
The State Of The Art
in r2land
ArkCon 2019
by pancake
ARKCON

Who Am I?

- Sergi Àlvarez i Capilla (known as '**pancake**')
- Senior Mobile Security Research Engineer at **NowSecure**
- Building tools to make the mobile ecosystem safer
- Author of Radare, Acr, Valabind, 0xFFF and many other OSS

What is r2?

Hopefully at some point I will be able to remove this slide from my presentations O:)

- Free/Libre framework and tooling for reverse engineering
- Follows the UNIX principles, small, portable and fast
- Huge and friendly community

What's this talk about?

Maybe you remember a talk from me at r2con named

"hidden gems in r2land"

- R2land is an imaginary place where all the r2 users and devs live
- It's a place full of hidden or little known places and features.
- Many people is scared of it (because of.... "TEXT!")
- This talk aims to show some of those gems and show new stuff

Strengths

Of radare2

- UNIX friendly
- Powerful command line
- Very portable and small
- Flexible and extensible
- Active Community
- Easy to contribute
- Lots of resources

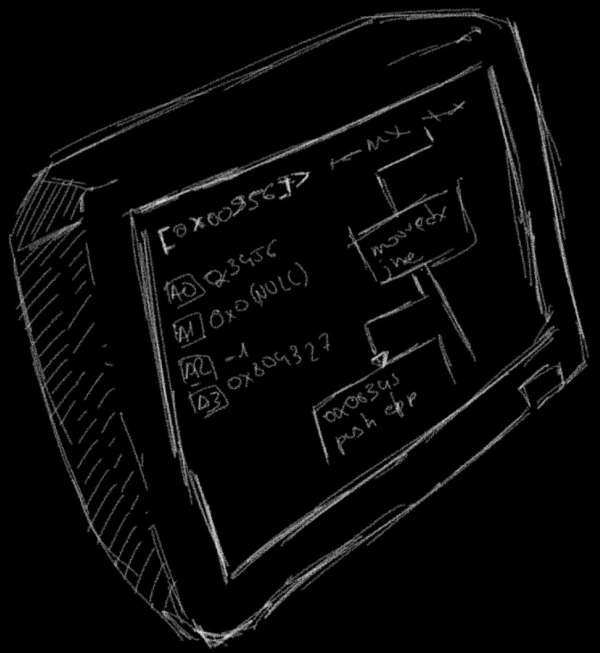
Weaknesses

Of radare2

- Moving Target
- Always WIP
- No backward compatibility
- Obscure/cryptic
- So many open fronts

