НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА ПРОГРАМУВАННЯ»

«Безпечне програмування»

*Звiт з лабораторної роботи №3*

*Тема: «Редагування виконуваного файлу»*

Виконав:

ст. гр. 16Б

Башкатов В.Є.

.

Харків – 2019

***Завдання***

1. Створити функцію, що виводить переданий рядок на екран. Реалізувати виклик даної функції безпосередньо myFunc (" Hello World ") і через покажчик на функцію.
2. Визначте розмір кода виклику з урахуванням занесення аргументів в стек для обох випадків.
3. За допомогою hex редактора (напр. \* Hexedit \*) занулити виклики функцій, щоб їх виконання пропускалося.
4. Створити функцію, яка вводить пароль з клавіатури і порівнює його з "еталонним".
5. Змінити асемблерний (машинний) код програми таким чином, щоб при введенні невірного пароля програма "думала", що пароль вірний і надавала доступ для подальшої роботи.
6. У звіті навести результати дослідження та кроки проведення змін файлу, що виконується.

**Текст програми**

#include <stdio.h>

#include <string>

#include <iostream>

using namespace std;

void myFunctionPrint(const char str[])

{

printf("Hello from myFunctionPrint,input str is>> %s\n", str);

}

int main()

{

//1

myFunctionPrint("Hello World");

//2

void(\*myFunc)(const char[]);

myFunc = myFunctionPrint;

myFunc("Hello World Number2");

//3

char password[] = "12345";

char inptpass[80];

int counter = 3;

do

{

if (counter == 0)

{

printf("Get out!\n");

system("pause");

return 0;

}

printf("Input password(you have %d chanse)>>", counter);

cin >> inptpass;

} while (strcmp(inptpass, password) != 0 && counter--);

printf("\nPassword is correct!Welcome!\n");

system("pause");

return 0;

}

Хід роботи:

На рис.1 видно, що розмір функції, що виводить переданий рядок на екран з урахуванням занесення аргументів в стек дорівнює 16 байт.



Рисунок 1– розмір функції myFunctionPrint(“Hello World”)

На рис.2 видно, що розмір функції яка реалізована через покажчик на функцію , що виводить переданий рядок на екран з урахуванням занесення аргументів в стек дорівнює 10 байт.



Рисунок 2– розмір функції, яка реалізована через покажчик на функцію

На рис.3 представлено результат виконання обох функцій



Рисунок 3 – результат виконання обох функцій

Для занулення першої функції знаходимо дану послідовність,(рис.4) у виконуваному файлі за допомогою Hex Editor та замінюємо її командами nop(0x00)



Рисунок 4 – послідовність виклику першої функції

Результат занулення послідовності виклику першої функції показано на рис.5



Рисунок 5 – занулення послідовності виклику першої функції

Для занулення другої функції, яка реалізована через покажчик на функцію знаходимо дану послідовність,(рис.6) у виконуваному файлі за допомогою Hex Editor та замінюємо її командами nop(0x00)



Рисунок 6 – послідовність виклику другої функції

Результат занулення послідовності виклику функції, яка реалізована через покажчик на функцію показано на рис.7



Рисунок 7 – занулення послідовності виклику другої функції

Результат роботи програми після занулення виклику обох функцій представлено на рис.8



Рисунок 8 – результат занулення виклику обох функцій

Як видно з рис.8 при зануленні виклику функцій вони не викликались, результат успішний.

Для того, щоб при введенні невірного пароля(рис.9) програма "думала", що пароль вірний(«вірний пароль(12345)») і надавала доступ для подальшої роботи, треба замінити 1 строчку асемблерного (машинного) коду програми(перехід jne(якщо не еквівалент)), яка виконується після команда порівняння “cmp” (рис.10), на перехід який переходить на ту строчку код де викликається команда,якщо результат був успішний, тобто введений пароль співпав з "еталонним"(рис.11).



Рисунок 9 ­– введення невірного пароля

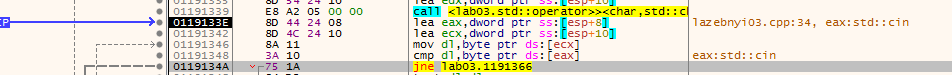


Рисунок 10 – перехід jne(якщо не еквівалент)

На рис.11 продемонстрована заміна переходу



Рисунок 11 – заміна переходу

На рис.12 продемонстровано результат роботи програми після заміни асемблерного (машинного) коду програми таким чином, щоб при введенні невірного пароля програма "думала", що пароль вірний і надавала доступ для подальшої роботи.

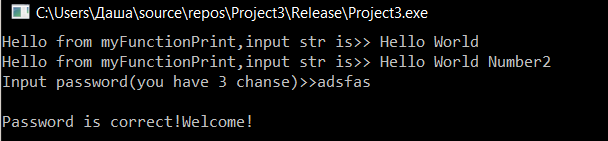


Рисунок 12 – результат роботи програми після заміни переходу

**Висновки:**

В результаті виконання лабораторної роботи було з’ясовано, що основна причина вразливості – використання деяких функцій мови С, які працюють з рядками і не перевіряючих розміри своїх аргументів (наприклад strcpy, strcat, gets або sprintf). Тому переважна більшість (якщо не всі) уразливих програм написані на С.