MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL TECHNICAL UNIVERSITY

"KHARKIV POLYTECHNICAL INSTITUTE"

Department of Computer Engineering and Programming

«Software Means of Information Protection »

*Laboratory work report No 4*

*Topic: «* **Code injection. Adding new section** *»*

Student:

Group. KH 919 I.E

Ayoub EL-HADDADI

Verified by:

Lecturer Viktor CHELAK

Kharkiv – 2022

***Purpose of work***:

To acquire practical skills for correcting errors in software that is in operation;

Creation of new functionality using implicit in use software for the x64 platform in the masm64 environment.

.

***Individual task:***

In accordance with the student number in the group, select the task option and write a program in the assembly language in masm64. The numbers a є {a1, a2, a3, a4} are given by an array.

Variant 8:

**Algorithm of the program**

First step of the program is to check the ability of the microprocessor to support the AVX and AVX2 commands. Therefore we use **cupid** command to check bit 28 (for **AVX)** of **ecx** register if it’s 1 then they are supported.

Then, we move to part of working with the equation. We’ve initialized an array of “a” (array A = {a1, a2, a3, a4} with 4 random values, and constants “b, c, d and e”, we move each constant to a register “xmm”, and calculate

Next step is we move into a loop, and calculate result of equation for every “a” value in array A, for such reasons we use AVX command, and we are going to need:

VSQRTSD - Compute Square Root of Scalar Double-Precision Floating-Point Value

VMULSD - Multiply Scalar Double-Precision Floating-Point Value

VCVTTSD2SI - Convert with Truncation Scalar Double-Precision Floating-Point Value to Signed Integer

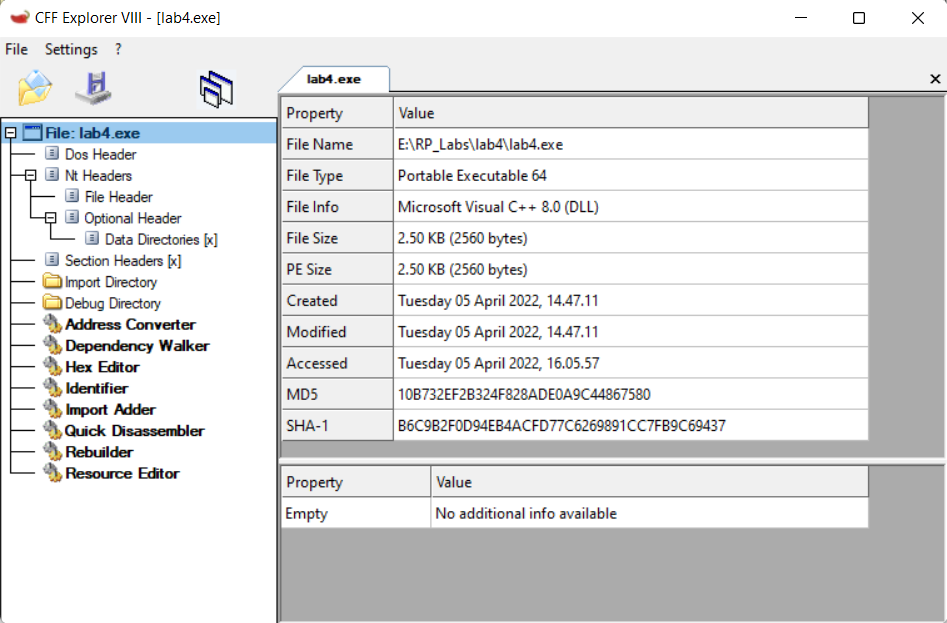
VDIVSD - Divide Scalar Double-Precision Floating-Point Value

VADDSD - Add Scalar Double-Precision Floating-Point Values

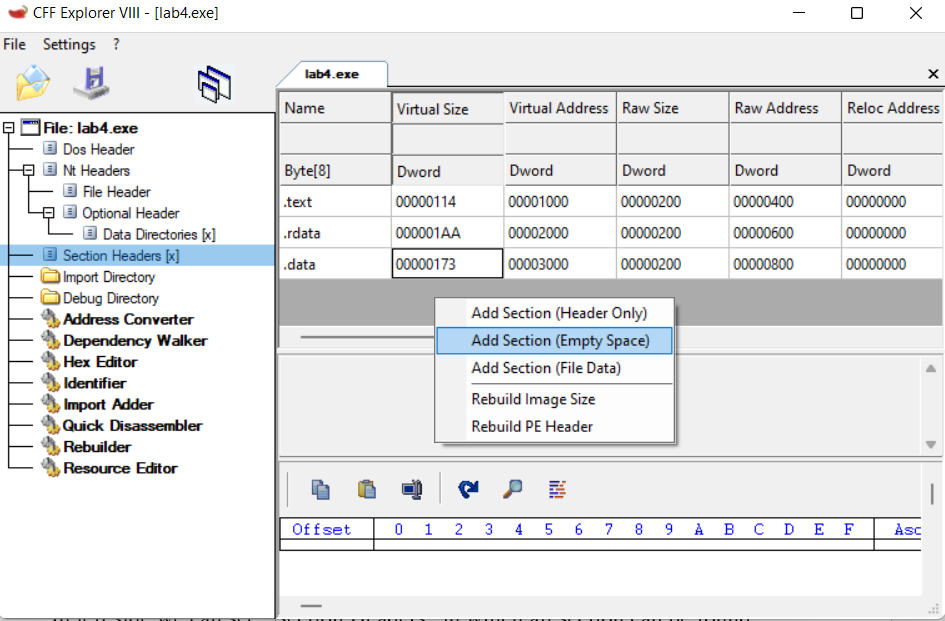
We fill an empty array with results and output result in form of message box