User Interfaces

- Shell
- Visual
- Panels
- Graph
- Cutter
- WebUI



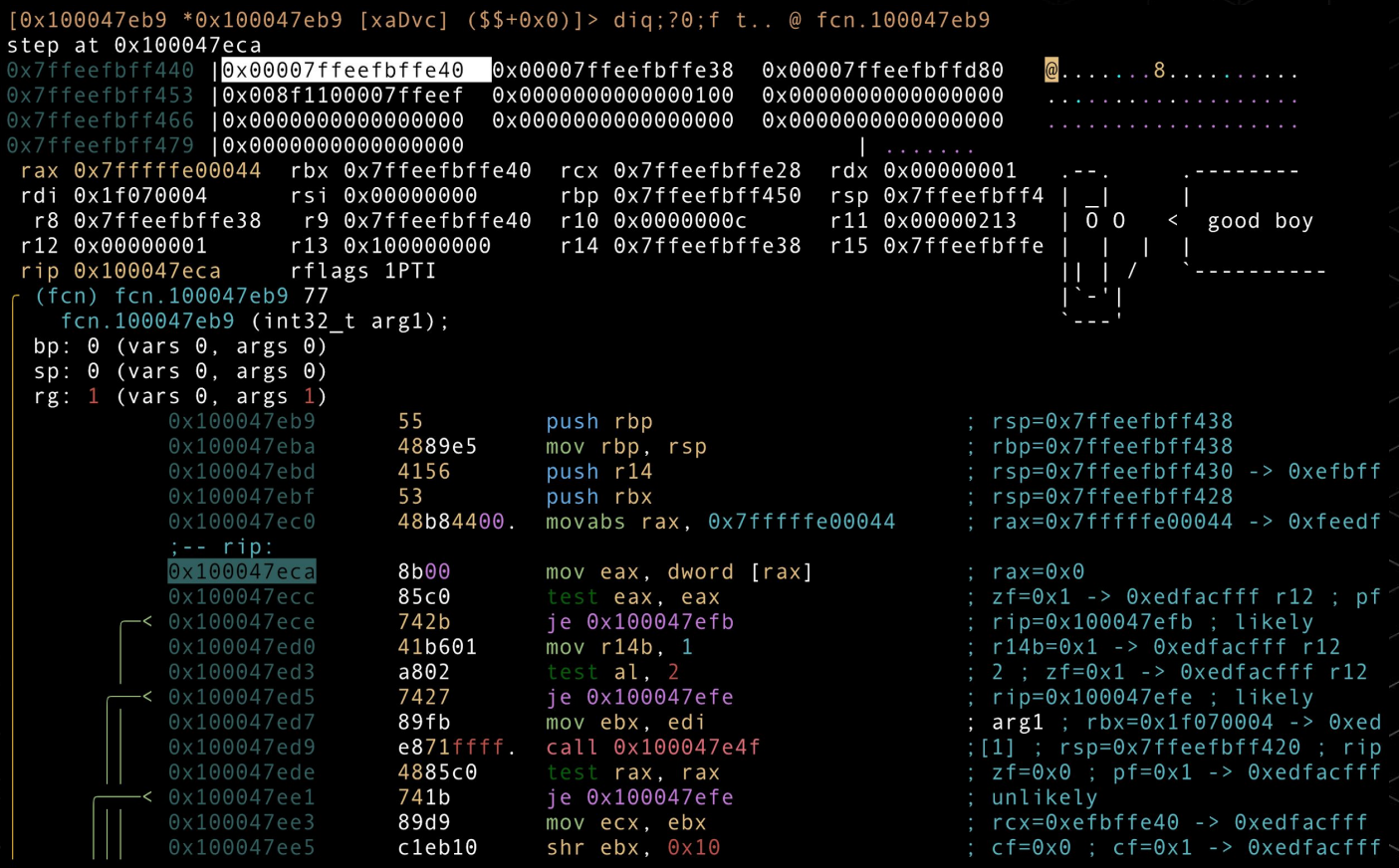
Developers

- C API / Bindings
- R2pipe
- R2Pipe-API

Shell

```
$ r2 /bin/ls
 -- In radare we trust
[0x1000011e8]> pd 10
            ;-- main:
            ;-- entry0:
            ;-- func.1000011e8:
            ;-- rip:
            0x1000011e8      55          push rbp
            0x1000011e9      4889e5      mov rbp, rsp
            0x1000011ec      4157        push r15
            0x1000011ee      4156        push r14
            0x1000011f0      4155        push r13
            0x1000011f2      4154        push r12
            0x1000011f4      53          push rbx
            0x1000011f5      4881ec18.   sub rsp, 0x618
            0x1000011fc      4989f7      mov r15, rsi
            0x1000011ff      4189fe      mov r14d, edi
[0x1000011e8]> px 64
- offset -   E8E9 EAEB ECED EEEF F0F1 F2F3 F4F5 F6F7  89ABCDEF01234567
0x1000011e8  5548 89e5 4157 4156 4155 4154 5348 81ec  UH..AWAVAUATSH..
0x1000011f8  1806 0000 4989 f741 89fe 488d 85c0 fdff  ....I..A..H.....
0x100001208  ff48 8945 d085 ff7f 05e8 dc31 0000 488d  .H.E.......1..H.
0x100001218  35cb 3800 0031 ffe8 4633 0000 bf01 0000  5.8..1..F3......
[0x1000011e8]>
```

