Introducing radare2 for humans

(+surprise)

Arnau Gàmez i Montolio | @arnaugamez

September 7, 2018

R2CON2018 - Barcelona

Who am I

Arnau Gàmez i Montolio | @arnaugamez

- 21yo. Maths & CS student @ UB
- President @HackingLliure
- r2con collaborator
- Music (pianist), rubik's cubes...

*Who am I **NOT**: RE pro, r2 expert (at all), r2 dev (yet?)

Who are you

How many of you...

- Are students?
- Are working in infosec?
- Know radare2?
- Use radare2?

Preliminars

Who is this talk for:

- Newcomers to r2 / Quick reference of basics.
- People who did not attend the intro training.

Why this talk:

- Pancake can make more interesting stuff for r2con.
- Basic explanation of basic stuff by a (very) basic user.

Also, it will be recorded and uploaded for the Internet people.

Agenda

1. Introducing radare2
2. Basic usage
3. More features
4. Myths
5. Resources

Introducing radare2

What is radare2

- Free and open source reversing framework.
- (Re)written in C from radare(1) *by* ***pancake***.
- Built from scratch without any third-party dependency.
- Portable, scriptable, extensible via plugins.
- Release every 6 weeks.
- Great community.
- r2con: annual congress @ Barcelona (early Sept).

What can radare2 do (non-exhaustive)

- Disassemble binaries of several archs and OSs.
- Analise code, data, refs, structures.
- Debugging, tracing, exploiting.
- Binary manipulation, code injection, patching, bindiffing.
- Forensics: mount FS, detect partitions, data carving.
- Extract information and metrics for binary classification.
- Kernel analysis and debugging.

What can radare2 do (non-exhaustive)

radare2 has support for...

Architectures

i386, x86-64, ARM, MIPS, PowerPC, SPARC, RISC-V, SH, m68k, AVR, XAP, System Z, XCore, CR16, HPPA, ARC, Blackfin, Z80, H8/300, V810, V850, CRIS, XAP, PIC, LM32, 8051, 6502, i4004, i8080, Propeller, Tricore, Chip8 LH5801, T8200, GameBoy, SNES, MSP430, Xtensa, NIOS II, Dalvik, WebAssembly, MSIL, EBC, TMS320 (c54x, c55x, c55+, c66), Hexagon, Brainfuck, Malbolge, DCPU16

File Formats

ELF, Mach-O, Fatmach-O, PE, PE+, MZ, COFF, OMF, TE, XBE, BIOS/UEFI, Dyldcache, DEX, ART, CGC, Java class, Android boot image, Plan9 executable, ZIMG, MBN/SBL bootloader, ELF coredump, MDMP (Windows minidump), WASM (WebAssembly binary), Commodore VICE emulator, Game Boy (Advance), Nintendo DS ROMs and Nintendo 3DS FIRMs, various filesystems.

Operating Systems

Windows (since XP), GNU/Linux, OS X, [Net|Free|Open]BSD, Android, iOS, OSX, QNX, Solaris, Haiku, FirefoxOS

What can radare2 do (non-exhaustive)

TL;DR

-

Runs everywhere

Supports everything

Get radare2

Clone repo

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

Go to r2 created directory

```
$ cd radare2
```

Install/update

```
$ ./sys/install.sh #automatically pulls last version from git
```

*Check rada.re for Windows & Mac installation instructions.

Get radare2

Main tools included

- rabin2
- rax2
- rahash2
- radiff2
- rafind2
- rasm2
- r2pm
- radare2

Basic usage

Spawning an r2 shell

r2 command is a symlink for radare2

Open file

```
$ r2 /bin/ls
```

Open file in write mode

```
$ r2 -w /bin/ls
```

Open file in debug mode

```
$ r2 -d /bin/ls
```

Don't load settings or scripts

```
$ r2 -N /bin/ls
```

Alias for r2 malloc://512

```
$ r2 -
```

Open r2 with no file opened

```
$ r2 --
```

Basic commands

Commands follow simple mnemonic rules:

- s -> seek
- px -> print hexdump
- pd -> print disassembly
- wx -> write hexpairs
- wa -> write assembly
- aa -> analyse all (code)
- q -> quit

- Append ? to any command to get help about it.
- Temporary seek with @.