Injection in an executable file

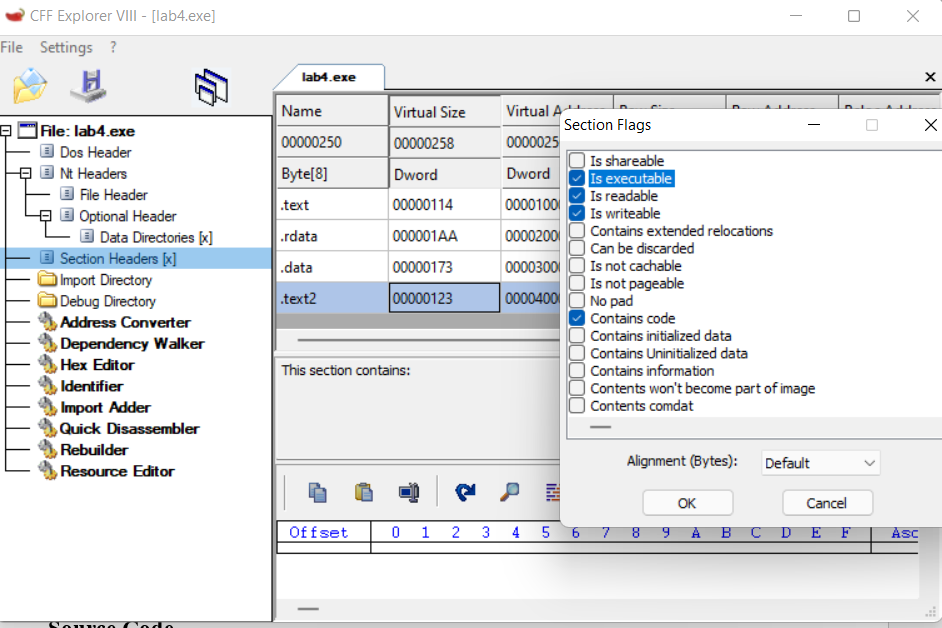
For this part we must add a new section of code into which the implementation of the Message Box message output function with the output of the “Infected. Name, group number. ". For adding new section part we use CFF\_Explorer program



In left side we can see “Section Headers” in which all section can be found and created

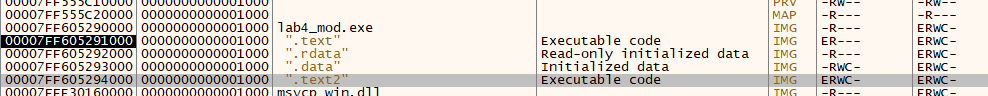


Right click in empty space and “Add section (Empty space)”, and input size of the new section, now at this step new section has been created and can be named as desired. And we must also add some flags to make is executable like the following:

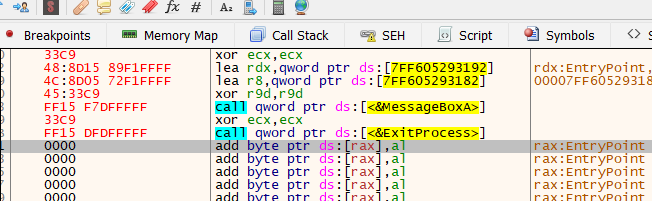


Next, we click save or save as in file menu, now, a new section has been created and ready for injection new code with x64dbg.

After opening modified executable file in x64dbg, in **memory map** section e can see the new section created “.text2”



Double click on it and it opens in CPU section, now we are ready to inject new code



In .text section we simply add a jump instruction to first address of this section, and final step is to patch file.

**Source Code**

Full source code of this lab you can find it in:

[**https://github.com/Elh-Ayoub/RP\_Labs/blob/main/lab4/lab4.asm**](https://github.com/Elh-Ayoub/RP_Labs/blob/main/lab4/lab4.asm)

**Results of the program:**



Figure 1 – Check support of AVX

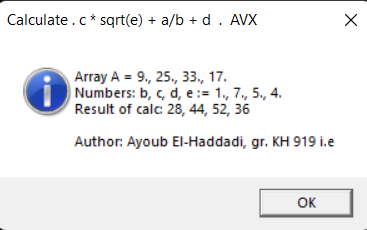


Figure 2 – Program results

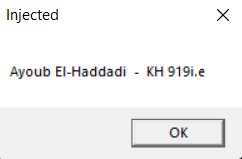


Figure 3 – patched exe file after injecting

**Conclusions:**

As a result of laboratory work we gained a practical skills in correcting errors in software that is in operation and creation of new functionality using implicit in use software for the x64 platform in the masm64 environment.

**You can also find this report in:**

<https://github.com/Elh-Ayoub/RP_Labs/tree/main/Docs>