

Radare2 REFERENCE CARD

Survival Guide

aa auto analyse
pdf@fcn<Tab> Disassemble function
f fcn<Tab> List functions
f str<Tab> List strings
fr [flagname] [newname] Rename flag
psz [offset] Print string
arf [flag] Find cross ref for a flag

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

Print string

psz [offset] Print stringZ'
psb [offset] Print strings in current block
psx [offset] Show string with scaped chars
psp [offset] Print pascal string
psw [offset] Print wide string

Visual mode

V Enter visual mode
p/P rotate modes (hex, disasm, debug, words, buf)
c toggle (c)ursor

q back to radare shell
h/j/k/l 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
/ 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
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
C toggle (C)olors
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

/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

Saving

Po [file] open project
Ps [file] save project
Pi [file] show project informations

Usable variables in expression

\$\$ here (current virtual seek)
\$o here (current disk io offset)
\$s file size
\$b block size
\$w get word size, 4 if asm.bits=32, 8 if 64
\$c,\$r get width and height of terminal
\$\$ section offset
\$\$S section size
\$j jump address (jmp 0x10, jz 0x10 => 0x10)
\$f jump fail address (jz 0x10 => next instruction)
\$I number of instructions of current function
\$F current function size
\$Jn get nth jump of function
\$Cn get nth call of function
\$Dn get nth data reference in function
\$Xn get nth xref of function
\$m opcode memory reference (mov eax,[0x10] => 0x10)
\$l opcode length
\$e 1 if end of block, else 0
\$ev get value of eval config variable
\$? last comparison value