
The background of the slide is a dark blue, abstract image. It features a perspective view of a tunnel or a path that recedes into the distance. The walls of the tunnel are composed of glowing binary code (0s and 1s) in various shades of blue and green. Light streaks and bokeh effects are visible, suggesting motion and depth. The overall aesthetic is futuristic and technological.

CWE Challenge - Array Sort

Michael Mendoza

2023-01-30

Contents

Introduction	2
Initialization	2
Struct	2
Function Declarations	2
File Pointer	2
Local Variables	3
Reading in the Data	3
Sort and XOR	3
Flag Creation	4
XOR	4
Compiling the Program	4
Flag	4
Conclusion	5
References	5

Introduction

This challenge provides a binary file that contains an array that needs to be read in, sorted, and then xor'd.

Initialization

Struct

The information from the README.md file specifies the data the struct will contain.

```
1 //create a struct for each element in the array
2 typedef struct element
3 {
4     uint16_t value;
5     uint8_t flagPiece[13];
6 }element;
```

Function Declarations

The following function declaration is what is expected for the qsort(). I modified this to fit the struct that was created.

```
1 //function declaration
2 int cmpfunc (element *a, element *b);
```

File Pointer

Inside the main function, the code starts with creating a file pointer to start reading in the data.

```
1 int main()
2 {
3     //create the file pointer to read in the data
4     FILE *fp = fopen("./input_stream.bin", "rb");
5     if(fp == NULL)
6     {
7         exit(1);
8     }
```

Local Variables

The rest of the variables initialized are as follows: the “numberOfBytes” variable was initialized to the number of bytes in the file, the “numberOfElements” was obtained by dividing the total amount of bytes by 15 (2 bytes for the value, 13 bytes for the flag piece), and the “arrayOfElements” which will be the array of elements that the data will be written to.

```
1  /* move the file pointer to the end of the file,
2     * save the total bytes of the file and set the
3     * file pointer back to the beginning of the file
4     */
5  fseek(fp, 0, SEEK_END);
6  long numberOfBytes = ftell(fp);
7  rewind(fp);
8
9  //save the number of elements
10 int numberOfElements = numberOfBytes / 15;
11
12 //allocate memory for and create the array of elements
13 element *arrayOfElements = (element *)malloc(numberOfBytes);
14 if(arrayOfElements == NULL)
15 {
16     exit(1);
17 }
```

Reading in the Data

A for loop is used to read in the data from the binary file.

```
1  //read in the data to the array
2  for(int i=0; i < numberOfElements; i++)
3  {
4      fread(&arrayOfElements[i].value, sizeof(uint16_t), 1, fp);
5      fread(&arrayOfElements[i].flagPiece, sizeof(uint8_t), 13, fp);
6  }
7  fclose(fp);
8  fp = NULL;
```

Sort and XOR

The following function was used to sort the array

```
1  //sort the elements using qsort()
2  qsort(arrayOfElements, numberOfElements, sizeof(element), cmpfunc);
    //cmpfunc is changed to accommodate the struct
```

```
3 }
```

Flag Creation

Creating the flag using calloc will initialize the flag with 0's.

```
1 //allocate memory for and initialize the flag with 0's
2 char *flag = calloc(13, sizeof(char));
3 if(flag == NULL)
4 {
5     exit(1);
6 }
```

XOR

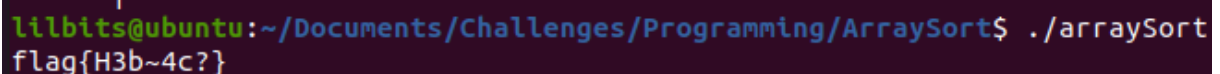
The following for loop will iterate over each element of the “arrayOfElements” and xor each byte of the flag pieces (13 bytes) with the flag.

```
1 //perform xoring with even indexes of "ArrayOfElements"
2 for (int i = 0; i < numberOfElements; i+=2)
3 {
4     for (int j = 0; j < 13; j++)
5     {
6         flag[j] ^= arrayOfElements[i].flagPiece[j];
7     }
8 }
```

Compiling the Program

Flag

After compiling the program and making sure it works we get our flag!



```
l1lbits@ubuntu:~/Documents/Challenges/Programming/ArraySort$ ./arraySort
flag{H3b~4c?}
```

Figure 1: Flag

Conclusion

Learning how to work with arrays as pointers and understanding how to do bitwise operations was the experience I needed from this challenge to move on to the next programming challenges.

References

1. https://www.tutorialspoint.com/c_standard_library/c_function_rewind.htm
2. https://www.tutorialspoint.com/c_standard_library/c_function_qsort.htm#:~:text=The%20C%20library%20func
3. <https://www.educative.io/answers/what-is-the-qsort-function-in-c>
4. <https://www.javatpoint.com/c-program-to-sort-the-elements-of-an-array-in-ascending-order>