Introduction to radare2

sym.hack // June 30th 2018
mandlebro

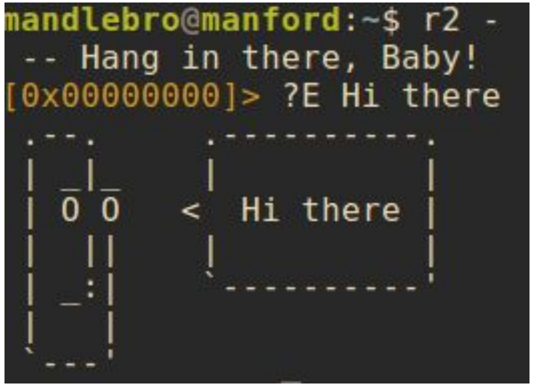
$whoami

- @fcasal on Github
- @filipe_casal on Twitter
- maths/compsci background
- GSOC student with r2

What is radare2?

Reverse Engineering framework

- Forensics
- Assembler/disassembler
- Reversing
- Exploitation
- Debugger
- Emulation



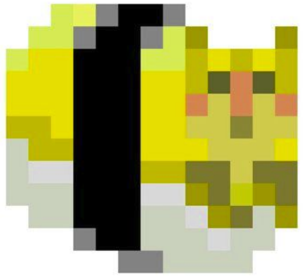
```
mandlebro@manford:~$ r2 -
 -- Hang in there, Baby!
[0x00000000]> ?E Hi there
 .--.     .----------.
 | _|_    |          |
 | O O   <  Hi there |
 |  ||    |          |
 | _:|    `----------'
 |   |
 `---'
```

Why radare2?

- FOSS
- Very active development 18,519 commits
- Runs everywhere
- Scriptable
- Supports multiple architectures
- Handles many file formats

History

- Created by pancake in ~2006



- Started as a forensic analysis tool do dump and recover data from disks
- Quickly grew and was rewritten from scratch
- Large community 513 contributors
- r2con since 2016

Installing r2

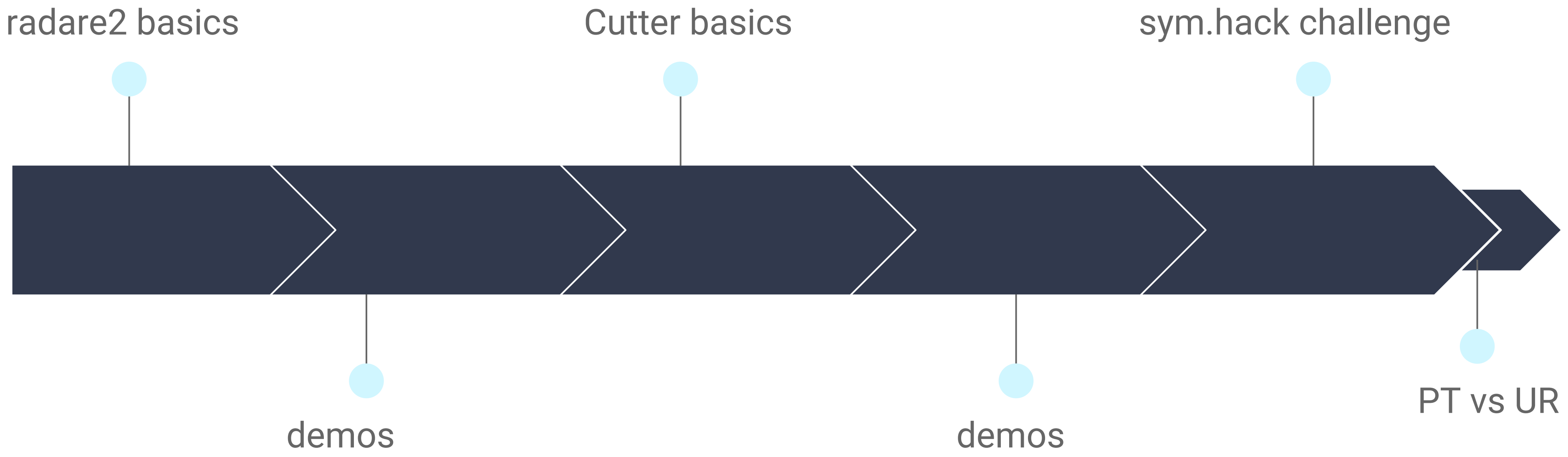
Always install from git:

```
$ git clone git@github.com:radare/radare2.git

$ git clone --depth=1 git@github.com:radare/radare2.git

$ cd radare2/

$ sys/install.sh

        or

$ sys/user.sh
```

