A journey through *ESIL*

Understanding code emulation within radare2

Arnau Gàmez i Montolio | *@arnaugamez*

September 6, 2019
r2con - Barcelona

Who am I

- Student - *Maths* & *CS* @ UB
- President - *@HackingLliure*
- Collaborator - *#r2con*

Motivation

- Force myself to **understand basics** of emulation and ESIL (what, why & how)
- Provide an **easy intro to ESIL** for people wanting to understand/get into it
- Show simple use cases

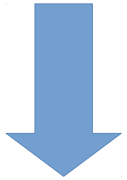
Outline

What is emulation?

- Simulate the execution of code of the **same or different CPU**

What is emulation?

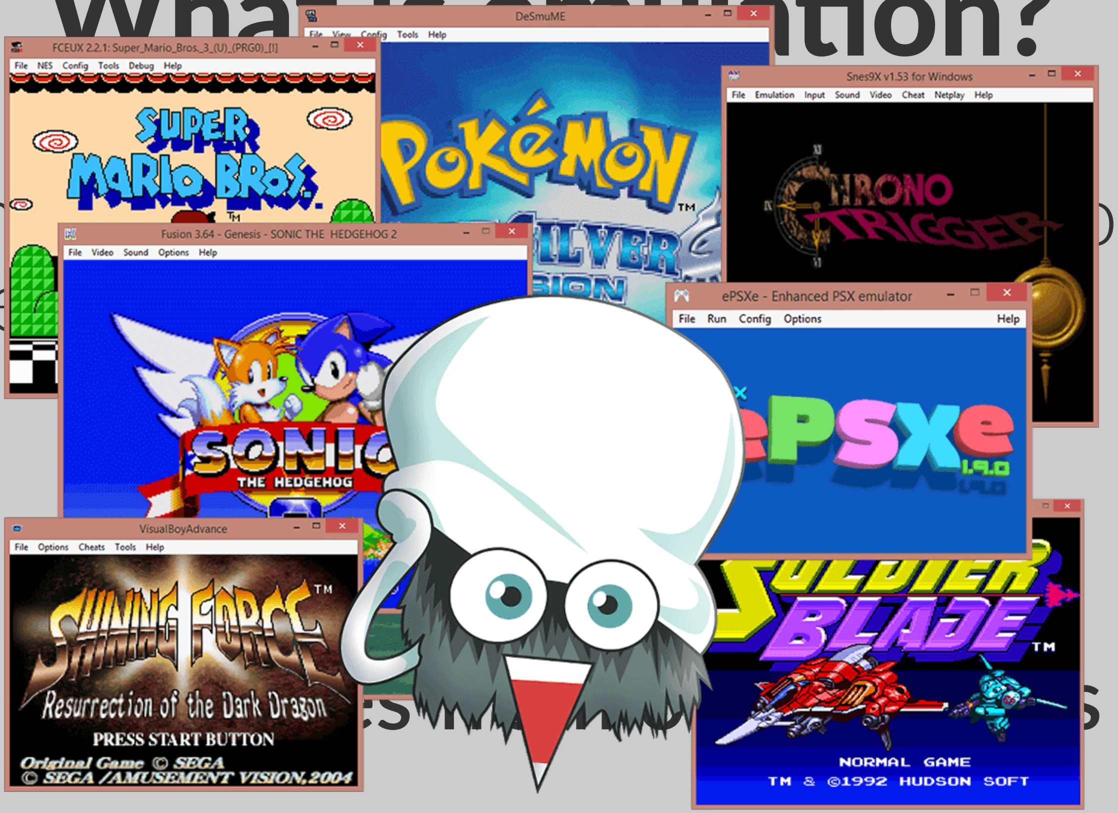
- Simulate the execution of code of the **same or different CPU**



Run **games from old consoles**

What is emulation?

- Si… of the…


FCEUX 2.2.1: Super_Mario_Bros._3_(U)_(PRG0)_[!]
File NES Config Tools Debug Help
SUPER MARIO BROS.
DeSmuME
File View Config Tools Help
POKéMON SILVER
Snes9X v1.53 for Windows
File Emulation Input Sound Video Cheat Netplay Help
CHRONO TRIGGER
Fusion 3.64 - Genesis - SONIC THE HEDGEHOG 2
File Video Sound Options Help
SONIC THE HEDGEHOG
ePSXe - Enhanced PSX emulator
File Run Config Options Help
ePSXe 1.9.0
VisualBoyAdvance
File Options Cheats Tools Help
SHINING FORCE
Resurrection of the Dark Dragon
PRESS START BUTTON
Original Game © SEGA
© SEGA /AMUSEMENT VISION,2004
SOLDIER BLADE
NORMAL GAME
TM & ©1992 HUDSON SOFT

Why emulation?

