Radare2 REFERENCE CARD

Survival Guide

```
aa ...................................... auto analyse
pd@fcn⟨Tab⟩ .................... Disassemble function
f fcn⟨Tab⟩ ......................... List functions
f str⟨Tab⟩ ............................ List strings
fr fcn⟨Tab⟩ newname .............. Rename function
fr str⟨Tab⟩ newname ................ Rename string
```

Flagspaces

```
fs .................................. display flagspaces
fs * ............................. select all flagspace
fs [sections] ................... select one flagspace
```

Flags

```
f ............................................. list flags
fj ............................... display flags in json
fl .................................... show flag length
fx ........................... show hexdump of flag
fC [name] [cmt] .................. set flag comment
```

Infos

```
ii ..................................... Info on imports
iI .................................... Info on binary
ie ................................. Display entrypoint
iS ................................... Display sections
ir ................................. Display relocations
```

Visual mode

```
V .................................... Enter visual mode
p/P ....rotate modes (hex, disasm, debug, words, buf)
q .................................. back to radare shell
hjkl .... move around (or HJKL) (left-down-up-right)
Enter ..................... follow address of jump/call
sS .................................. step / step over
o ............................. go/seek to given offset
. ............................. seek to program counter
```

Searching part 1

```
/ ............... in cursor mode search in current block
:cmd ........................... run radare command
;[-]cmt ......................... add/remove comment
/*+-[] ........ change block size, [] = resize hex.cols
>||< ....................... seek aligned to block size
i/a/A (i)nsert hex, (a)ssemble code, visual (A)ssembler
b/B ......... toggle breakpoint / automatic block size
c/C ..................... toggle (c)ursor and (C)olors
d[f?] ................. define function, data, code, ..
D ............ enter visual diff mode (set diff.from/to)
e ................... edit eval configuration variables
f/F ................................... set/unset flag
gG .............. go seek to begin and end of file (0-$s)
mK/'K .................... mark/go to Key (any key)
M ....................... walk the mounted filesystems
n/N ....... seek next/prev function/flag/hit (scr.nkey)
o ............................. go/seek to given offset
R ...................... randomize color palette (ecr)
t ............ track flags (browse symbols, functions..)
T .................... browse anal info and comments
v .......................... visual code analysis menu
V/W ............. (V)iew graph (agv?), open (W)ebUI
uU ..................................... undo/redo seek
x ................... show xrefs to seek between them
yY ........................... copy and paste selection
z ................................... toggle zoom mode
```

Searching

```
/ foo\00 .................... search for string 'foo\0'
/b ................................. search backwards
// ................................ repeat last search
/w foo ............. search for wide string 'f\0o\0o\0'
/wi foo .......... search for wide string ignoring case
/!  ff .......search for first occurrence not matching
/i foo ............ search for string 'foo' ignoring case
/e /E.F/i ................. match regular expression
/x ff0033 ..................... search for hex string
/x ff..33  search for hex string ignoring some nibbles
/x ff43 ffd0 .......... search for hexpair with mask
/d 101112 .....search for a deltified sequence of bytes
/!x 00 ...inverse hexa search (find first byte != 0x00)
/c jmp [esp]  search for asm code (see search.asmstr)
/a jmp eax ..... assemble opcode and search its bytes
/A ..................... search for AES expanded keys
/r sym.printf .... analyze opcode reference an offset
/R ............................search for ROP gadgets
/P ................ show offset of previous instruction
/m magicfile .........search for matching magic file
/p patternsize ...... search for pattern of given size
/z min max ........... search for strings of given size
/v[?248] num ...look for a asm.bigendian 32bit value
```