Handy tricks

- Append j (j~{}) for json (indented) output
  - Example: izj, izj~{}.
- Append q for quiet output.
  - Example: izq
- Pipe with shell commands.
  - Example: iz | less
- Run shell commands with ! prefix.
  - Example: !echo hello there r2con2018
- Internal grep with ~.
  - iz~string

Visual mode and Graph view

- Access visual mode with V command:
  - Rotate print mode with p command.
  - Press ? to get visual mode help.
  - Use : to run r2 command.
- Access graph view with VV command:
  - Really useful to see workflow of functions.
  - Have to be seeked on a function or won't show anything.
  - Move with arrows or hjkl.
  - Zoom in/out with +/-.

Configuration

Use e commands to tune radare2

Add ASM description

e asm.describe=true

Use UTF-8 chars

e scr.utf8=true

Enable truecolor

e scr.color=3

Enable temporary write

e io.cache=true

You can add e commands to ~/.radare2rc file for them to be loaded by default (remember -N to prevent r2 from parsing it).

More features

Scripting

- Bindings for many languages: C/C++, Java, Go, NodeJS, Perl, Python... (not really maintained)
- r2pipe API:
  - Input -> r2 commands. Output -> r2 output.
  - JSON deserialization to native objects.
- Python example:
  - Installation: pip(3) install r2pipe.
  - Usage: open(), cmd(), cmdj(), quit().

Debugging

- Debugging options under d command (Hint: use d?):
  - db -> set breakpoint
  - dc -> continue execution
  - ds -> step
- Starts debugging at dynamic loader (not entrypoint).
- Low level debugger. Not aiming to replace source one.
- Tiled visual mode V! is extremely useful here.
- Many backends:
  - gdb (in core).
  - r2frida (via r2pm): mem access, bin instrumentation, hooking...

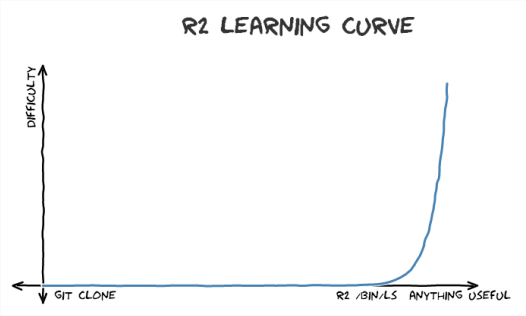
ESIL

- Stands for Evaluable Strings Intermediate Language.
- Standard intermediate language in r2.
- Each instruction is translated into a single string
  - mov eax 13 -> 33,eax,=
- Used for emulation and assisted debugging.
- Search expressions, predict jumpts, find references.
- ae subcommands used to manipulate the VM of ESIL (Hint: use ae?).

Myths

R2 IS DIFFICULT / MANY COMMANDS

- You can make 90% of things with basic commands showed.
- Simple mnemonic rules (again, append ? for inline help).
- Seriously, append ?.



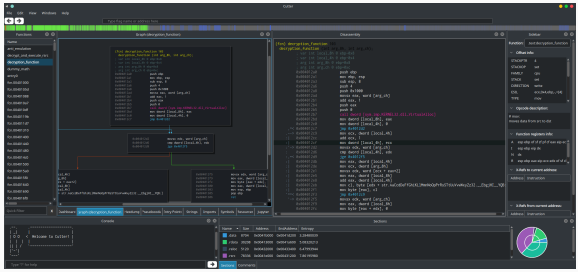
Scripting is hard

- Forget about native bindings & APIs.
- Just use r2pipe.
- If you know r2 then you know r2pipe.
- Seriously, use it.

WHERE RADARE2 MEETS PYTHON

There is no GUI

- Cutter: C++ & QT
- Released alongside r2 releases.
- Is there anyone who doesn't know it yet, seriously?