- **Understand** specific snippet of code
- **Avoid risks** of native code execution
- Help **debugging** and **code analysis**
- Explore **non-native executables**

Outline

1 Emulation

2 **Intermediate languages & ESIL**

3 ESIL operation

4 Demos

Intermediate languages

"Language of an ***abstract machine*** *designed to aid in the analysis of computer programs" -- wikipedia*

↓

Vital for (de)compilation

What is ESIL?

- **E**valuable **S**trings **I**ntermediate **L**anguage
- Small set of instructions
- Based on reverse polish notation (stack)
- Designed with **emulation and evaluation in mind**, not human-friendly reading

What is ESIL?

- Infinite memory and set of registers
- Native register aliases
- Ability to implement **custom ops** and call external functions

Why ESIL?

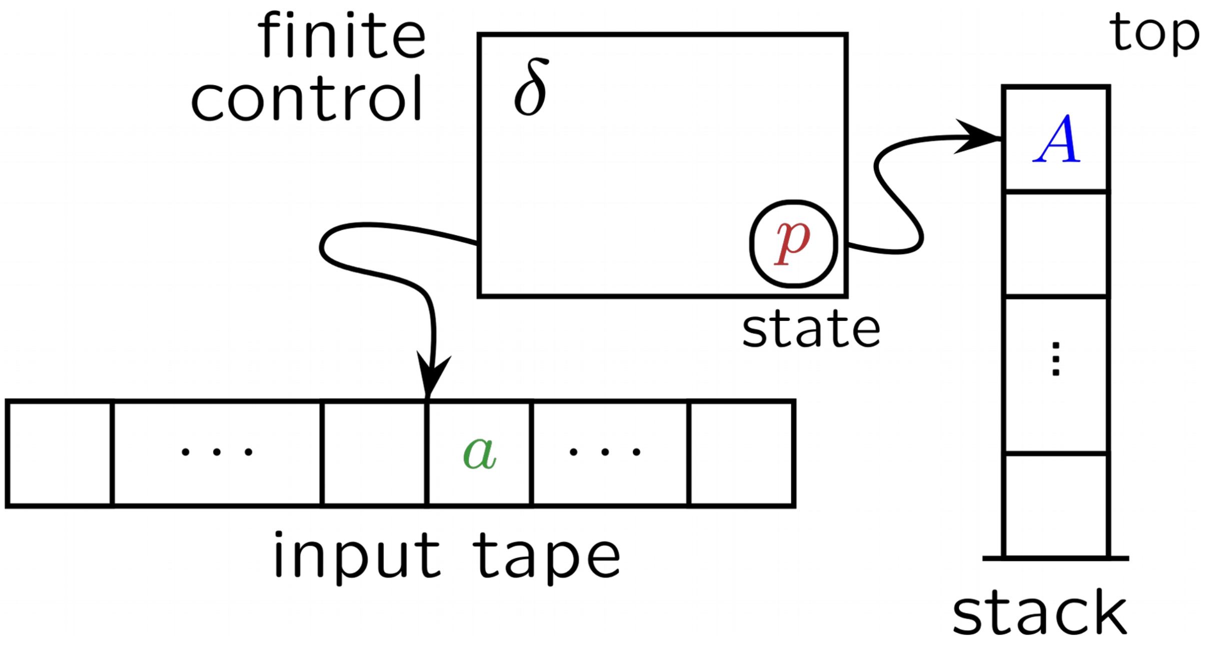
- Need for emulation on r2land
- Easy to generate, parse and modify
- Extensibility
- Why not?