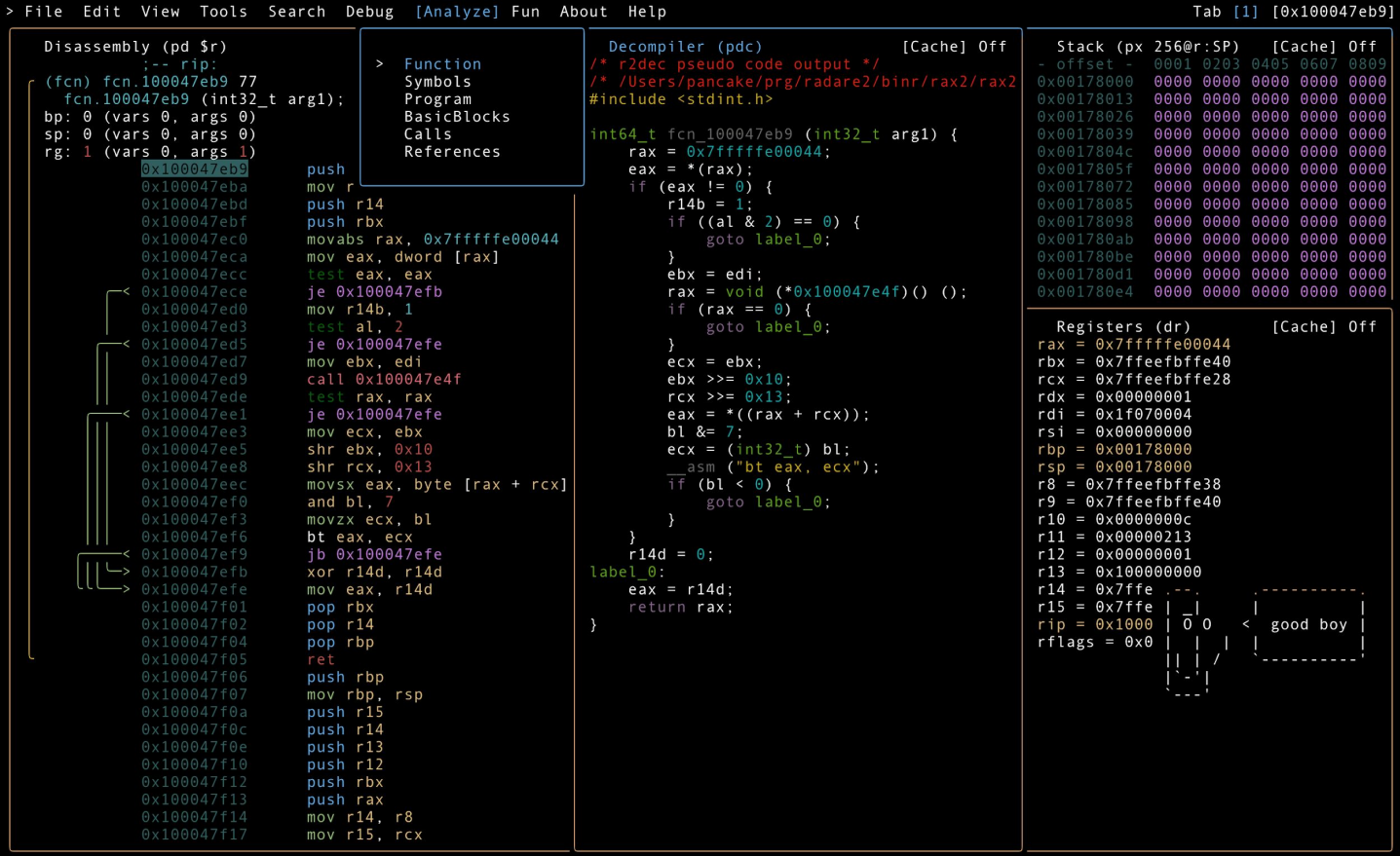
Visual


[0x100047eb9 *0x100047eb9 [xaDvc] ($$+0x0)]> diq;?0;f t.. @ fcn.100047eb9
step at 0x100047eca
0x7ffeefbff440 |0x00007ffeefbffe40 0x00007ffeefbffe38 0x00007ffeefbffd80 @.......8.........
0x7ffeefbff453 |0x008f1100007ffeef 0x0000000000000100 0x0000000000000000 ....
0x7ffeefbff466 |0x0000000000000000 0x0000000000000000 0x0000000000000000 ....
0x7ffeefbff479 |0x0000000000000000 | ....
rax 0x7fffffe00044 rbx 0x7ffeefbffe40 rcx 0x7ffeefbffe28 rdx 0x00000001
rdi 0x1f070004 rsi 0x00000000 rbp 0x7ffeefbff450 rsp 0x7ffeefbff4
r8 0x7ffeefbffe38 r9 0x7ffeefbffe40 r10 0x0000000c r11 0x00000213
r12 0x00000001 r13 0x100000000 r14 0x7ffeefbffe38 r15 0x7ffeefbffe
rip 0x100047eca rflags 1PTI
.--. .--------
| _| |
| O O < good boy
| | | |
|| | / `----------
|`-'|
`---'
(fcn) fcn.100047eb9 77
fcn.100047eb9 (int32_t arg1);
bp: 0 (vars 0, args 0)
sp: 0 (vars 0, args 0)
rg: 1 (vars 0, args 1)
0x100047eb9 55 push rbp ; rsp=0x7ffeefbff438
0x100047eba 4889e5 mov rbp, rsp ; rbp=0x7ffeefbff438
0x100047ebd 4156 push r14 ; rsp=0x7ffeefbff430 -> 0xefbff
0x100047ebf 53 push rbx ; rsp=0x7ffeefbff428
0x100047ec0 48b84400. movabs rax, 0x7fffffe00044 ; rax=0x7fffffe00044 -> 0xfeedf
;-- rip:
0x100047eca 8b00 mov eax, dword [rax] ; rax=0x0
0x100047ecc 85c0 test eax, eax ; zf=0x1 -> 0xedfacfff r12 ; pf
< 0x100047ece 742b je 0x100047efb ; rip=0x100047efb ; likely
0x100047ed0 41b601 mov r14b, 1 ; r14b=0x1 -> 0xedfacfff r12
0x100047ed3 a802 test al, 2 ; 2 ; zf=0x1 -> 0xedfacfff r12
< 0x100047ed5 7427 je 0x100047efe ; rip=0x100047efe ; likely
0x100047ed7 89fb mov ebx, edi ; arg1 ; rbx=0x1f070004 -> 0xed
0x100047ed9 e871ffff. call 0x100047e4f ;[1] ; rsp=0x7ffeefbff420 ; rip
0x100047ede 4885c0 test rax, rax ; zf=0x0 ; pf=0x1 -> 0xedfacfff
< 0x100047ee1 741b je 0x100047efe ; unlikely
0x100047ee3 89d9 mov ecx, ebx ; rcx=0xefbffe40 -> 0xedfacfff
0x100047ee5 c1eb10 shr ebx, 0x10 ; cf=0x0 ; cf=0x1 -> 0xedfacfff