There is no documentation

- *"It's already documented in C"* –pancake
- radare2 book:
  - https://radare.gitbooks.io/radare2book
- radare2 explorations:
  - https://monosource.gitbooks.io/radare2-explorations

No, seriously.
Where is the documentation?

Surprise

R2DOCU

What

- Not aiming to replace r2book (at all).
- Not aiming to be a full reference.
- Common installation and very basics.

Why

- From zero to be working with r2 in one minute.
- r2 can/should be much more beginner friendly.
- Website ought to be updated *soon.

- It’s still a work in progress.
- Hopefully available alongside website update.
- Totally opened to suggestions and discussion about it.

Call for volunteers

Welcome

This documentation is aimed to be a quick and easy-to-follow first contact to radare2 (r2) so you can have it install and start using it in few minutes.

It is not meant to replace the comprehensive guides and explanations from the r2book neither a full reference of every single feature, but a place where newcomers can feel and put in practice the r2 power from the very beginning.

What is radare2

In a nutshell, r2 is a free and open source reverse engineering framework built from scratch in C without any third-party dependency that runs on very major (and many others) [platforms] and has native support for a lot of [file formats]. It can be used as is or you can built your own tools on top of r2 using your favourite scripting language.

What can radare2 do

At a first glance, here you are a non-exhaustive list of things that r2 can be useful for:

- Disassemble binaries of several [architectures] and [operating systems]
- Analise code, data, references and structures.
- Binary manipulation, code injection, patching, bindiffing.
- Mount filesystems, detect partitions, carve for data.
- Extract information and metrics that can be used for binary classification.
- Kernel analysis and debugging.

R2DOCU

Quick start

You can follow step-by-step the following commands.

Open a file

```
$ r2 /bin/ls
```

(You can prefix `-w` for opening in write mode or `-d` for opening in debug mode)

Basic commands

Commands in r2 follow simple mnemonic rules.

`s` -> seek [to (relative) address]

```
[0x00005850]> s 0x10
[0x00000010]> s 0
[0x00000000]> s 0x5850
[0x00005850]> s + 0x100
[0x00005950]>
```

`px` -> print hexdump [number of bytes]

```
[0x00005850]> px 64
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF
0x00005850  31ed 4989 d15e 4889 e248 83e4 f050 544c  1.I..^H..H...PTL
0x00005860  8d05 ca0a 0100 488d 0d53 0a01 0048 8d3d  ......H..S...H.=
0x00005870  1ce6 ffff ff15 5ea7 2100 f40f 1f44 0000  ......^.!....D..
0x00005880  488d 3de1 a921 0055 488d 05d9 a921 0048  H.=..!.UH....!.H
[0x00005850]>
```

`pd` -> print disassembly [number of instructions]

```
[0x00005850]> pd 5
            0x00005850      31ed           xor ebp, ebp
            0x00005852      4989d1         mov r9, rdx
            0x00005855      5e             pop rsi
            0x00005856      4889e2         mov rdx, rsp
            0x00005859      4883e4f0       and rsp, 0xfffffffffffffff0
[0x00005850]>
```

Resources

To know more

- Books: r2book, r2explorations.
- Talks (tons of them, just make a quick search):
  - r2con talks are uploaded few weeks after the congress.
- Blogposts (obviously non-exhaustive):
  - http://radare.today
  - https://www.megabeets.net
- Support & help:
  - IRC #radare at irc.freenode.net
  - Telegram: https://t.me/radare
- Attend to r2con ;)

Final advices

- There are many ways to collaborate with radare.
  - Not everyone has/knows to write (good) code.
- For the ones just starting. Don't be afraid.
  - Get dirty as soon as possible.
  - Ask for help.
- You think something is wrong? Don't be shy.
  - Fill an issue.
  - Send a PR.

Keep appending ? to every command for subcommands and inline help.

Questions?

Thank you

-

arnaugamez@pm.me
@arnaugamez | @HackingLliure