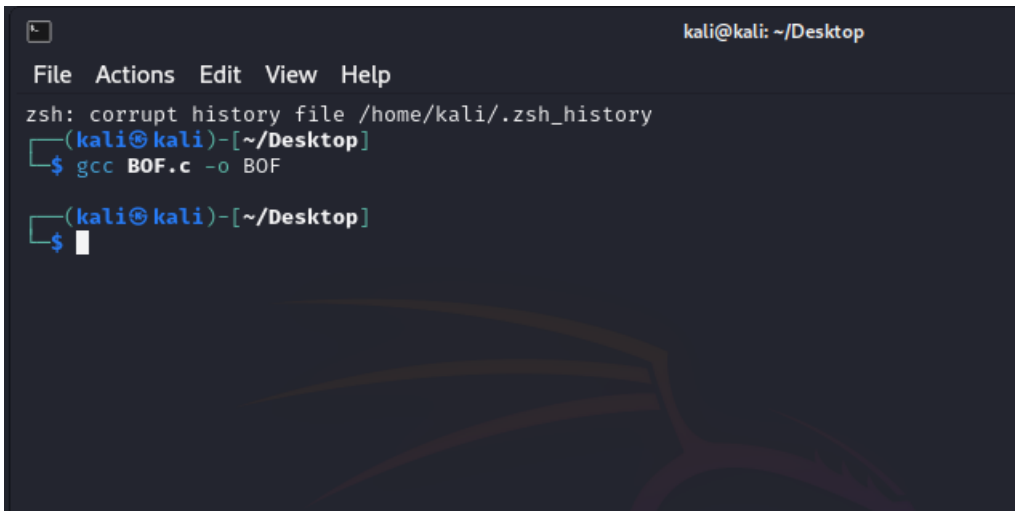
Nell'esercizio di oggi vediamo nel pratico il funzionamento del Buffer overflow, una vulnerabilità causata da errori di programmazione. Nello specifico l'errore avviene poiché viene messo un tetto massimo ai caratteri che il programma prende in input, ma se l'utente inserisce più caratteri di quelli previsti, i valori in più vanno ad occupare aree della RAM destinate inizialmente ad altri processi.

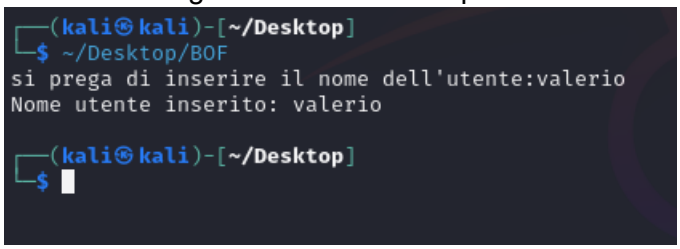
Per l'esperimento ci serviremo di un programma molto semplice in linguaggio c, che non fa altro che restituire un nome che noi gli diamo in input. NOTA BENE: il parametro buffer è stato impostato a 10, questo significa che, una volta eseguito il programma, questo si aspetterà di ricevere un nome composto al massimo da 10 caratteri. Se l'input dovesse eccedere anche solo di un carattere, la RAM non sarebbe in grado di gestirlo ed il pc crasherebbe.

Ricordiamo che per ogni processo il pc alloca solamente la memoria necessaria (o che crede sia necessaria).

Una volta creato il codice con il comando "nano", lo salviamo e lo compiliamo con il comando "gcc –g BOF.c –o BOF" in modo che possa essere eseguito.

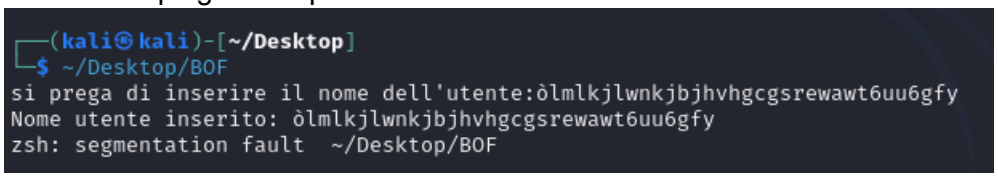


Possiamo eseguire il file sul desktop con il comando “./BOF” ed inserire il nostro nome.



Il comando funziona poiché il nome inserito è entro i 10 caratteri.

Ma se inseriamo un input superiore otteniamo un errore legato proprio alla segmentazione della memoria impiegata nel processo.



Per settare questo errore basterà semplicemente cambiare il valore di input del buffer (proviamo ad impostarlo a 30 invece che a 10) e a dargli un input di 22 caratteri.

Vediamo che non si verifica più il buffer overflow.