Outline

ESIL

Stack machine on steroids

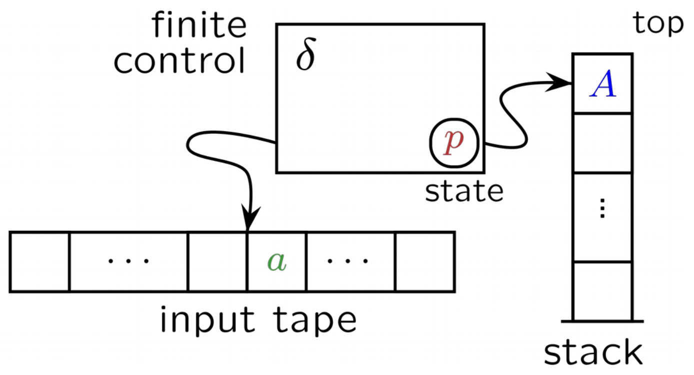
Stack machines / PDA's



Stack machines / PDA's

- input symbol
- current state
- stack symbol

- state transition
- manipulate stack (push/pop)



Parser idea

```
while not at end of esil_string {
   cur = get_next_element()
   if cur is esil_operation {
      op = get_esil_operation(cur)
      op ()
   } else {
      push (cur)
   }
}
```

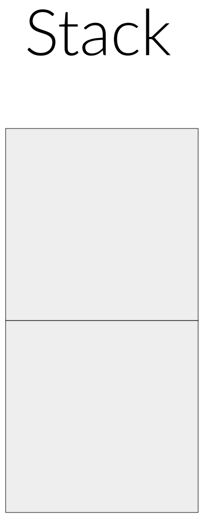
Parser idea

```
while not at end of esil_string {
    cur = get_next_element()
    if cur is esil_operation {
        op = get_esil_operation(cur)
        op ()
    } else {
        push (cur)
    }
}
```

Will pop and use previously pushed symbols as operands

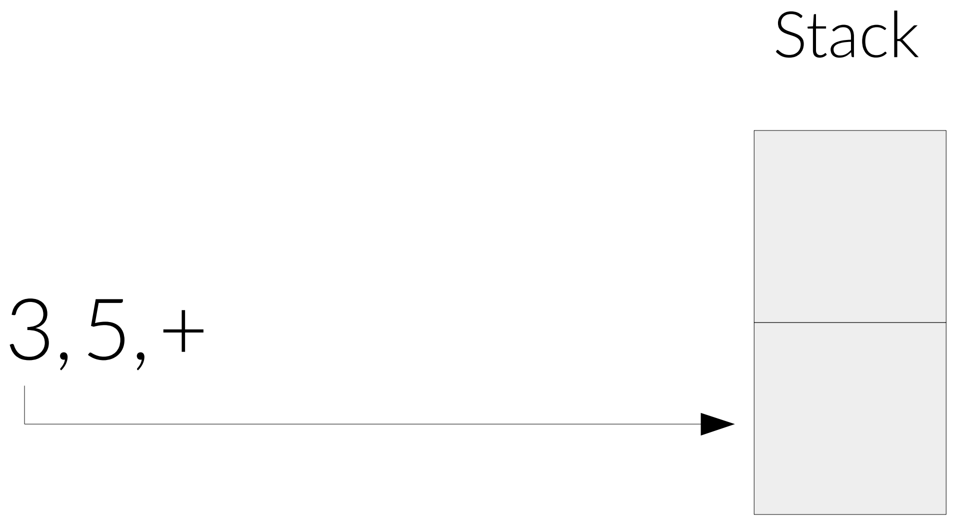
Visual animation

$3, 5, +$



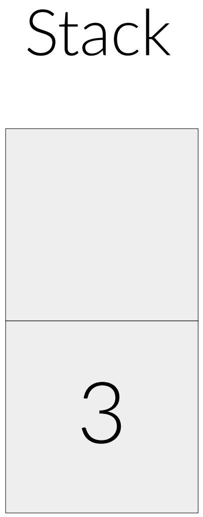
Not end of ESIL string → "3" symbol not an operation → So push to stack

