Reverse Engineering

Steven Spinner
Michael Fleming
Regan Tam
Blas Kojusner


Googling stuff about binary analysis
Reverse Engineering

Reverse Engineering

- Reverse engineering is analyzing a compiled file to understand how it works

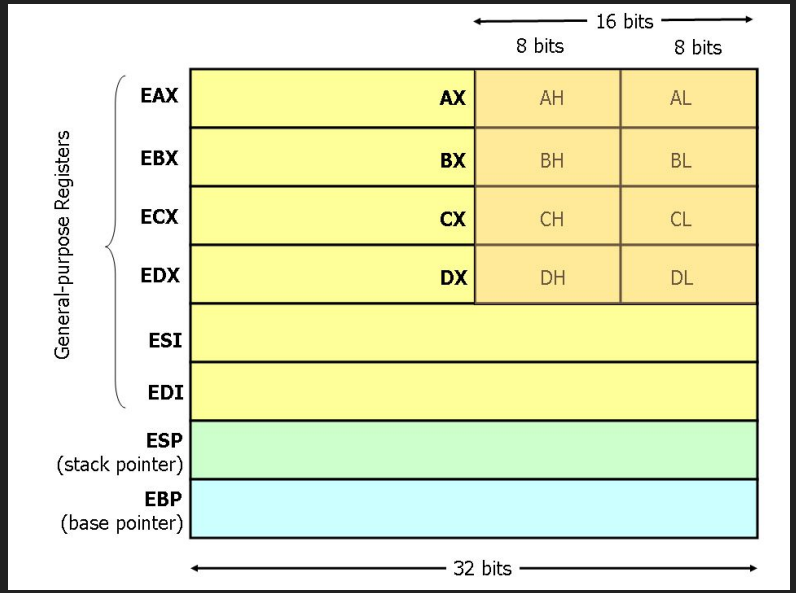
- Reverse engineering can be broken down into two types of analysis
  - Static Analysis: Examining the instructions in an executable file without running the program
  - Dynamic Analysis: Examining the instructions in an executable file through running the program

Intro to x86

- x86 is a Complex Instruction Set Computer (CISC)
  - This means one instruction can perform several operations
  - This is in contrast to Reduced Instruction Set Computer (RISC) where each instruction only does one operation
- Has hundreds of instructions
  - Many of them are not used in most programs
- 32 bit instruction set
- x86_64: the 64 bit extension to x86
  - Is very similar to the 32 bit versions, with register sizes increased to 64 bits, more registers, and new calling conventions

x86 Registers

- EAX: Accumulator
- EBX: Base
- ECX: Count
- EDX: Data
- ESI: Source
- EDI: Destination
- ESP: Stack pointer
- EBP: Base pointer
- EIP: Instruction pointer / Program counter
- EFLAGS: Flags register



Important x86 Instructions

- MOV dest, src
  - Moves the data in the source register to the destination register
  - Can load data from memory into a register by doing MOV dest, [src]
  - Can store data into memory from a register by doing MOV [dest], src
  - The [] operator tells the CPU to access the memory at the address stored in the register.
- PUSH src
  - Pushes the src register onto the stack and decrements ESP.
- POP dest
  - Pops the data at the top of the stat to the destination register and increments ESP.
- RET
  - Pops an address from the top of the stack into EIP.
- CALL dest
  - Pushes EIP onto the stack and jumps to the destination.

More x86 Instructions

- CMP reg1, reg2
  - Can be used to see if two values equal, less than, or greater than each other by setting flags in EFLAGS.
- TEST reg1, reg2
  - Does a logical and on reg1 and reg2 and sets EFLAGS based on the output.
  - Can be used to see if a register contains zero by using the same register for both operands
- JMP src
  - Moves the location in source to EIP.
- Conditional jumps
  - Take the form of J__ offset
  - Uses the EFLAGS register to see if a condition is met for a jump.
  - Examples: JE, JNE, JNZ, JG, JG, JL

Even More x86 Instructions

- Math and logic instructions
  - Most take the form of ___ dest, src
  - Most of them performs the operation between the source and destination and stores the result in destination
  - Examples: ADD, SUB, AND, OR, XOR

- INC src
  - Adds 1 to the value in source
- DEC src
  - Subtracts 1 from the value in source

