Software Cracking and Keygenning

OVERVIEW

This project was designed to expand your knowledge of fundamental reverse engineering practices: software cracking and keygenning. These tasks are basic; yet, they use many fundamental reverse engineering techniques that are important for the reverse engineer to master.

ASSIGNMENT

This assignment is comprised of two parts. The first involves patching code to bypass security. Introducing vulnerabilities (or rather, bypasses) is a basic software cracking practice. The second involves reversing code and replicating its output. This is an essential technique for keygenning.

Part 1: Cracking and Patching using OllyDBG

Minesweeper is frustrating at best. You're winning; and better yet, your time is fast. You have two tiles left and there's no way to determine which one has the last bomb hidden beneath it. Your time is still counting. You choose, but you chose wrong. You're angry, but you know you're smarter than a computer program. Your task is to cripple the Minesweeper program by patching the random number assembly code using OllyDbg. Make the placement of bombs predictable for every new game. Target the seed of the pseudorandom number generator (PRNG). Alter the code so that Minesweeper always seeds the PRNG with the same value. Make sure to keep a clean copy of winmine.exe before you patch the machine code. Answer the following questions:

1. **Describe (precisely and accurately) how winmine.exe seeds its pseudorandom number generator.**

2. **How does your patch subvert the PRNG?**

3. **Show your patch (e.g., Offset, Original Code, Patched Code)**

4. **Print and attach screenshots from multiple executions to show that the randomness of bomb placement has been defeated. Screenshots must be distinguishably different executions (do this by clicking on a different bomb).**

Part 2: Keygenning using IDA

Reverse the machine code for keygenme.exe using IDA. Program a keygenner in Python, Java, Ruby (Metasploit plugin language of choice), or C/C++ (please ask an instructor for permission to use any others). **No assembly code is permitted, inline or otherwise! Your keygen must randomly generate valid keys.**

1. **Generate a valid key:**
2. **Generate another valid key:**
3. **Print your keygen code and staple it to this assignment sheet.**