LAB

3

Developing Puzzle Program Using C in Kali

By the end of this section of the practical, you should be able to:

- Writing a C program in Kali
- Compile a C source code in Kali

3.1 Introduction

To start the journey of malware analysis, first we need to know a bit of reverse engineering skill. The skill to know how a program works from it binary form. To start doing a reverse engineering on a software, we need to have a program or binary file to be dissected. This lab session is a refresher of how to write and compile a C base program. The output of this lab session is a puzzle program, where a user need to input the correct number/characters to obtain a flag.

3.2 The program flowchart

The flowchart of the puzzle program is illustrated in Figure 1.

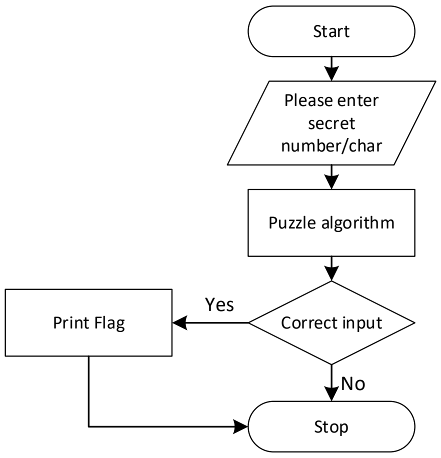

Figure 1: Puzzle flowchart

The program start by asking the user to key in a valid number/characters. Then the program send the input data to the puzzle validation algorithm in which will determined whether the keyed in data is the right number/characters.

Once the puzzle algorithm determined the validity of the keyed in data, the puzzle program will display the flag and if it is not the right number/characters the puzzle program will print “Wrong”. The interface of the program is displayed in Figure 2.

```
Please enter secret number : X

Congratulation you guest the correct number !!!!!!

Your flag is : flag_XXXXXXX_XXXXX
```

Figure2: Puzzle Interface

Task 1

Write the program in C

1. For the main program follow the source code below:

```
#include<stdio.h>
#include<stdbool.h>
#include<stdlib.h>

bool puzzle(int x);
int flag();

int main(){
   int x;
   printf("This is a puzzle program\n");
   printf("Please enter secret number : ");
   scanf("%d",&x);

   if(puzzle(x)){
```

```
            printf("Congratulation you guest the correct number !!!!!\n");
            flag();
            }
    else
            printf("Please try again !!!!!\n");
}
bool puzzle(int x){
```

Design your own puzzle algorithm here, the function must return a bool (TRUE/FALSE).
A puzzle algorithm can simply be like testing input to a formula and return true when the result of the formula is equal to certain number.
For example

```
Return (((x*4)/2)==10);
```

OR

```
int total;
total=(x*8)/4;
if(total==10)
return 1;
else
return 0;
```

```
}

int flag(){
    char c;
    char      *data[22]={"01100110",      "01101100",      "01100001",
"01100111",  "01011111",  "01100010",  "01101001",  "01110100",
"01110011",  "00110011",  "00110100",  "00110101",  "00110011",
"01011111",  "01111001",  "01101111",  "01110101",  "01100111",
"01101111", "01110100", "01101101", "01100101",};

            printf("\n your flag is =");
            for(int a=0; a<22; a=a+1){
                      c = strtol(data[a], 0, 2);
                      printf("%c", c);
                      }
            printf("\n");
            return(0);
}
```

2. Save the source code as puzzle{groupname}.c and compile the code.

3.3 Compile C in Kali

To compile a C source code in kali you need a gcc tool which by default is installed in kali 2020. However you need to compile the C source code into a 32 bit program(if you are using a 64 bit machine)

In order to compile it in a 32biot program your kali 2020 requires another tool which you can install by issuing the command below:-

```
> sudo apt-get install gcc-multilib
```

Once install the source code can be compile using the following command

```
>gcc  [sourcecode.c] -o puzzle -m32 -fno-stack-protector -no-pie
```

To run the program just type in

```
>./puzzle
```

Do this lab in a group of 4 members, each group must develop a very strong puzzle. We will use the puzzle program in doing reverse engineering during our lecture 5.

Task 2

write the program in C++ and compile the source code as windows base PE. (hint: use mingw-64)