New Visual Modes

- Tabs in visual and panels
- Folding functions and basic blocks
- Nn scr.nkeys
- Scrollbar and navigation bar
- Gadgets and Top/Side commands
- 2 Dimensional Views (print modes vs alternatives)
- Bit editor
- Esil debugger
- Browse types, vars, imports, symbols, functions, ..
- Snow/Sakura

Panels


> File Edit View Tools Search Debug [Analyze] Fun About Help
Tab [1] [0x100047eb9]
Disassembly (pd $r)
;-- rip:
(fcn) fcn.100047eb9 77
fcn.100047eb9 (int32_t arg1);
bp: 0 (vars 0, args 0)
sp: 0 (vars 0, args 0)
rg: 1 (vars 0, args 1)
0x100047eb9 push
0x100047eba mov r
0x100047ebd push r14
0x100047ebf push rbx
0x100047ec0 movabs rax, 0x7fffffe00044
0x100047eca mov eax, dword [rax]
0x100047ecc test eax, eax
0x100047ece je 0x100047efb
0x100047ed0 mov r14b, 1
0x100047ed3 test al, 2
0x100047ed5 je 0x100047efe
0x100047ed7 mov ebx, edi
0x100047ed9 call 0x100047e4f
0x100047ede test rax, rax
0x100047ee1 je 0x100047efe
0x100047ee3 mov ecx, ebx
0x100047ee5 shr ebx, 0x10
0x100047ee8 shr rcx, 0x13
0x100047eec movsx eax, byte [rax + rcx]
0x100047ef0 and bl, 7
0x100047ef3 movzx ecx, bl
0x100047ef6 bt eax, ecx
0x100047ef9 jb 0x100047efe
0x100047efb xor r14d, r14d
0x100047efe mov eax, r14d
0x100047f01 pop rbx
0x100047f02 pop r14
0x100047f04 pop rbp
0x100047f05 ret
0x100047f06 push rbp
0x100047f07 mov rbp, rsp
0x100047f0a push r15
0x100047f0c push r14
0x100047f0e push r13
0x100047f10 push r12
0x100047f12 push rbx
0x100047f13 push rax
0x100047f14 mov r14, r8
0x100047f17 mov r15, rcx
> Function
Symbols
Program
BasicBlocks
Calls
References
Decompiler (pdc) [Cache] Off
/* r2dec pseudo code output */
/* /Users/pancake/prg/radare2/binr/rax2/rax2
#include <stdint.h>
int64_t fcn_100047eb9 (int32_t arg1) {
rax = 0x7fffffe00044;
eax = *(rax);
if (eax != 0) {
r14b = 1;
if ((al & 2) == 0) {
goto label_0;
}
ebx = edi;
rax = void (*0x100047e4f)() ();
if (rax == 0) {
goto label_0;
}
ecx = ebx;
ebx >>= 0x10;
rcx >>= 0x13;
eax = *((rax + rcx));
bl &= 7;
ecx = (int32_t) bl;
__asm ("bt eax, ecx");
if (bl < 0) {
goto label_0;
}
}
r14d = 0;
label_0:
eax = r14d;
return rax;
}
Stack (px 256@r:SP) [Cache] Off
- offset - 0001 0203 0405 0607 0809
0x00178000 0000 0000 0000 0000 0000
0x00178013 0000 0000 0000 0000 0000
0x00178026 0000 0000 0000 0000 0000
0x00178039 0000 0000 0000 0000 0000
0x0017804c 0000 0000 0000 0000 0000
0x0017805f 0000 0000 0000 0000 0000
0x00178072 0000 0000 0000 0000 0000
0x00178085 0000 0000 0000 0000 0000
0x00178098 0000 0000 0000 0000 0000
0x001780ab 0000 0000 0000 0000 0000
0x001780be 0000 0000 0000 0000 0000
0x001780d1 0000 0000 0000 0000 0000
0x001780e4 0000 0000 0000 0000 0000
Registers (dr) [Cache] Off
rax = 0x7fffffe00044
rbx = 0x7ffeefbffe40
rcx = 0x7ffeefbffe28
rdx = 0x00000001
rdi = 0x1f070004
rsi = 0x00000000
rbp = 0x00178000
rsp = 0x00178000
r8 = 0x7ffeefbffe38
r9 = 0x7ffeefbffe40
r10 = 0x0000000c
r11 = 0x00000213
r12 = 0x00000001
r13 = 0x100000000
r14 = 0x7ffe
r15 = 0x7ffe
rip = 0x1000
rflags = 0x0
good boy

