radare2 Reverse engineering toolkit

CONCEPTS

Commands are short letter chains: append ? to any prefix (a?, p?, pd?) to list what it offers.

Open r2 file
↓ aaa
Analyse functions, xrefs
↓ s / pdf
Inspect seek + print
↓ w / ood
Patch or debug write, step

Almost everything acts at the current seek; use cmd @ addr to run it elsewhere without moving.

Writing needs write mode: open with r2 -w, or reopen with oo+ . Without it, w commands fail.

OPEN AND ANALYSE

r2 -A *<file>*
Open and run full analysis

r2 -d *<file>*
Open under the debugger

r2 -q -c '*<cmds>***'** *<file>*
Run commands and quit

aaa
Deeper analysis of calls and refs

afl
List functions

axt *<addr>*
Find references to an address

BINARY INFO

iI
Binary info and protections

iz
Strings in data sections

ii
Imported functions

ie
Entry points

iS
Sections

rabin2 -I *<file>*
Binary info without opening r2

SEEK AND PRINT

s *<addr>*
Seek to address, flag or function

pdf
Disassemble current function

pd *<n>*
Disassemble n instructions

px [len]
Hexdump

pdc
Pseudo-decompile function

<cmd> **@** *<addr>*
Run cmd at addr without seeking

<cmd>~*<text>*
Grep the output

? *<expr>*
Evaluate a number expression

f
List flags

VISUAL MODE

V
Enter visual mode

VV
Graph view of function

p / P
Cycle print modes

h j k l
Move

Enter
Follow jump or call

u / U
Seek back / forward

x / X
Xrefs to / from here

:
Run a command

q
Back to shell

SEARCH

/ *<text>*
Search for a string

/x *<hex>*
Search hex bytes

WRITE AND PATCH

wx *<hex>*
Write hex bytes

wa *<asm>*
Assemble and write

oo+
Reopen in write mode

DEBUGGER

ood [args]
Reopen file in debug mode

dc
Start or continue

dcu *<addr>*
Continue until address

db *<addr>*
Set breakpoint

ds [n]
Step into

dso
Step over

dr
Show registers

dbt
Backtrace

r2 -d *<pid>*
Attach to a running process

COMPANION TOOLS

rasm2 -a x86 -b 64 '*<asm>***'**
Assemble to hex

rasm2 -a x86 -b 64 -d *<hex>*
Disassemble hex

RECIPES

First look at an unknown binary

```
rabin2 -I ./a.out
r2 -A ./a.out
iz~http
afl~main
s main; pdf
```

Find who uses a string

```
iz~password
# note the vaddr, say 0x00402010
axt 0x00402010
s <caller>; pdf
```