Reverse Engineering

with angr and z3

Play with zeros and ones

RAKESH KUMAR

Content

1. $Whoami
2. What is reverse engineering?
3. Process of compilation.
4. Disassemble, Decompile and Debug.
5. Static and Dynamic Analysis.
6. Tools.
7. Angr.
8. Z3.
9. **Live Demo**

<< WHOAMI >>

Rakesh Kumar (0xrakesh)

Computer science student

CTF Player

TamilCTF Member

Interested in reverse engineering and cryptography

Social Media

http://0xrakesh.github.io

0xrakesh @GITHUB

0xcyberhackz @TWITTER

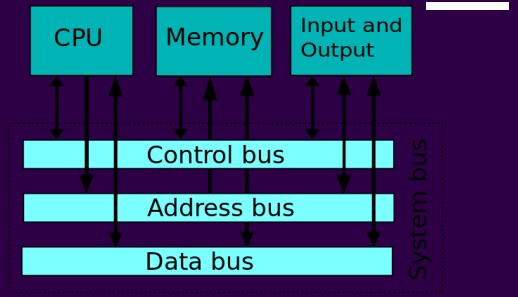
What is reverse engineering

Reverse engineering is the process of analyzing the program's function and structure.

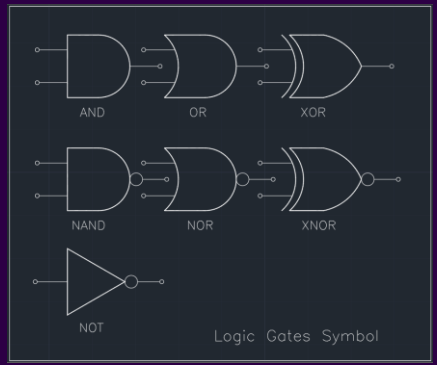
Reverse engineering play a major role in malware analysis.

How CPU work

A **CPU executes an instruction by fetching it from memory, using its ALU to perform an operation**, and then storing the result to memory.



The actual mathematical operation for each instruction is performed by a combinational logic circuit within the CPU's processor known as the arithmetic–logic unit or ALU



CPU Registers

General Purpose Register : **EAX, EBX, ECX, EDX, ESP, EBP, EDI, ESI**

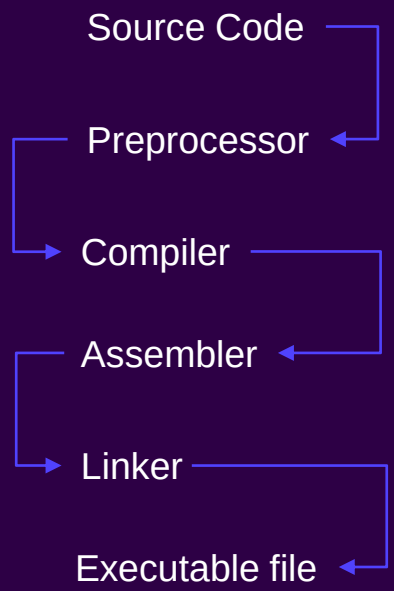
Special Purpose Register: **EIP, Program Counter**

Assembly Instruction

```
mov eax, 4
mov ecx, 2
add eax, ecx
sub eax, ecx
```

A = 4
B = 2
A+B
A-B

Compilation Process:



Decompiler

Decompiling is the process of translating the object code or compiled code to source code.

(Low-Level Code -> High Level Code)

Disassembler

Disassemble is the process of translating the object code or compiled code to assembly code.

(Low-Level Code -> Assembly Code)

Debug

- Debugging is the process of finding and removing errors from the program.
- In simple words, tracing the program.

Reverse Engineering

Reverse engineering is the process of analyzing the program.
There are two types of analysis:

1. Static Analysis.
2. Dynamic Analysis.

Static Analysis: Analyze the program without actually executing it.
In other words analyze the program's structure.

Dynamic Analysis: Analyze the program's behavior by executing it.

Basic tools

- file
- strings
- ltrace
- Objdump
- Gdb
- Ghidra
- IDA
- Binary Ninja
- WinDbg
- DNSpy

Demo

ANGR

- Angr is a python framework for analysis binary.
- Angr is a multi-architecture binary analysis toolset.
- Analyzes both static and dynamic binary.

USES

- ❑ Disassembly.
- ❑ Symbolic execution.
- ❑ Control-flow analysis.
- ❑ Decompilation.

Let’s analysis the binary with angr

Angr

angr.Project(‘filename’)	->	Open the binary
project.arch	->	Display the arch of the binary
project.entry	->	Display the entry address

Loader

Loader is represent the binary in virtual address space

project.loader	->	Display the entire memory address of binary (libc)
project.loader.main_object	->	Display the program function memory address.
project.loader.shared_objects	->	Display the memory address of shared objects
project.loader.find_symbol(‘func’)	->	Display the memory address of the function.
proj.loader.main_object.execstack	->	Check the stack is executable.
proj.loader.main_object.pic	->	Check if the binary is position independent
proj.loader.main_object.plt[‘func’]	->	Display the address of the func in plt.

Factory

project.factory.block(‘main’).pp()	->	Disassemble the function
state = project.factory.entry_state()	->	Simulation Program State.
state.regs.rip	->	Address of instruction pointer.
state.regs.rax	->	Address of rax register.

Solve the challenge with angr

Z3 Framework

Z3 is a high-performance theorem prover developed at Microsoft Research.

Z3 is used in many applications like constraint solving, analysis of hybrid systems.

Z3 is based on SMT (Satisfiability modulo theories)

Structure of Z3 program

```
from z3 import *
s = Solver()
x = Bool('x')      # Init the variable
s.add(x == 4)      # Adding Constraint
s.check()          # Check the constraint is satify or not
s.model()          # Final output
```

Solve the challenge using z3

Resources

- crackmes.one
- ctfsites.github.io
- challenges.re
- Tryhackme : Reversing ELF, Reverse Engineering, Windows RevEng, Brainstorm.
- HackTheBox
- RESources

Thank you