Python bindings r2pipe

```
$ pip install r2pipe --user
```

Overview of today



radare2 tool family

radare2	r2pm	rarun2	ragg2
rabin2	radiff2	rax2	rahash2
rasm2	rafind2	r2agent	

Basic r2 commands

Spawning r2

```
$ r2 -            # opens r2 with malloc://1024
$ r2 --           # opens r2 without any file
$ r2 /bin/ls      # opens ls in r2
$ r2 -d /bin/ls   # opens in debugger mode
```

Help command: ?

```
> ?               # most useful r2 command
```

Internal grep: ~

```
> ?*~write        # 2nd most useful r2 command
```

Basic r2 commands

Seeking: s

```
> s 0xcafebabe          # seek to addr
```

Printing: p

```
> p8 [len]   # print sequence of bytes
```

Writing: w

```
> w foo      # write foo at current seek
```

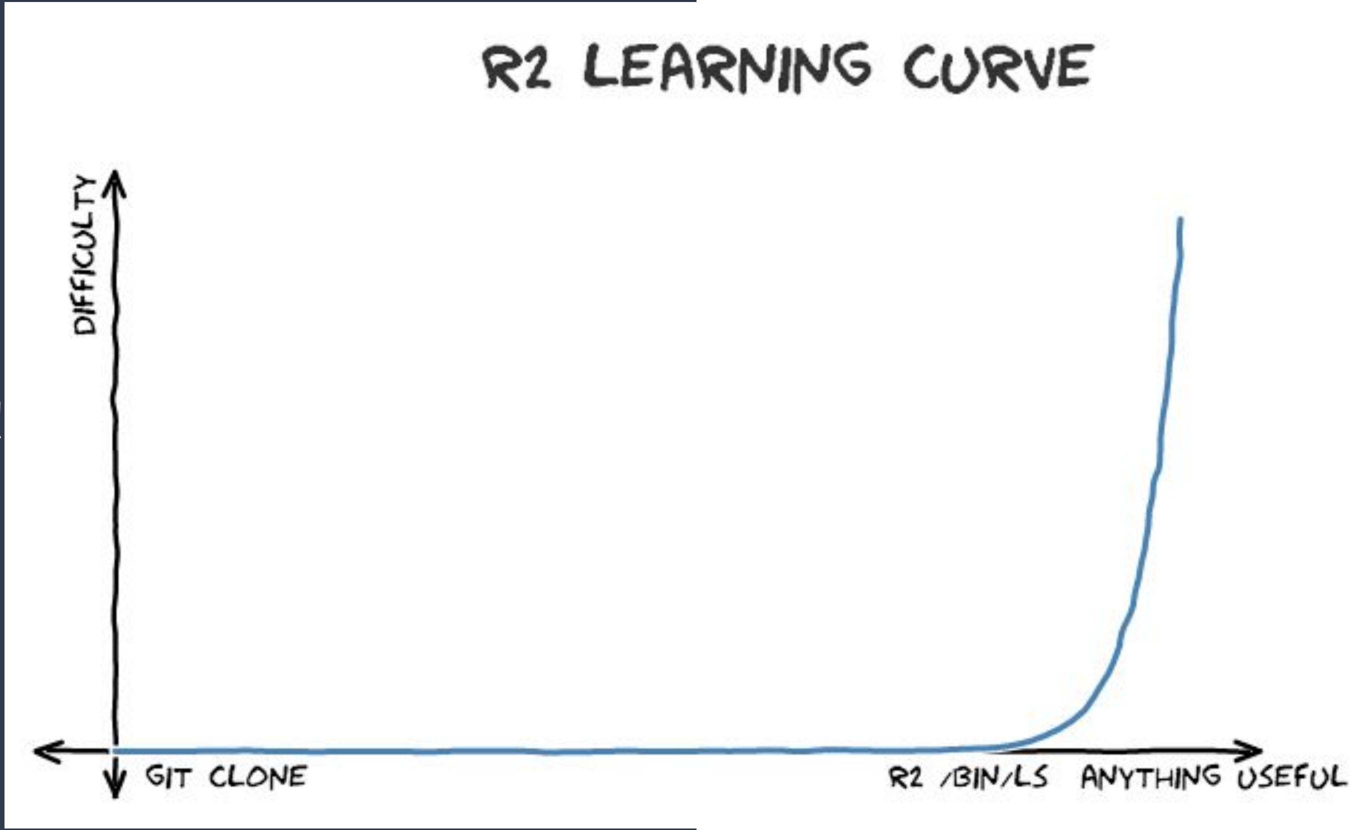
Searching: /

```
> / 'foo\0'  # search for string

> /c   # search asm code

> /m  # search magic numbers
```

