S7L4

Sarah Ortiz

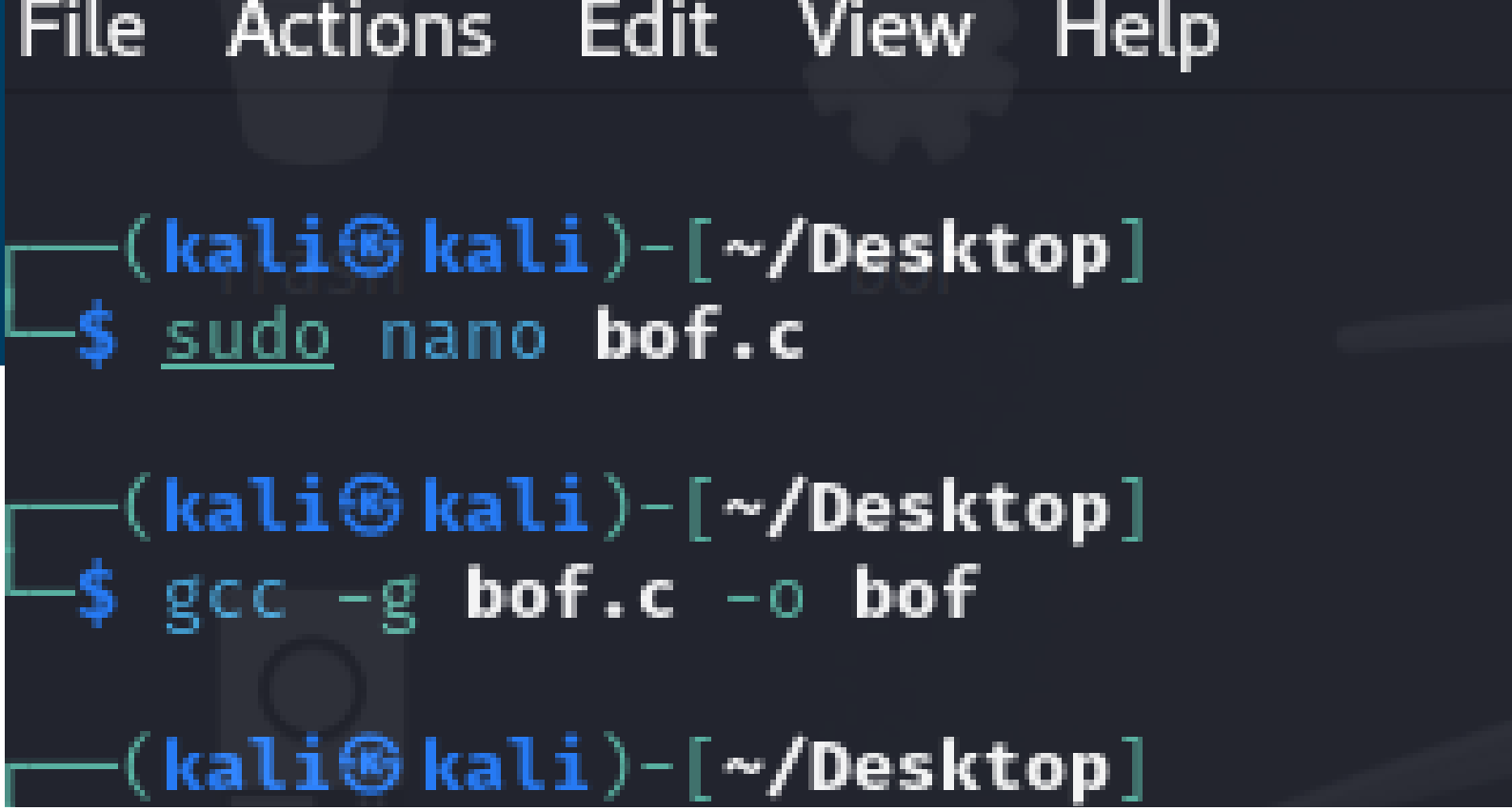
Traccia

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)

Seconda parte della traccia:

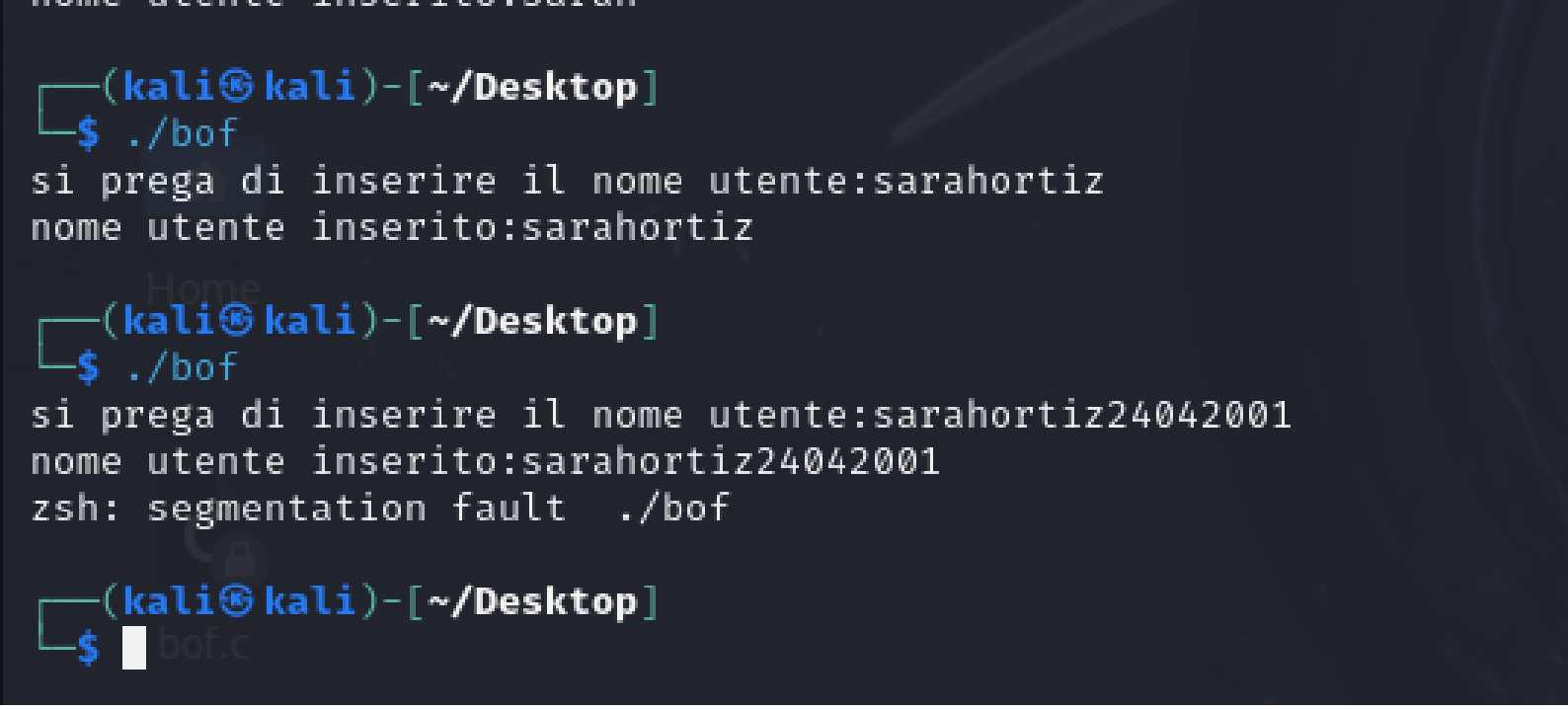
Provate a riprodurre l'errore di segmentazione modificando il programma come di seguito:

- Aumentando la dimensione del vettore a 30;