Visual animation



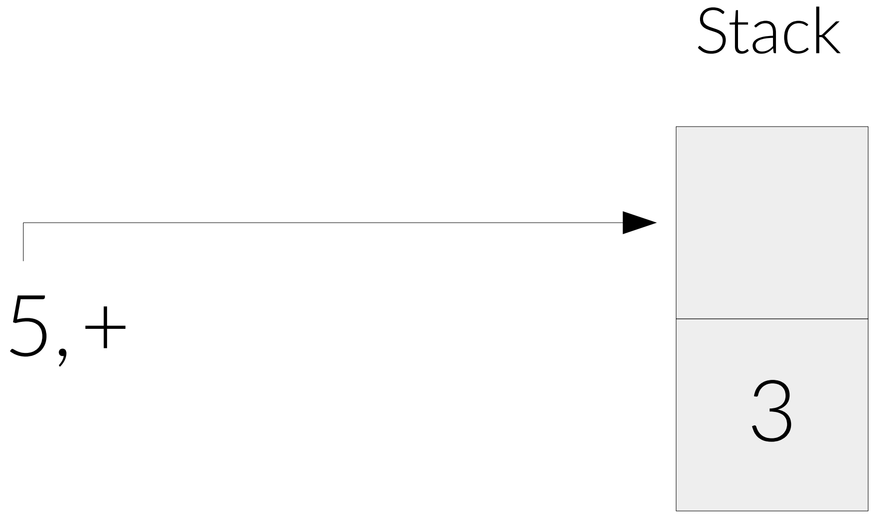
Visual animation

5, +

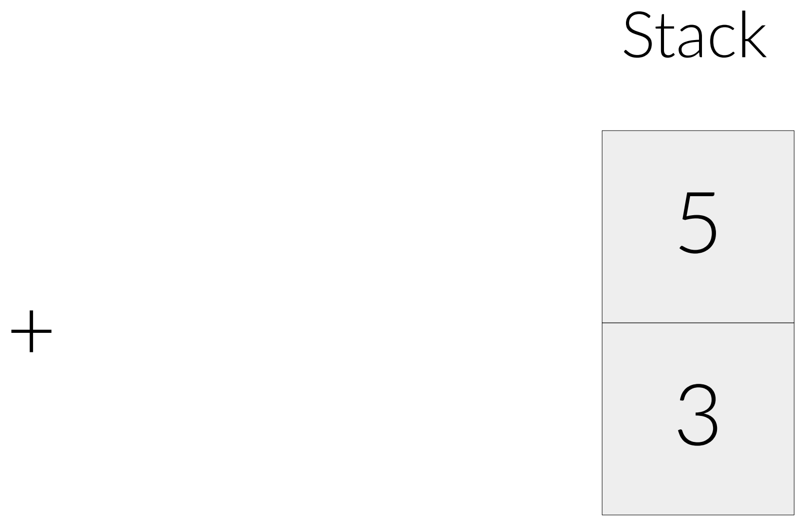


Not end of ESIL string → “5” symbol not an operation → So push to stack

Visual animation

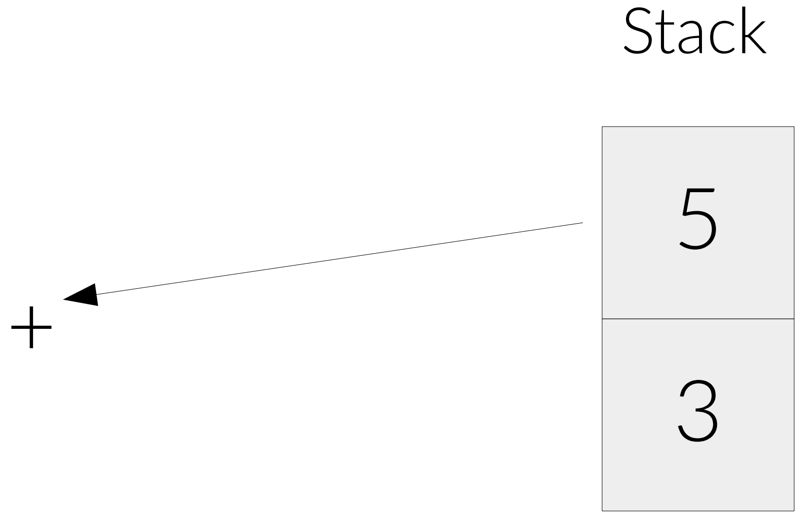


Visual animation



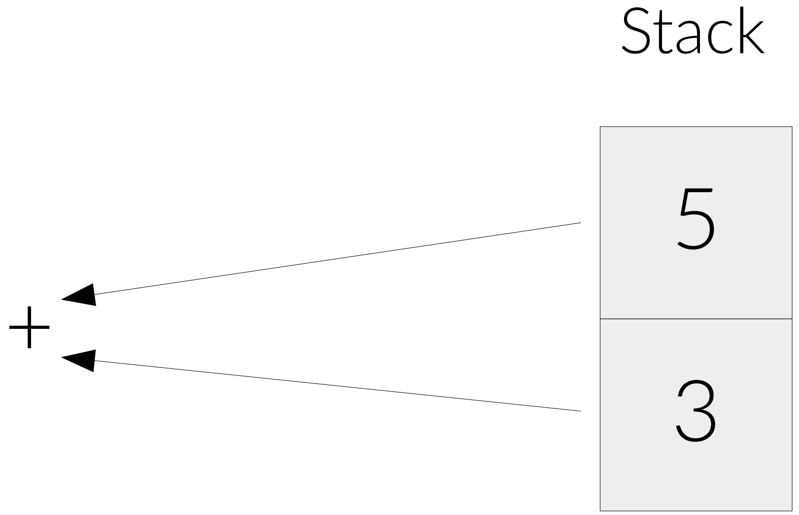
Not end of ESIL string → “+” symbol is an operation → So…

