Radeco Pseudo C Code Generation

GSoC’2018

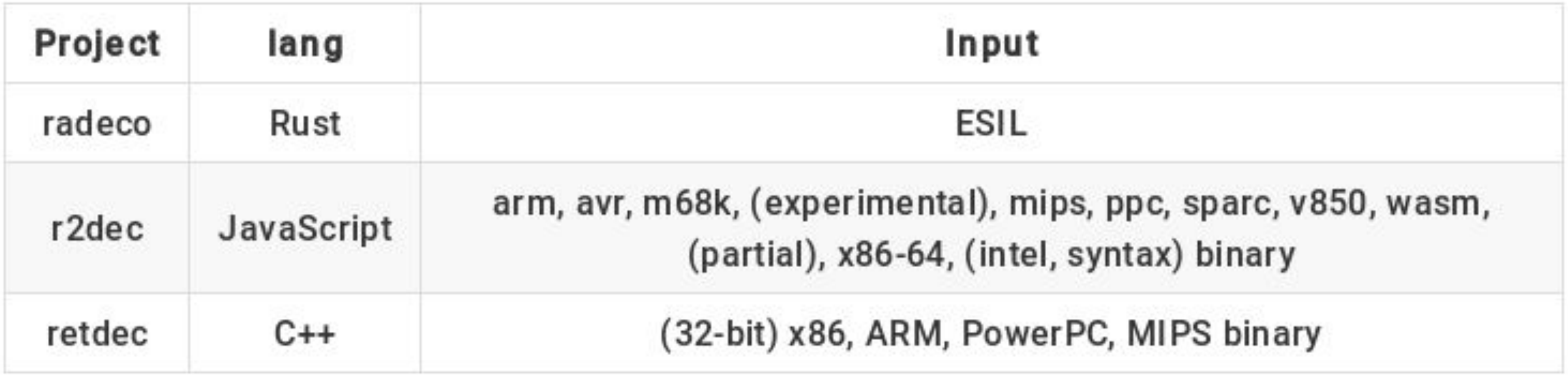
Radeco Pseudo C Code Generation

I have done

- writing pseudo C Code generator
- writing r2 integration

Demo

Decompilers



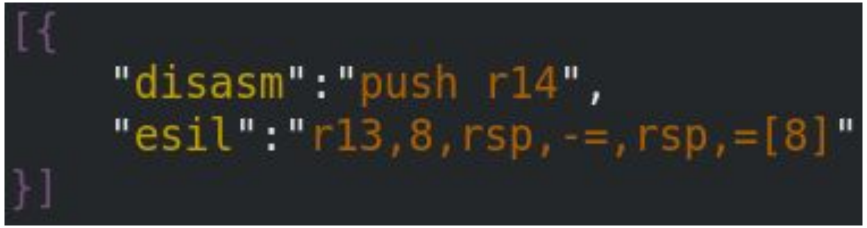
Project	lang	Input
radeco	Rust	ESIL
r2dec	JavaScript	arm, avr, m68k, (experimental), mips, ppc, sparc, v850, wasm, (partial), x86-64, (intel, syntax) binary
retdec	C++	(32-bit) x86, ARM, PowerPC, MIPS binary

Projects

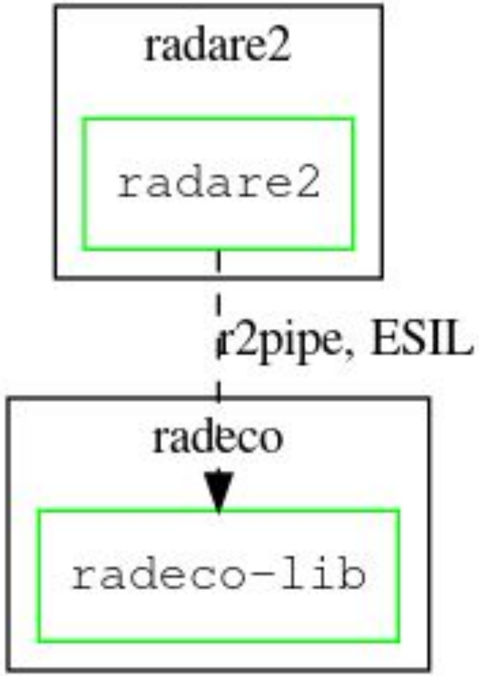
- radeco-lib
  - https://github.com/radareorg/radeco-lib
  - Core library for binary analysis, decompilation
- radeco
  - https://github.com/radareorg/radeco
  - User interface

