

# Number Guessing

## Author

Alec Howard

## Time Taken

~ 3 Hours

## Overview

The purpose of this project was to show my ability to properly navigate IDA in order to reverse all functionalities of a disassembled C++ program.

## Objectives

- Disassemble & Reverse the game, naming every method & defining all structures/classes used by the main method.
- Write a script that leverages knowledge of how the game actually works to win it.
- Identify & decrypt the winning message.
- Identify & use any hidden features in the game

## Terminology

- Disassemble: To convert machine language code into human-readable assembly code.
- Structure: A way to group several related variables into one place.

## Key Technologies/Tools Used

- IDA Educational 8.3
- Visual Studio Code
- [Cyber Chef](#)




## Challenges

My biggest challenge was labeling the structures when reversing the program. This is a relatively new process for me, so it will take some time to adjust.

## Results/Findings

My completely labeled project can be seen in the included i64 file. The script I wrote to win the game leveraged the finding that the game uses the system time as the seed for the random number generator. I replicated this in python when writing my version, and luckily there were no discrepancies of system time between the executable and my python program. The only secret functionality that I found was the ability to run the program with a debug flag. This allows for the user to see both the system time and random number used for each iteration of the program. Since my solution is in the form of a command line interface, I implemented this functionality as well as an optional flag. This script allowed me to win the game and obtain the secret message, however, in order to glean as much practice as I could out of this project, I also reversed the secret message by hand. I found that it was simply being encrypted using xor, with a key of the decimal value of 50. I plugged the values of the array into cyber chef, and used the proper filters to

decrypt the message:

[Download CyberChef](#)  Last build: 11 days ago - Version 10 is here! [Read about the new features here](#) [Options](#)  [About / Support](#) 

Operations

xor

XOR

XOR Brute Force

XKCD Random Number


Hex to Object Identifier

Unicode Text Format

Text Encoding Brute Force

Lorenz

Magic

Favourites 

Data format

Encryption / Encoding

Public Key

Recipe

From Decimal

Delimiter  
Space

☐ Support signed values



XOR

Key  
50

DECIMAL

Scheme  
Standard

☐ Null preserving

STEP  BAKE!  Auto Bake

Input

101 93 69 19 18 107 93 71 18 83 64 87 18 83 92 18 87 94 91 70 87 18 85 71 87 65 65  
87 64 19

raw 91 1 Raw Bytes LF

Output

Wow! You are an elite guesser!

raw 30 1 2ms Raw Bytes LF

## Conclusion

This project was a simple, yet super fun introduction to reverse engineering programs using IDA. It showed me how fun and rewarding reversing can be, and excites me to learn more about the subject.

## References

<https://docs.python.org/3/library/subprocess.html>