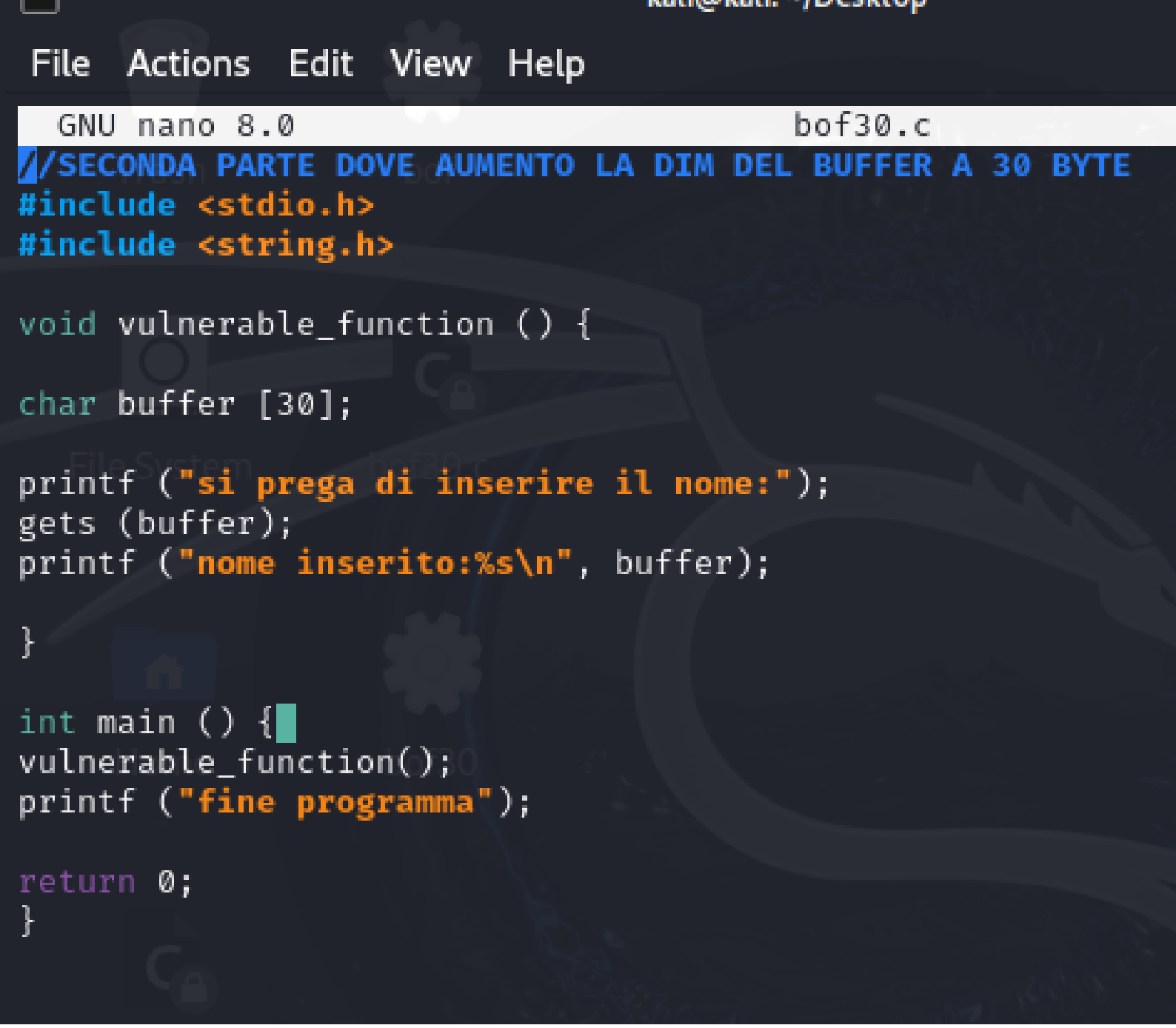


Per prima cosa va creato un file tramite il comando:
“sudo nano bof.c”,
dopo di che con il comando
“gcc -g bof.c -p bof”
facciamo invocare il compilatore GNU per il linguaggio C e includiamo informazioni di debug nel file oggetto generato.

