





# Introduction



radare2

Reverse engineering framework

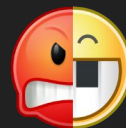
<https://github.com/radareorg/radare2>



modality

Integration

<https://github.com/0xchase/Modality>



angr

Symbolic execution tool

<https://github.com/angr/angr>



# Symbolic Execution Refresher

---

Determine what input values are required to reach certain program states

Interpreter follows the program, providing symbolic values for program input, and computing constraints for these symbolic parameters

In angr, the program is lifted to **VEX IR**, and the program is interpreted by the **simulation manager**, using what are called **bitvectors** as symbolic values, adding constraints which can eventually be passed to **z3**

Each path in the execution has an associated **state**, and at any point in the **exploration** each state is in a stash, normally the **active**, **deadended**, **found**, **unconstrained** stashes.

```

;-- main:
[0x00400133] sym.main.133
; var int local_118h @ rbp-0x118
; var int local_110h @ rbp-0x110
; var int local_104h @ rbp-0x104
; var int local_100h @ rbp-0x100
; var int local_1h @ rbp-0x1
; DATA xREF from 0x00400000 (entry0)
0x0040095a 55 push rbp
0x0040095b 4889e5 mov rbp, rsp
0x0040095e 4881ec200100 sub rsp, 0x120
0x00400965 89bdfcfe1111 mov dword [rbp - local_104h], edi
0x0040096b 4889b5f0fe1111 mov qword [rbp - local_110h], rsi
0x00400972 488995e8fe1111 mov qword [rbp - local_118h], rdx
0x00400979 b800000000 mov eax, 0
0x0040097e c863111111 call sym.hanner
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x00400
0x004009b0 4889c7111111 mov rdi, rcx
0x004009b9 e86111111111 call sym.checkPassword
0x004009be 84c0 test al, al
--< 0x004009c0 740c je 0x4009ce
0x004009c2 bf2e0b4000 mov edi, str.Password_accepted_1 @ 0x400b1a
0x004009c7 e8c4fd111111 call sym.imp.puts
==< 0x004009cc eb0a jmp 0x4009d8
; JMP xREF from 0x004009c0 (sym.main)
--> 0x004009ce bf410b4000 mov edi, str.Wrong_1 @ 0x400b41
0x004009d3 e8b8fd111111 call sym.imp.puts
; JMP xREF from 0x004009cc (sym.main)
--> 0x004009d8 b800000000 mov eax, 0
0x004009dd c9 leave
0x004009de c3 ret
[0x0040095a]>

```

# Reversing



```

-- main:
[0x00400133] sym.main.133
; var int local_118h @ rbp-0x118
; var int local_110h @ rbp-0x110
; var int local_104h @ rbp-0x104
; var int local_100h @ rbp-0x100
; var int local_1h @ rbp-0x1
; DATA xREF from 0x00400000 (entry0)
0x0040095a 55 push rbp
0x0040095b 4889e5 mov rbp, rsp
0x0040095e 4881ec200100 sub rsp, 0x120
0x00400965 89bdfcfeffff mov dword [rbp - local_104h], edi
0x0040096b 4889b5f0feffff mov qword [rbp - local_110h], rsi
0x00400972 488995e8feffff mov qword [rbp - local_118h], rdx
0x00400979 b800000000 mov eax, 0
0x0040097c c863f0ffff call sym.hanner
0x00400980
0x00400984
0x00400988
0x0040098c
0x00400990
0x00400994
0x00400998
0x0040099c
0x004009a0
0x004009a4
0x004009a8
0x004009ac
0x004009b0 4889c7ffff mov rdi, rcx
0x004009b9 e861f0ffff call sym.checkPassword
0x004009be 84c0 test al, al
--< 0x004009c0 740c je 0x4009ce
0x004009c2 bf2e0b4000 mov edi, str.Password_accepted_0 ; Password_accepted_0 @ 0x400b1a
0x004009c7 e8c4fdffff call sym.imp.puts
==< 0x004009cc eb0a jmp 0x4009d8
; JMP xREF from 0x004009c0 (sym.main)
--> 0x004009ce bf410b4000 mov edi, str.Wrong_0 ; Wrong_0 @ 0x400b41
0x004009d3 e8b8fdffff call sym.imp.puts
; JMP xREF from 0x004009cc (sym.main)
--> 0x004009d8 b800000000 mov eax, 0
0x004009dd c9 leave
0x004009de c3 ret
[0x0040095a]>

```

# Exploitation





Exploration

Constrained

Symbolic?

Constrain

Payload

Explore all paths using current active states. Make stdin symbolic.

Check if any states are unconstrained (more than 256 possible pc values)

Check if program counter is symbolic and can be constrained

Constrain the program counter and all stdin bytes to make a base payload

Build a payload, bypassing protections, etc from the base payload







# Review

Showed various state initialization techniques

# PC overwrite vulnerability discovery

## Basic exploit generation

## A few ASLR/DEP bypasses



# Looking Forward

---

Integrate analyses

Integrate symbion

Track heap usage

More ASLR/DEP bypass techniques

Bug fixes

And more

