvero errore di segmentazione. 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.



```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Quale è il tuo nome:ABCDEFGHIJKLMNPQRSTUVWXYZ
Nome inserito: ABCDEFGHIJKLMNOPQRSTUVWXYZ
zsh: segmentation fault ./BOF
```

PARTE 3

MODIFICA DEL CODICE

```
GNU nano 8.0
#include <stdio.h>
int main () {
char buffer [30]; ←
printf ("Quale è il tuo nome:");
scanf ("%s", buffer);

printf ("Nome inserito: %s\n", buffer);
return 0;
}
```

Come richiesto dalla traccia si è poi modificato il codice aumentando la dimensione del vettore a 30 caratteri, quindi si è riaperto il file con il comando “nano BOF.c”, modificato, salvate le modifiche e poi eseguito il comando per la compilazione.

Si sono fatte poi varie prove e osservato che come per quando la dimensione del vettore era a 10, se non si supera la cifra indicata nel codice, ora a 30, non si verificherà un buffer overflow.

```
(kali㉿kali)-[~/Desktop]
$ ./BOF
Quale è il tuo nome:ABCDEFGHIJKLMNOPQRSTUVWXYZ
Nome inserito: ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