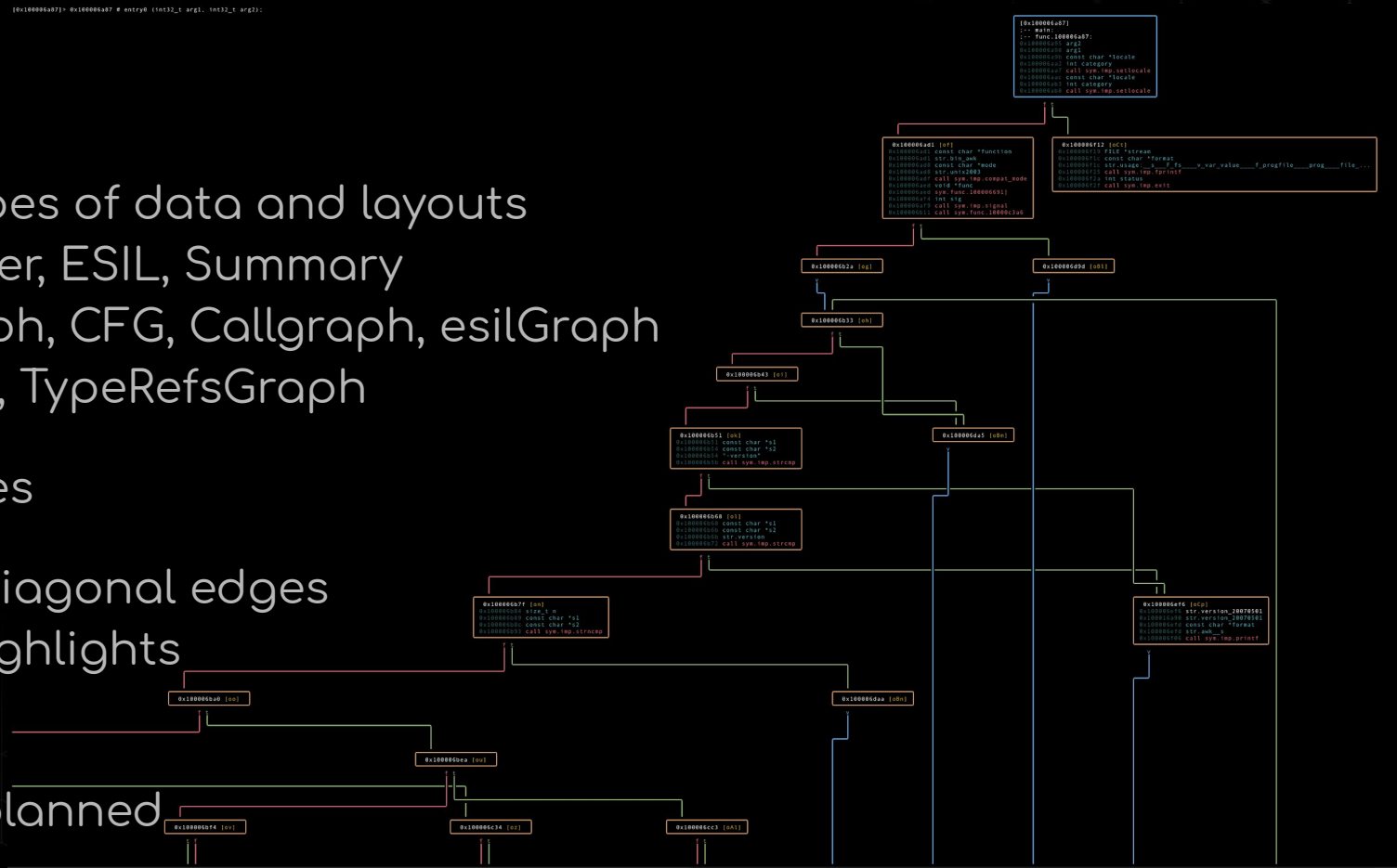
Graph

- Many types of data and layouts
- Decompiler, ESIL, Summary
- Tracegraph, CFG, Callgraph, esilGraph
- RefGraph, TypeRefsGraph

