W17D4 – Pratica

Epic Education Srl

Buffer Overflow

Simone Giordano

05/11/2025

Contatti:

Tel: 3280063044
Email: mynameisimone@gmail.com
Linkedin: https://www.linkedin.com/in/simone-giordano-91290652/

Sommario

Sintesi esecutiva

Durante il test è stato analizzato un programma scritto in linguaggio C per individuare e comprendere la vulnerabilità di tipo **Buffer Overflow (BOF)**.
Il test ha mostrato come un input non controllato possa sovrascrivere aree di memoria non destinate a contenerlo. Sono state quindi applicate modifiche al codice e introdotti meccanismi di validazione dell'input per mitigare il rischio.

Perimetro

L'analisi ha riguardato un **programma C** con un buffer dimensionato per contenere un numero limitato di caratteri.

Panoramica delle vulnerabilità

La vulnerabilità identificata è un **Buffer Overflow** causato da un uso improprio della funzione scanf("%s", buffer); senza limiti di lunghezza dell'input.

- Il programma consente all'utente di inserire più dati di quelli che il buffer può gestire.
- Quando l'input supera la capacità dell'array, la memoria adiacente viene sovrascritta, portando a errori di segmentazione o comportamenti indefiniti.
- L'aumento della dimensione dell'array (da 10 a 30) riduce la probabilità di crash ma non elimina la vulnerabilità, poiché l'overflow può comunque verificarsi se si supera la nuova soglia.

Azioni di rimedio

Per mitigare la vulnerabilità è stata implementata la limitazione dell'input per mezzo del modificatore di formato in scanf (es. %9s) per specificare il numero massimo di caratteri accettabili, prevenendo l'eccesso rispetto alla dimensione del buffer.

Buffer overflow

Traccia

Provate a riprodurre l'errore di segmentazione modificando il programma come di seguito:
●Aumentando la dimensione del vettore a 30.
Questo basta ad eliminare la possibilità di generare un 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;

}
```

Esercizio

Aumento la dimensione dell'array a 30 aumentando il parametro come da immagine riportata di seguito.

```c
#include <stdio.h>

int main () {

char buffer [30];

printf ("Si prega di inseirre il nome utente:");
scanf ("%s", buffer);

printf ("Nome utente inserito: %s\n", buffer);

return 0;

}
```

Compilo di nuovo il programma in C.

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

Ho inserito 20 caratteri e l'input è stato acquisito correttamente perché la capienza è aumentata da 10 a 30.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inseirre il nome utente:qqqqqqqqqqqqqqqqqqqq
Nome utente inserito: qqqqqqqqqqqqqqqqqqqq
```

Se inserisco oltre 30 elementi si verificherà di nuovo il buffer overflow.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq
qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq: command not found
```

Esercizio facoltativo – Prevenzione buffer overflow

Traccia

Aggiungete nel programma precedente dei controlli di sicurezza per prevenire il BOF e sanitizzare l'input.

Esercizio

%9s specifica che la funzione deve leggere al massimo 9 caratteri per evitare che si superi la capacità del buffer, proteggendo così da eventuali overflow di memoria.

```c
#include <stdio.h>

int main () {

char buffer [30];

printf ("Si prega di inseirre il nome utente:");
scanf ("%9s", buffer);

printf ("Nome utente inserito: %s\n", buffer);

return 0;

}
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inseirre il nome utente:rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr
Nome utente inserito: rrrrrrrr

┌──(kali㉿kali)-[~/Desktop]
└─$ ./BOF
Si prega di inseirre il nome utente:ggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
Nome utente inserito: ggggggggg
```

Bonus balckbox Jangow 01 – W16D1

Traccia

Scaricare ed importare la macchina virtuale da questo link: VirtualBox:
https://download.vulnhub.com/jangow/jangow-01-1.0.1.ova
Effettuare gli attacchi necessari per diventare root. Studiare a fondo la macchina per scoprire tutti i segreti.
L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è test di BlackBox puro.

Esercizio

Prova di accesso root!