Basic

$ r2 crackme

ntry0 (aa)



Useful subcommands

JSON output j
Most commands can output in JSON format using the ‘j’ suffix. Very useful for scripting!

```
 > pdj 1
 > drj
 > drj~{}                    # pretty prints JSON
```

Temporary seek @

```
 > cmd  @ addr       # run command cmd at addr
 > pdi 1 @ 0xcafebabe
```

Foreach @@

```
 > cmd  @@ off1 off2 ...   # run cmd at offsets
 > x @@c:cmd         # run x at the offsets returned by cmd
```

Visualization

```
 VV, Vv, V!, V_
```

Visual Assembler

```
 A       # in visual mode (V)
```

Basic debugging session

Debugging basics

```
> db 0x1337       # add bp at 0x1337
> dc              # continue
> ds              # step into
> dso                   # step over
> dr              # view registers
```

ESIL

- Evaluable Strings Intermediate Language
- General intermediate representation
- Allows to **emulate** most architectures r2 supports!

Emulation basics

```
> aei; aeim; aeip    # init emulation engine
> aes                # emulation step
> ae?                # emulation help
```

Cutter

- radare2’s GUI
- Supports most essential r2 features
- Emulation/debugging support in development
- FOSS
- Developed in Qt C++
- https://github.com/radareorg/cutter

Demo

Other r2 projects

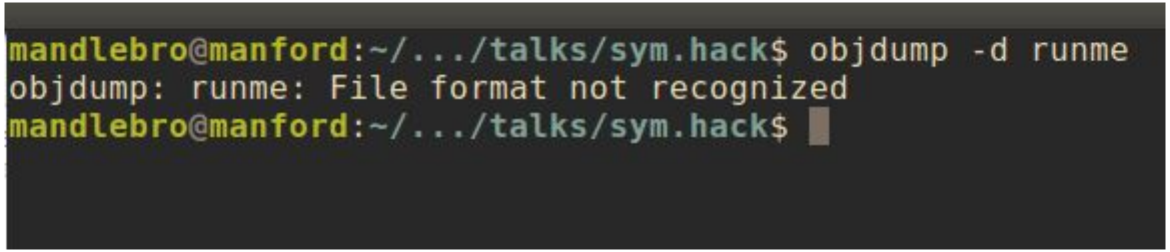
- radeco - radare2 decompiler project written in Rust
- rune - radare2 symbolic execution engine
- r2frida - radare2 integration with Frida

sym.hack challenge

Goal: Find the flag in the challenge!

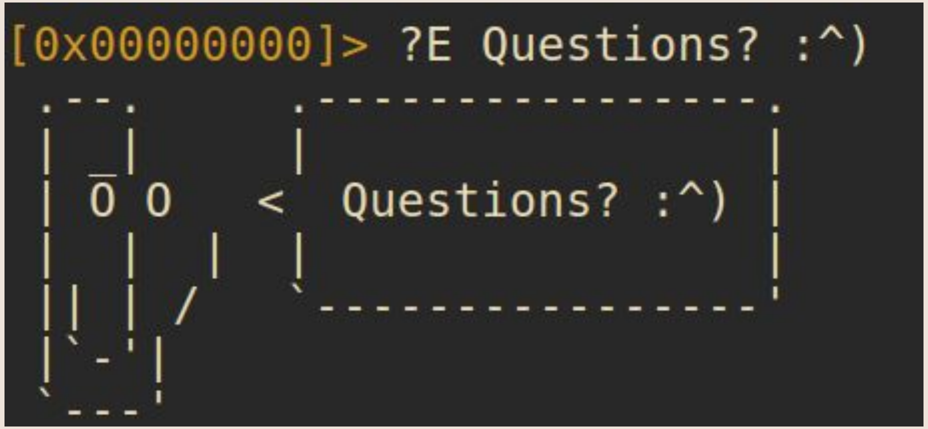
Link: https://fcasal.github.io/runme

Hint: use r2!



```
mandlebro@manford:~/.../talks/sym.hack$ objdump -d runme
objdump: runme: File format not recognized
mandlebro@manford:~/.../talks/sym.hack$ 
```

Questions?



```
[0x00000000]> ?E Questions? :^)
 .--.     .-----------------.
 | _|     |                 |
 | O O   <  Questions? :^)  |
 |  |  |  |                 |
 || | /   `-----------------'
 |`-'|
 `---'
```