Different styles

- Square/diagonal edges
- Colors/highlights
- Folding

Grouping is planned

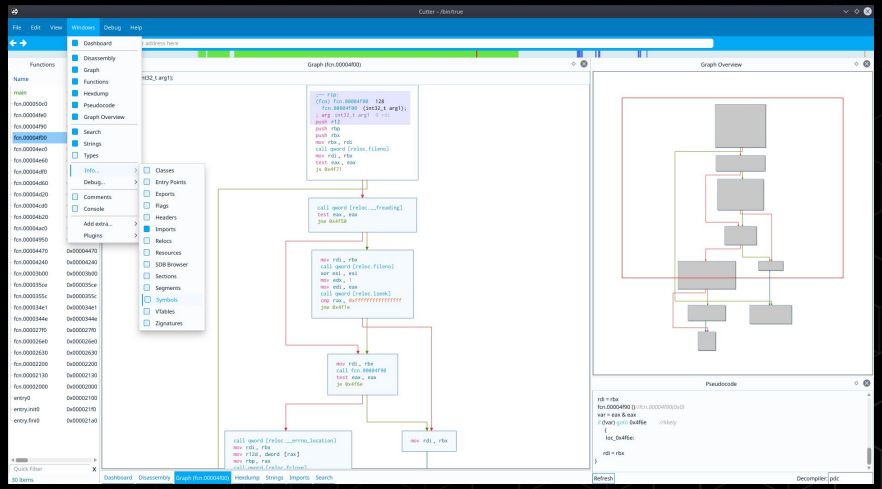


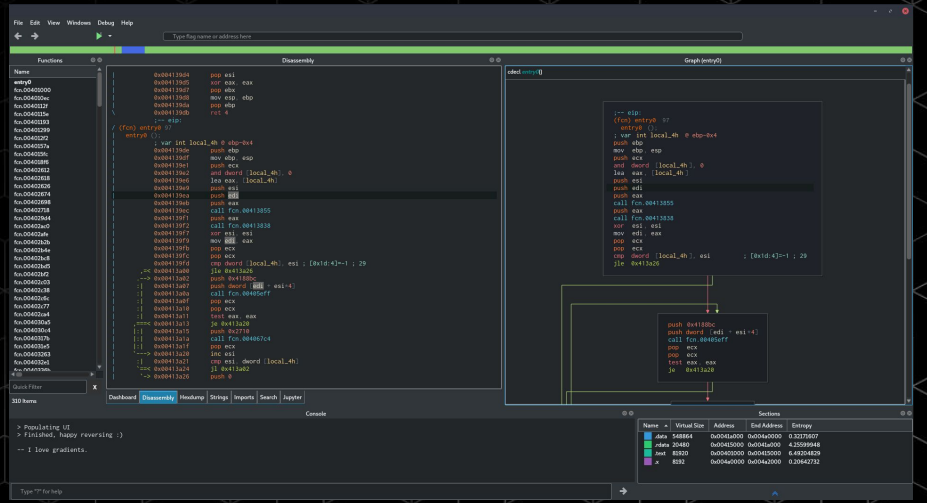
Cutter

The official Graphical User Interface for radare2

Multi Platform: Windows, macOS and GNU/Linux

- Releases 1 week after each r2 release





Extending r2

Extending and Scripting r2

As long as C apis are planned to be continuously refactored and improved, it's easier and simpler to use r2pipe and commands.

- Newest additions! r2pipe.sh and Prolog!

IO plugins thru the cmd interface can implement fs, debugger, ..

- Language bindings and native API is not recommended
- We need more hands to get all that stable!
- r2pipe-api is the best solution for idiomatic scripting

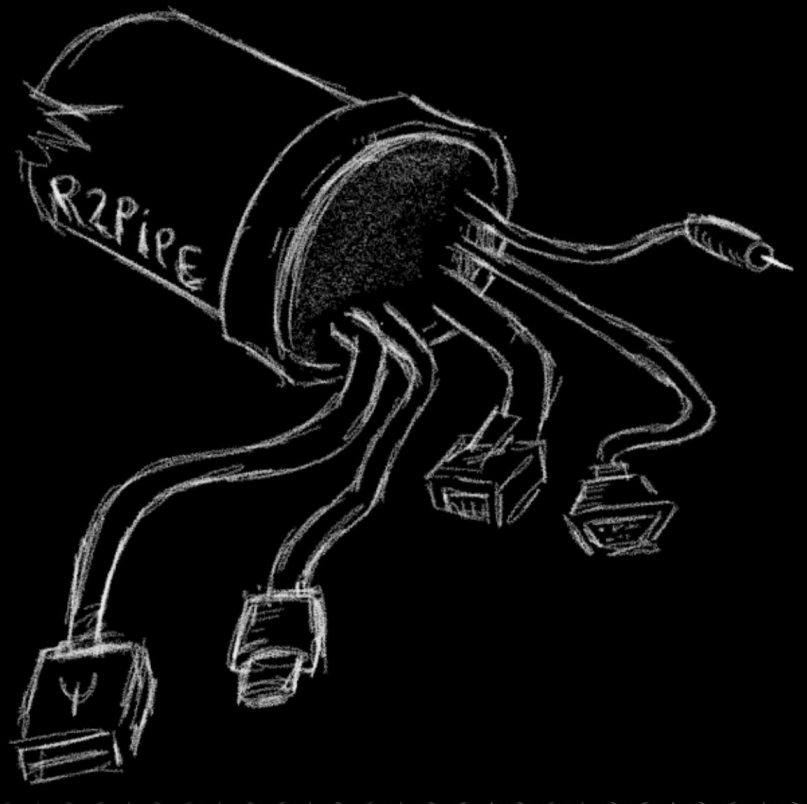
The r2pm provides a large list of packages to be used with r2...

