Exam 2 Extra Credit

sdh140430 Swapnil Hasabe

Motivation of learning aspects of CTF:

1. CTF introduces the real tools and real types of problems without breaking the law. It provides opportunity to learn O.S, applications, cryptography, architecture, attack-defense techniques.

2. It comes into following categories:

Reverse Engineering	Cryptography	Exploitation
Programming	Debugging	Shell Coding

3. Following are the important elements of Reverse Engineering:

3.1. Difference between Static and Dynamic

Points	Static	Dynamic
Description	Looking at the code, figure things out	Examine the process during execution
Analysis	Disassemblers are used for static analysis e.g. IDA Pro, objdump	Debuggers are used for dynamic analysis. E.g. WinDBG, IDA, GDB

3.2. Tools used to analyze following two file format:
PE (Portable Executable): CFF Explorer, IDA, pefile (python library)
ELF (Executable and Linkable Format): readelf, objdump, file

3.3. Assembly (More focused on Intel Instructions): This section demonstrates how the instructions like mov, cmp, jmp, call works. It also describes the data types and sizes in assembly. It provides process memory layout for windows, Linux and virtual memory.

3.4.1 Registers: This section describes the all types of registers e.g. EIP, ESP, EBP, EAX, EBX, ECX, EDX, ESI and EDI
3.4.1 Three stack frames Prolog, Epilog and Epilog 2 have been explained in this section.

3.5. IDA
3.5.1: We can do many things, including grouping a set of nodes, color coding them, and renaming them
3.5.2: Knowing that all these checks error out on failure we can simplify the graph

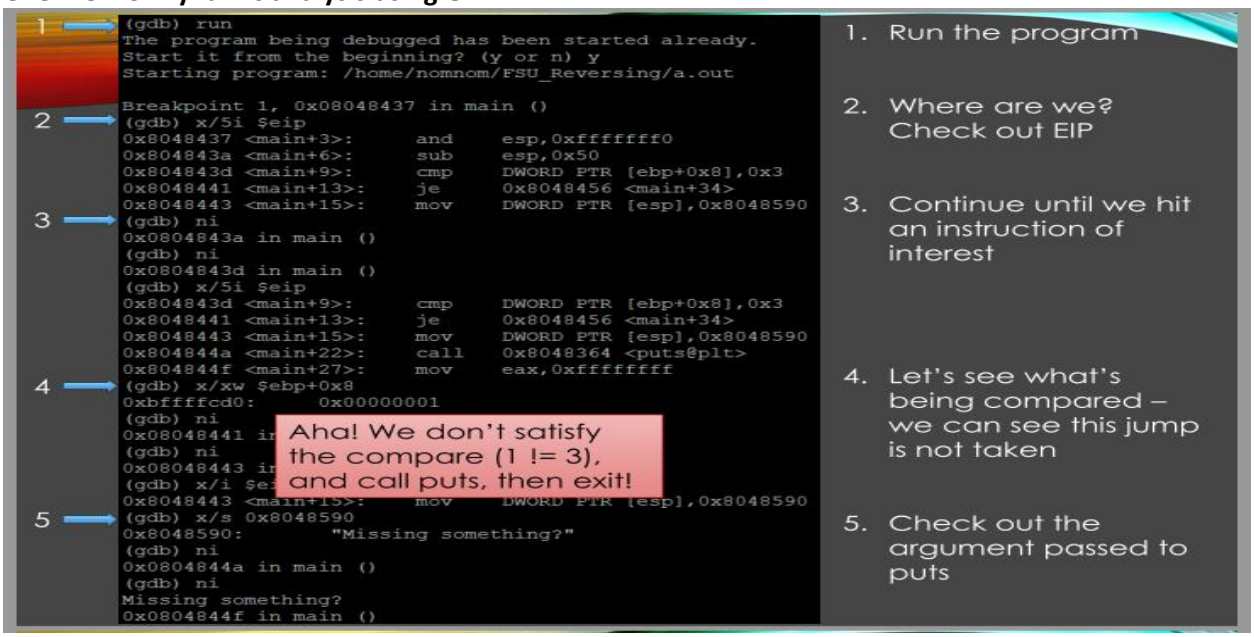
3.6 Debugging (Dynamic Analysis through Debugging):
A good debugger will have several useful features
- Set breakpoints
- Step into / over
- Show loaded modules, SEH chain, etc.
- Memory searching

Dynamic Analysis: You cannot control everything
- If you want to skip over an instruction, or a function call, do it!
- If you want to bypass the “authentication” method or make it return true… you can!
- You can change register contents and memory values, whatever you want.
- You can even patch programs (make changes and save it to a new executable).

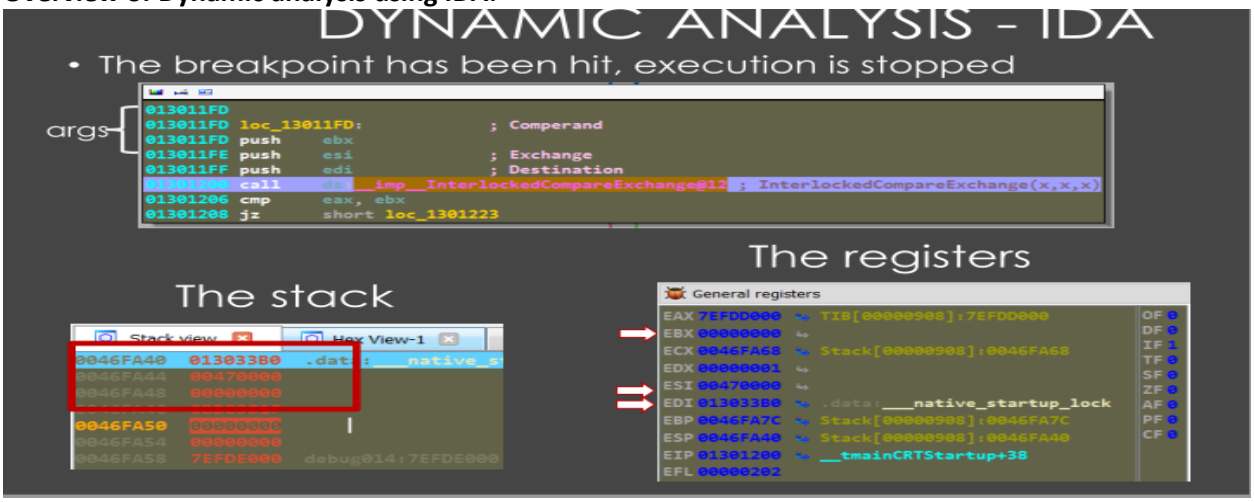
Exam 2 Extra Credit

sdh140430 Swapnil Hasabe

Overview of Dynamic analysis using GDB:



Overview of Dynamic analysis using IDA:



Everyone has their own preferences
But combination of the IDA and GDB will undoubtedly yield the best results.

Conclusion:
Following learning goals achieved:

- **Opening up and examining a binary and looking at its sections to get a feel for it**
- **Working with and simplifying the disassembly**
- **Converting back to source code where needed**
- **Using a debugger to fill in the gaps or manipulate program execution**