Visual animation



Pop values from stack

Visual animation



Pop values from stack → Use them as operands

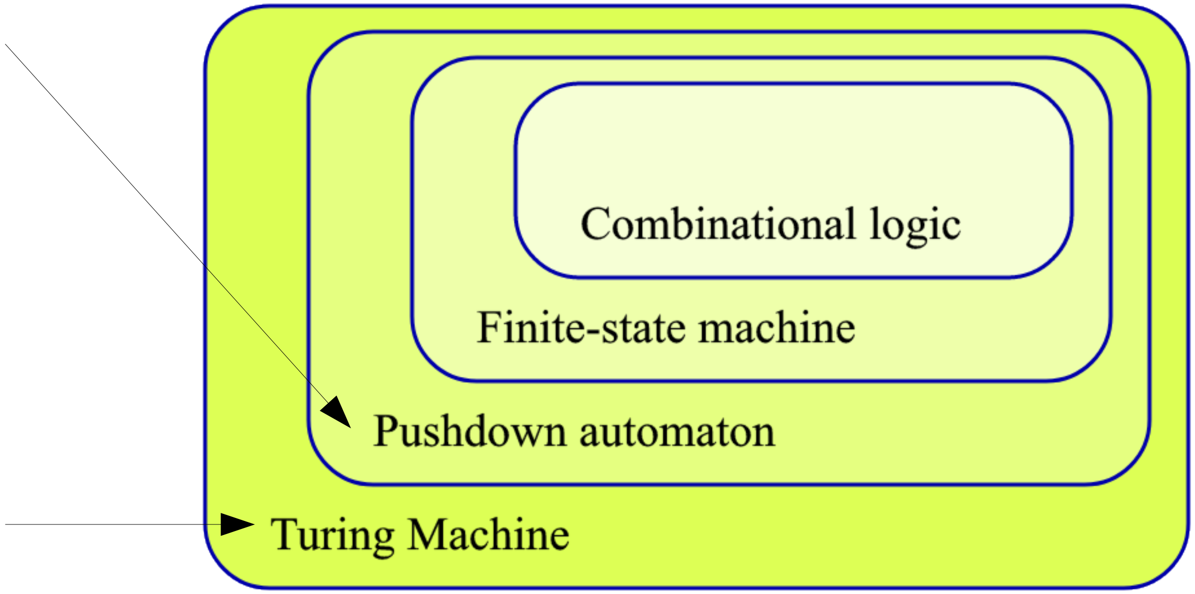
Visual animation

Pop values from stack → Use them as operands → Perform operation

Expanding stack machines

We are here

We want to be here



cc @condr3t

HOW?

HOW?

STEROIDS

(aka cheating)

Steroids x1

- Add **random access** operations
- Add **control flow** operations

Steroids x2

- **Register** access
- Add "**extra tape**" with random access (virtual memory, VM stack)

x	y	z	a	b	c	...

Basic practical usage

ESIL options are under **ae** (**a**nalysis **e**sil) subcommands

ae***i***	***i***nit	ae***su***	***s***tep ***u***ntil
ae***im***	***i***nit ***m***emory	ae***so***	***s***tep ***o***ver
ae***ip***	***i***nst. ***p***ointer	ae***ss***	***s***tep ***s***kip
ae***s***	***s***tep	ae***r***	***r***egisters

ESIL operands

Check ***ae??*** on a radare2 shell

(description and examples)

ESIL internal vars (flags)

Prefixed with ***$*** | read-only

- $z – zero flag
- $cx – carry flag from bit x
- …

Updated on each operation. Used to set flags for particular arch.

Outline

Demo

Defeat simple crackme

cc @pof @jvoisin

Demo

Deobfuscate encrypted code

cc @superponible

Thanks

- Pancake (*@trufae*)
- Xvilka (*@akochkov*)
- Condret (*@condr3t*)
- Skuater (*@sanguinawer*)

A journey through *ESIL*

Understanding code emulation within radare2

Arnau Gàmez i Montolio | *@arnaugamez*

September 6, 2019
r2con - Barcelona