PV204 Security technologies

Project presentation, Reverse engineering of binaries

Petr Švenda *svenda@fi.muni.cz* *@rngsec*
Centre for Research on Cryptography and Security, Masaryk University

CROCS
Centre for Research on Cryptography and Security

Laboratory

1. Go to disassembly in IDE, understand basic principles
2. Practical disassembling and binary debugging tutorial
   - Lena tutorial 1 & 2
   - Open file in debugger
   - Basic operations, jumps
   - Patching
   - Understanding structures from assembler

MIXED MODE IN IDE

Pair activity: mixed mode in IDE

- Form pairs
- Investigate together REMixedModeDemo (next slide)
- Explain to other person items from next slide

- Spring 2020
  - Make online call with one other colleague, discuss and explain to her/him

Mixed mode in IDE

- Use project REMixedModeDemo (from PV204_RE_2020.zip)
- Debugging session must be running (otherwise menu option is not available)
- Explanation of different parts from source code wrt assembler
  - Assignment, addition, if, loop, switch, while
  - Local copy of variable inside switch
    - prevention of race condition issues
- Difference between Debug and Release
  - Removal of dead code
  - Removal of compile-time decidable conditions
  - Optimizations via reserved register (esi)
  - Faster instruction XOR esi, esi
- Difference between Visual Studio and QTCreator (or your IDE)

Individual activity

BINARY DEBUGGING

Lena tutorials

- Tutorial 1: basics + binary patching
- Tutorial 2: reversing of algorithm

- IMPORTANT: Newer browsers may prevent Lena tutorial to run (Flash player required) – enable for use on this page only or use something older (e.g., IE)

OllyDbg - shortcuts

- **F3** ... Open binary file
- **F2** ... Toggle breakpoint (on opcodes, or double click)
- **F9** ... Run debugged program
- **Ctrl+F2** ... Restart program, all temporary changes are lost!
- **F8** ... Step over
- **F7** ... Step into
- **Spacebar** or double click ... allows to set new opcode. Use when you like to change program behaviour, e.g., replacing conditional jump (JGE) by unconditional jump (JMP) or to discard instruction (NOP).
- **Alt+BkSp** ... Undo change

OllyDbg - shortcuts

- ***Rightclick*->Search for->All referenced text strings** ... Constant text strings referenced in code. Use to find strings like hardcoded passwords, important messages (“Wrong license”). Double click on string will takes you to referencing instruction. Helps you to build mind model quickly.
- ***Rightclick*->Find references to->Address constant** ... will find references to particular memory elsewhere in the code – use when you like to know where in code the memory is set, changed or otherwise used.
- **Ctrl+F1** ... Help on Win32 API (WIN32 API help file already prepared in OllyDbg directory (WIN32.HLP)). Use to get meaning of the parameters pushed to stack just before the API function is called.
- **;** ... add or edit your comment for specific code line. Use to write down things you already understand. Use classic paper as well (program mind model)
- ***Rightclick*->Copy to executable->All modifications (**or **Selection**) … make changes permanent. New window with modified code is opened. ***Rightclick*->Save file** to write patched binary to disk.

ASSIGNMENT 06 – CRACKME DISASSEMBLING

Assignment 06 – Crackme disassembling

- Reverse engineer supplied crackme file pv204.exe (from PV204_RE_2020.zip)
  - Obtain information about its behavior (OllyDbg)
  - Make crackme to continue successfully without error message by
    a) Patching (modification of control routine, submit patched file)
    b) Creating valid license info (submit license file)
- Produce short (1xA4) text description of solution
  - How you performed analysis, what you learned, how you solved
- Bonus: More principally different solutions for the same problem might be awarded by extra points
- Submit before: 16.4. 23:59 (full number of points)
  - Every additional started day (24h) means 1.5 points penalization