



REPORT

BUFFER

OVERFLOW



Alejandro Cristino
S7 L4

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.

Successivamente 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).

Codice di esempio in C

Ecco un esempio di codice in C vulnerabile a buffer overflow:

```
#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;  
}
```

Procedura

1. Creazione di un documento su Kali Linux

Per creare un nuovo documento su Kali, avviate la vostra Kali Linux, ed una volta presentata la schermata principale cliccate sull'icona del terminale.

Dal terminale, poi, spostatevi sul desktop eseguendo il comando:

```
cd /home/Kali/Desktop
```

Successivamente, eseguiamo l'editor di testo nano, che ci permette di o aprire un file esistente oppure di crearne uno nuovo se il nome del file specificato non esiste.

Eseguiamo quindi dal terminale il comando di seguito per creare un file BOF.c

```
nano BOF.c
```

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

```
File Azioni Modifica Visualizza Aiuto
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;
}
```

2.Salvare il file

Una volta completato, chiudete e salvate il file. Per chiudere e salvare un file con l'editor di testo nano, dovete seguire una sequenza di comandi da tastiera:

1. Premete insieme i tasti Ctrl e x sulla vostra tastiera.
2. Il tool vi chiederà se volete salvare le modifiche. Digitate 'Y' e poi premete 'Invio' per salvare il file con il nome scelto.

3.Compilare il programma

A questo punto, compilate il file utilizzando il comando:

gcc -g BOF.c -o BOF

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

4.Eseguire il programma

Una volta fatto, potete eseguire il programma eseguendo il comando:

./BOF

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

4.Test del programma

Nel primo caso, il buffer è impostato per accettare 10 caratteri, quindi qualsiasi input superiore a questa lunghezza causerà un buffer overflow. Tuttavia, in questo caso specifico, viene mostrato che anche un input di 17 caratteri non causa un errore immediato. Questo potrebbe essere dovuto al comportamento specifico del programma o al sistema in uso.


Nel secondo caso, l'inserimento di 18 caratteri causa un segmentation fault, che è un chiaro segnale che il buffer è stato sovrascritto al punto di accedere a una memoria non autorizzata, causando l'errore di segmentazione.

Primo caso

```
(kali㉿kali)-[~/Desktop]
$ ./BOF

Si prega di inserire il nome utente:aaaaaaaaaaaaaaaaaaaaa
Nome utente inserito: aaaaaaaaaaaaaaaaaaaaaa
```

Secondo caso




```
(kali㉿kali)-[~/Desktop]
$ ./BOF

Si prega di inserire il nome utente:aaaaaaaaaaaaaaaaaaaaa
Nome utente inserito: aaaaaaaaaaaaaaaaaaaaaa
zsh: segmentation fault ./BOF
```

1.Modifica del programma

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

Aumentando la dimensione del vettore a 30



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

int main() {
    char buffer[30];
    printf("Si prega di inserire il nome utente:");
    scanf("%s", buffer);
    printf("Nome utente inserito: %s\n", buffer);
    return 0;
}
```

4.Test del programma

Nel primo caso, il buffer è impostato per accettare 30 caratteri, quindi qualsiasi input superiore a questa lunghezza causerà un buffer overflow. Tuttavia, in questo caso specifico, viene mostrato che anche un input di 39 caratteri non causa un errore immediato. Questo potrebbe essere dovuto al comportamento specifico del programma o al sistema in uso.

Nel secondo caso, l'inserimento di 40 caratteri causa un segmentation fault, che è un chiaro segnale che il buffer è stato sovrascritto al punto di accedere a una memoria non autorizzata, causando l'errore di segmentazione.

Primo caso

[illegible]

Secondo caso

[illegible]