Plugins

- R2Dec (retdec, ghidra, ..)
- Frida
- Yara
- Miasm (+ Sibyl)
- Kaitai
- Ewf
- Keystone
- Angr (r4ge, r2angr)
- MSDN, winapi documentation
- Unicorn Emulator
- Various Decompilers

Interacts With

- QIRA
- Windbg
- Winedbg
- Lldb
- Gdbserver
- Frida
- Bochs
- QEmu
- GHIDRA
- IDA



r2frida

r2 plugin to use frida as an IO handler

- Developed by me as an open-source project at **NowSecure**
- Use all the features of r2, even scripting with r2pipe on Frida
- Attach or spawn on local tcp remote or USB connections
- Supports v8 and duktape backends
- Supports Google Chrome developer tools
- IO plugins have a command interface to run cmds in the agent
    - Handle debugger, filesystem r_io primitives via r2pipe

r2frida plugins

- R2Frida can be extended in Javascript
  - Those files are dynamically loaded in the agent
  - Use babel to one-ify, es5 and compact the code
  - Requires a constructor, a destructor
- Handle commands executed, same as RCore plugins in r2

Some examples in the plugins/ directory

- Hook IO
- Run commands in r2 from the Frida agent
- Reuse any script from CodeShare
- ...

FileSystems

R2 have a virtual filesystem api with plugins that understand filesystem formats like fat, ntfs, hfs+, ext3 or reiserfs.

Alias files are not real files (just live inside the current session)

Can be extended by abusing the io cmd plugin interface.

- m io / 0

We can use this to pull and modify remote files via frida:// f.ex

Decompiler

- R2 comes with a very basic pseudo decompiler in “pdc”
  - Output is almost always buggy and unreliable, but is fast, works everywhere and gives u a quick understanding of what's going on without having to install anything else
- In r2land we have 2 native decompilers
  - R2dec - js
  - Radeco - rust
- But we support **retdec**, **snowman** and **ghidra**

R2dec is the easiest to use and best integrated, in opensource, things only improve when users use and report issues, we fix bugs fast!

- Handles constructions made from ObjC, Dalvik and C.

Decompiler and Tracing Graphs

R2dec can import all the decompiler output as comments

Using the # key in visual or graph view will toggle a mode to only display those comments, having a disasm like view but reading it as C

- r2 -Ac .pdd* /bin/ls
- v#

WIP support for decompiler debugger and emulation support

Emulation

Every RE tool out there have its own IL, and r2 is not an exception. I designed ESIL (Evaluable Strings Intermediate Language).

- Forth like VM as a microcode to emulate each instruction

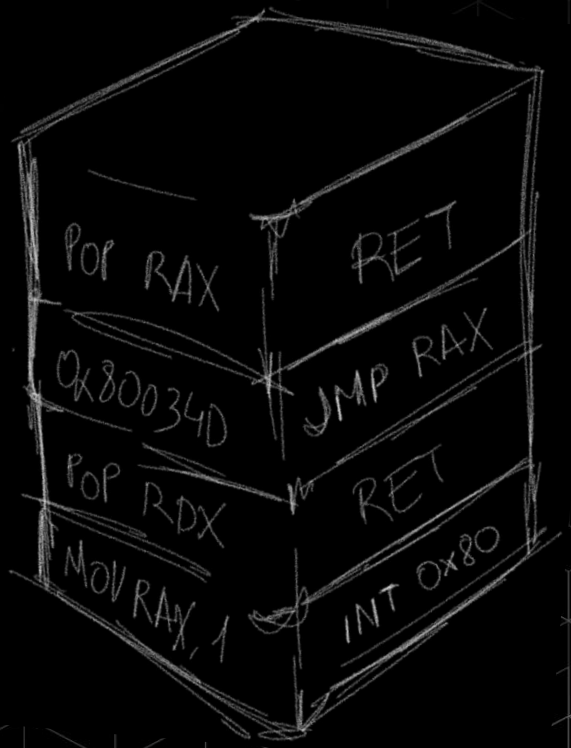
Used in many places inside r2.

- Search for addresses matching given ESIL expressions
- Convert it to other formats (graph, reil, ssa, ..)
- Identify register usage in functions (inputs, outputs, ..)
- Emulation of code, asisted debugging, sw watchpoints ..

