Assignment 2

The objective of this assignment is to gain understanding in identifying similar code and shared code structures between multiple Binaries.

Task 1-2 Due Date: February 1, 2017 11:59 PM (1−2) Person Teams

Task 3-4 Due Date: February 8, 2017 11:59 PM (1-2) Person Teams

Preparation:

Write a toy program, fairly small C++ program that performs some task. (100−200 lines of code should be sufficient, but probably less than 500 would be good). This program should contain at least one header file and one source file to include into the main program.

Example Blackjack game C++ program, includes Card.h and Card.cpp
It can be even simpler, but try to have at least 25% of the code in the 2 files.

Task 1: Calling Convention Generation

This task should be fairly straight forward, and guarantee you understand the process of what exactly a calling convention is. This may take some google−fu if you were not in the lecture or did not understand what was happening. You can post to piazza if you have any questions about this step.

With your toy program compile the program with 3 calling conventions. These are up to you to choose. You should be left with 3 executables from your 1 source program. If your platform does not support a package like visual studio, the 2nd floor labs have the software.

With the same header and source file compile into another executable using one of the 3 calling conventions you use previously. (If I used CDecl, StdCall, and FastCall, I could chose CDecl for this new program.)

You should end this task with 4 executables with 3 calling conventions. You are to turn these in as well.

Task 2:

Raw Data Extraction Task 2 is to find the 5 longest common substrings between:

Each pair of files (1,2), (1,3), (1,4), (2,3), (2,4), (3,4) Each triplet of files (1,2,3), (1,2,4), (1,3,4), (2,3,4) The one quadruplet (1,2,3,4)

In the report you should detail how you extract the binary to calculate the longest common strings as well as how your string matching algorithm works.

Task 3: Disassembly Problem

Given these 55 strings, some will contain data, some will contain strings, and some will contain padding. Practically this is a “Disassembly Problem” as you do not know if the LCS_i for each collection of input files starts on an instruction or in the middle of an instruction, or is not an instruction at all.

Since these binary strings will be extremely small (usually 100 bytes) you will most likely not be able to use a recursive disassembly. The suggested way would be linear disassembly which is what the online disassembler does “https://www.onlinedisassembler.com/odaweb/” .

You are to classify which of the input binary streams are instructions and which are data. (It is a hard problem)

If possible create a model for code and data to make this step automated and describe your model.

Task 4: Tool Creation

With your scheme in place find a tool that allows plugins such as OllyDBG, IDAPro, Immunity Debugger. There is literally hundreds of tools to choose from.

With this tool create a plugin to incorporate your technique and highlight the common code. (Data as well assuming non−trivial data (Also a Hard Problem)).

Highlighting will be left ambiguous for you to define, but think of it as trying your best to gain an edge over other software reverse engineers.

Use screenshots and a separate report to detail how it works, how well it works, and what process you went through to implement it. What resources did you use to learn how to implement as well as why you chose that tool as your target platform.

Tools :

IDA, Ollydbg
online assembler https://defuse.ca/online-x86-assembler.htm#disassembly and disassembler http://www2.onlinedisassembler.com/odaweb/
Coding Language of your choice

Report:

The report is the most important part of this project, it is how you show us what you have tried and what you were able to get working correctly. Since this is what you are graded on you should spend your time ensuring it is correct and accurately displays what went into this assignment.

As you complete this assignment you should also include a slide set to show the process of your assignment. For the first 2 tasks you should have 2 slides. The last week you can edit the previous slides and include one more. The best slides will be chosen and you will present your work to the class.

Be sure your report includes the following points.

1. Introduction to your projects ideas, and your thinking process behind each step
2. A brief description of your source code, and what each file your turning in is
3. A paragraph for each concept above and what the main topic of each was focused on
4. Implementation and demonstration using screen captures
5. A description of each part of your tool−chain and reasoning behind why you chose that tool
6. A CLEAR description of what each member of your team did with work distribution