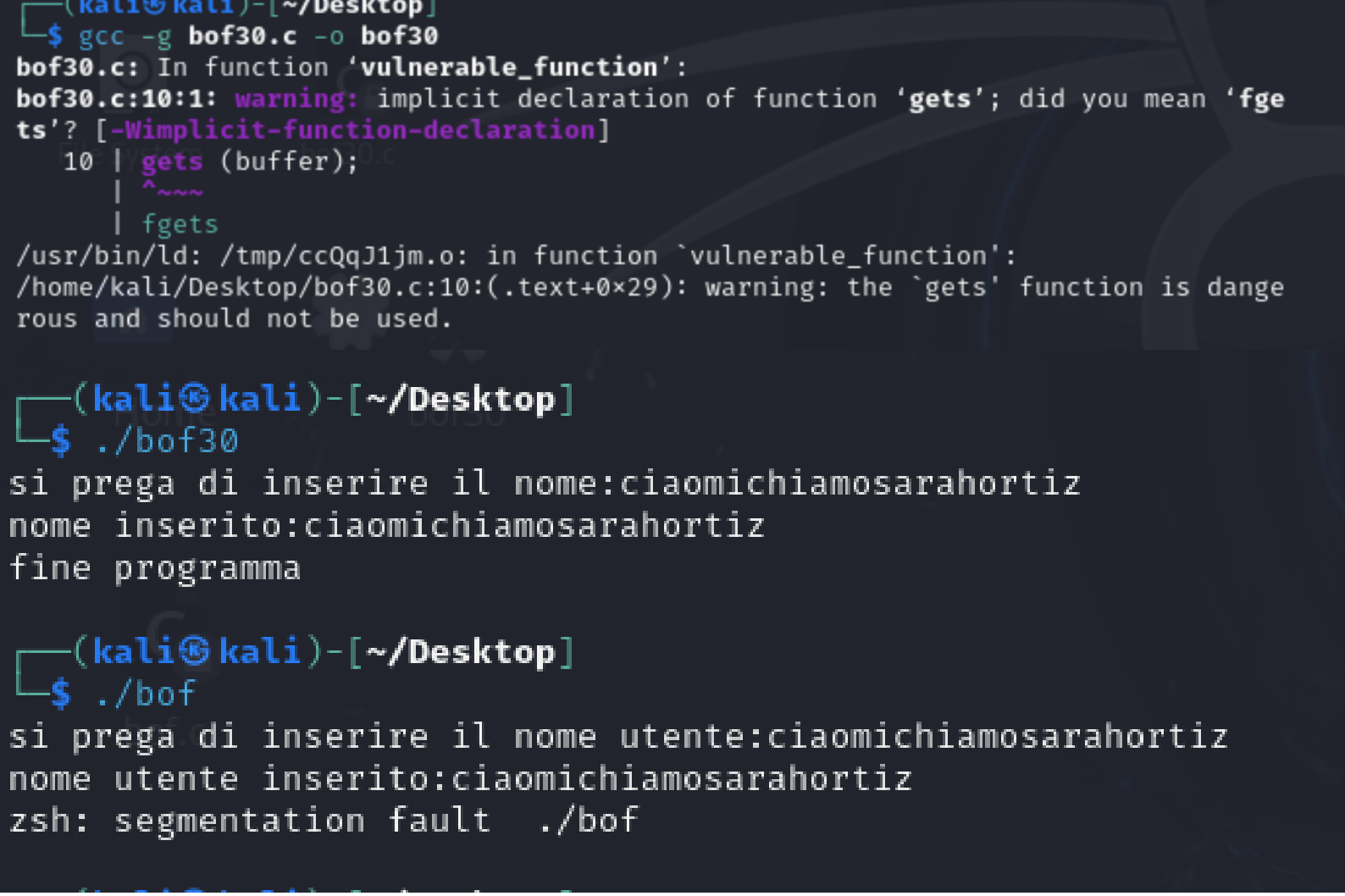

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./bof
si prega di inserire il nome utente:sarahortiz
nome utente inserito:sarahortiz

┌──(kali㉿kali)-[~/Desktop]
└─$ ./bof
si prega di inserire il nome utente:sarahortiz24042001
nome utente inserito:sarahortiz24042001
zsh: segmentation fault  ./bof

┌──(kali㉿kali)-[~/Desktop]
└─$
```



Possiamo notare come dipendendo da byte utilizzati possiamo causare un buffer oppure no. Nel secondo esempio l'input supera i 10 byte del buffer, causando un overflow e, potenzialmente, scrivendo su aree di memoria non consentite.



Per la seconda parte della traccia richiesta, creiamo un nuovo file con il codice che aumenta la dimensione del buffer a 30 byte



Ho messo a confronto entrambi i file con i codici di buffer diversi e possiamo notare come il prima con il buffer a 30 byte non presenta errori