But also used externally by radeco, rune, ... (and ghidra at some point)

ROP Gadgets

- Search for gadgets
- Rarop NodeJS web tool to construct the chains with d&d
- But now there's a Visual mode to create a ropchain without depending on a browser.
- WIP: emulate the ropchain , inject in target or dummy process to execute it.



Zignatures

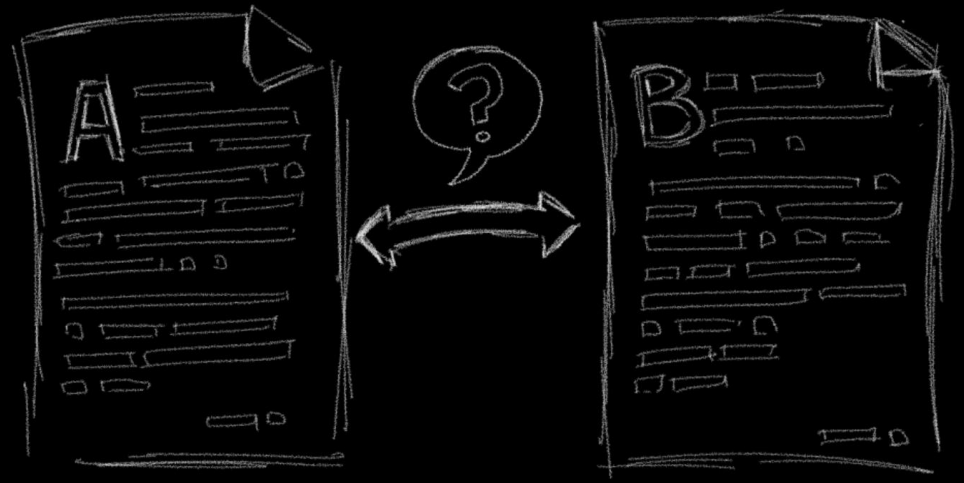
Extract metrics from functions for later comparing or diffing.

- Realname
- Comments
- Bytes and mask
- Control flow graph metrics
- Local Variables and Function Arguments
- Xrefs and Refs
- More to come!
    - Decompilation, Primer numbers, Simhash to reduce
    - Better ponderations

Diffing

There are many ways to diff code and data, r2 provides the tools to perform such actions:

- Byte-by-byte
- Delta diffing
- Levenstein Diff
- Code text diff
- Graph Diffing
- Function Diffing
- Zignatures Diffing

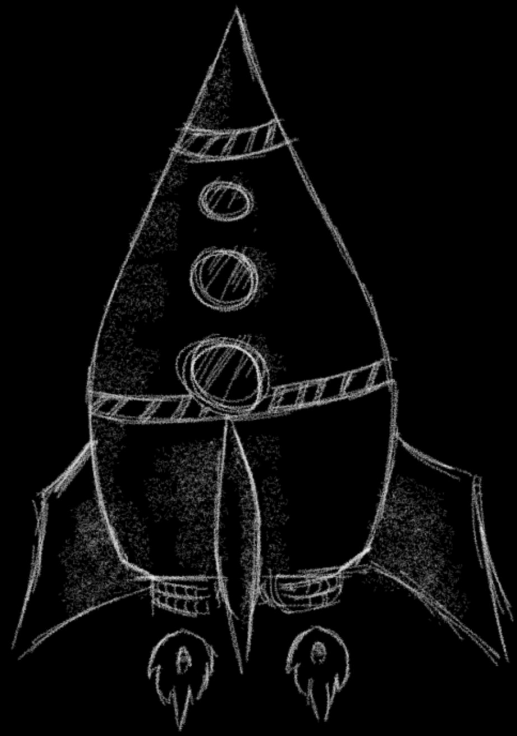


Firmware and Large File Visualizations

- Entropy/Statistical bars and dumps
- scr.scrollbar
- Types propagation
- Structs visualization
- Improved support for Thumb analysis (jump tables, ...)
- Data visualization in colors
- QR codes
- Searching for instruction types
- Matching magic
- afl=

Future

- Improve everything
  - Feedback is highly appreciated!
- Projects
  - Requires refactoring on many modules
- Undo/redo
  - Not just for seeks and writes
- Real Time Syncing
  - Already done for some data
- API and ABI stability
  - Not just the commands



RSoC & R2Con2019

- We didn't make it into the GSoC this year
  - So we organized our own **Summer Of Code**
  - Nowsecure and Tencent are sponsoring the two students
    - Types Analysis Improvements
    - User Interaction and Visual Stuff

Open conference for users and developers of r2:

- Only technical talks related to r2. (Reversing, exploiting, forensics..)
- 2 days of trainings + 2 days of conference talks

https://rada.re/con/2019

Questions?

Ktxby