Tools

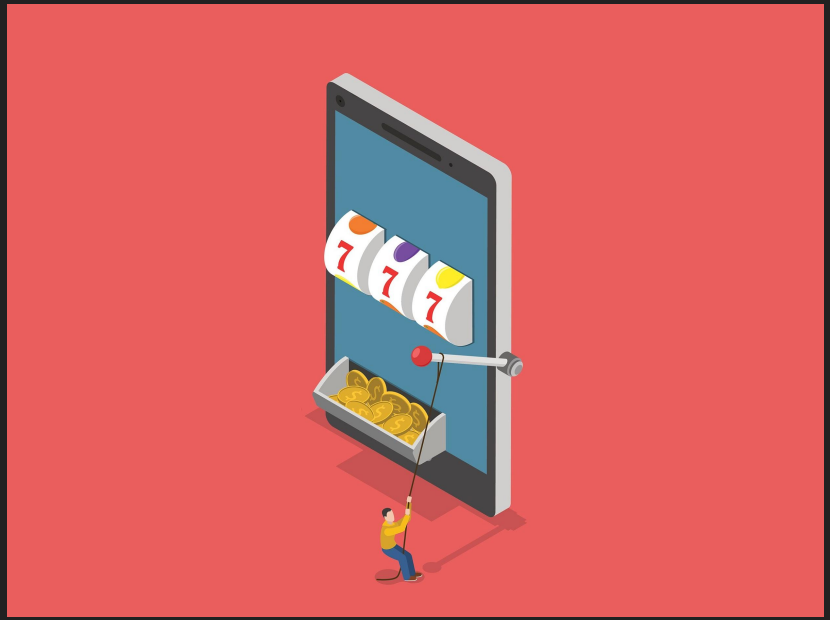
- Disassembler - Interactive Disassembler (IDA). Used for generating assembly language source code from machine-executable code.
- Debugger - GDB can be used to better understand how an executable operates.
- Packet sniffer/ bus analyzer - can be used to get internal data transmissions as well as inbound and outbound communication packets.
- API monitor - Helpful for viewing parameters passed into functions.

More RE

- Vulnerability:
  - Any code that can be understood by a computer can be RE’d.
  - Mitigations and prevention
    - Making common RE tools work against the engineer.
      - Dummy calls
      - Debugger detection
      - Trampolines
    - No way to prevent it.
    - Make the code open source.

Reverse Engineering Slot Machine Algorithms

- For the past decade, Alex, a Russian hacker, milked millions off of slot machines around the world.
- After reverse engineering a slot machine, he learned how to predict the outcomes of the slot machine and developed his own algorithm to exploit other machines.

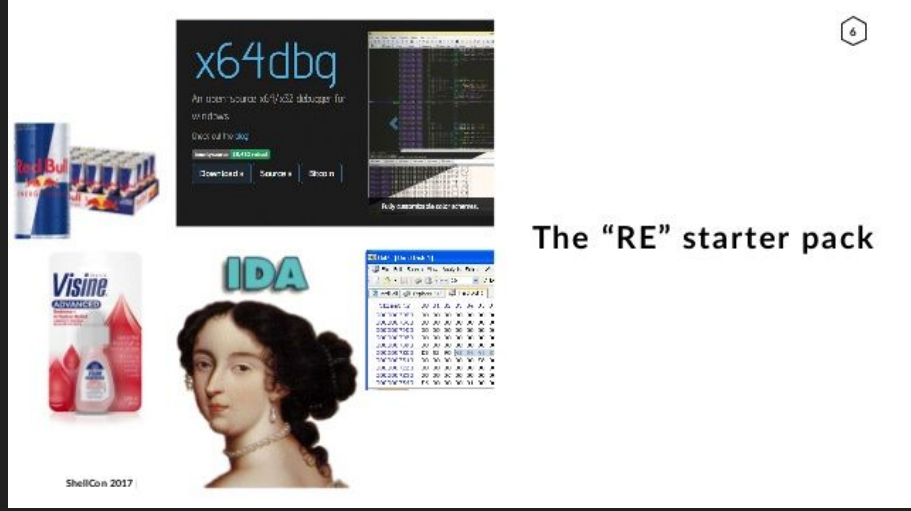


Pseudorandom Number Generators

- In 2008, Alex, was hired by a casino to “tweak” their slot machines to favor the house more than usual.
- He reverse engineered the software and learned of it’s pseudorandom number generator (PRNG) algorithm, and found a way to predict its future outcomes.
- He then hired agents all over the world to use an iPhone app that uses his algorithm to milk money from different casinos. The app figures out the machine’s PRNG parameters and tells them when to press spin.

Technical Demo

- Password Crackme
- Tools used:
  - Vagrant
  - Ghidra
  - GDB (Peda)
- Demo available at http://www.github.com/bkojusner



Resources

- https://www.felixcloutier.com/x86
- http://flint.cs.yale.edu/cs421/papers/x86-asm/asm.html
- https://www.wired.com/story/meet-alex-the-russian-casino-hacker-who-makes-millions-targeting-slot-machines/
- https://casino.guru/how-to-beat-slots