Design (radare2 to radeco)

Input (ESIL)



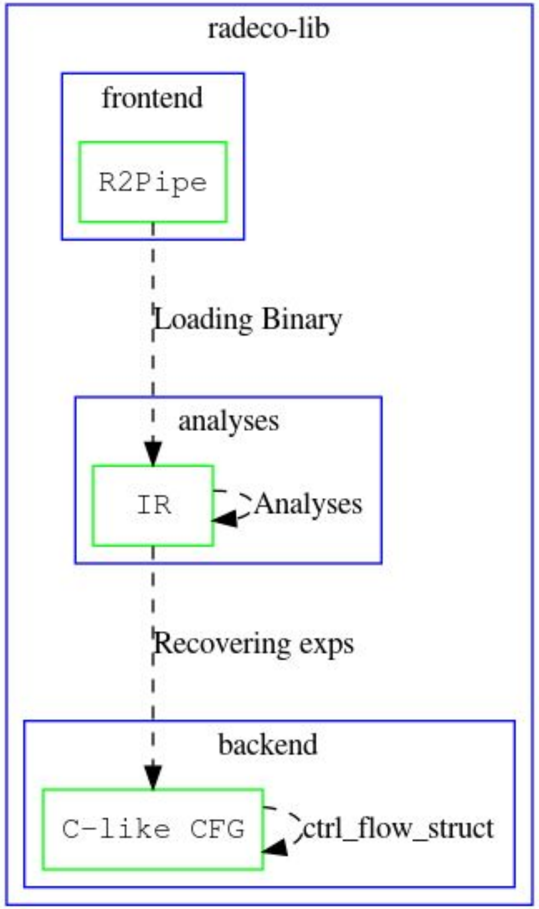
```
[{
    "disasm":"push r14",
    "esil":"r13,8,rsp,-=,rsp,=[8]"
}]
```



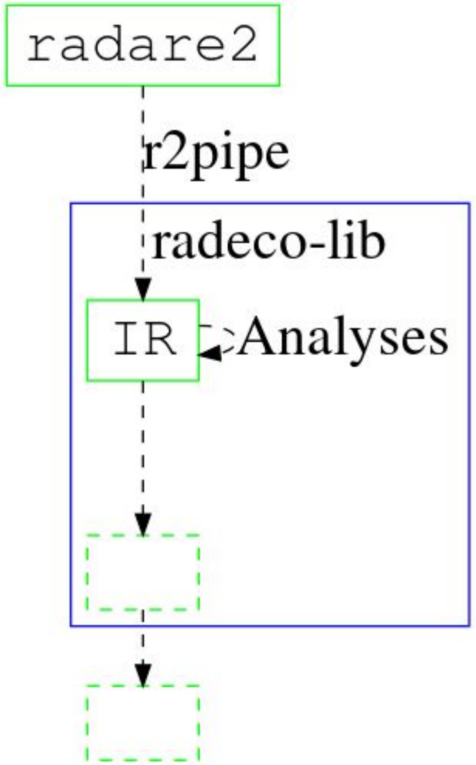
Design (radeco-lib)

radeco has 3 stages

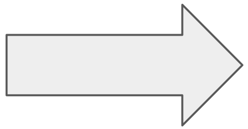
- Loading Binary (ESIL)
- Analyses
    - deadcode elimination, ...
- Decompilation
    - heuristics
    - control flow structuring

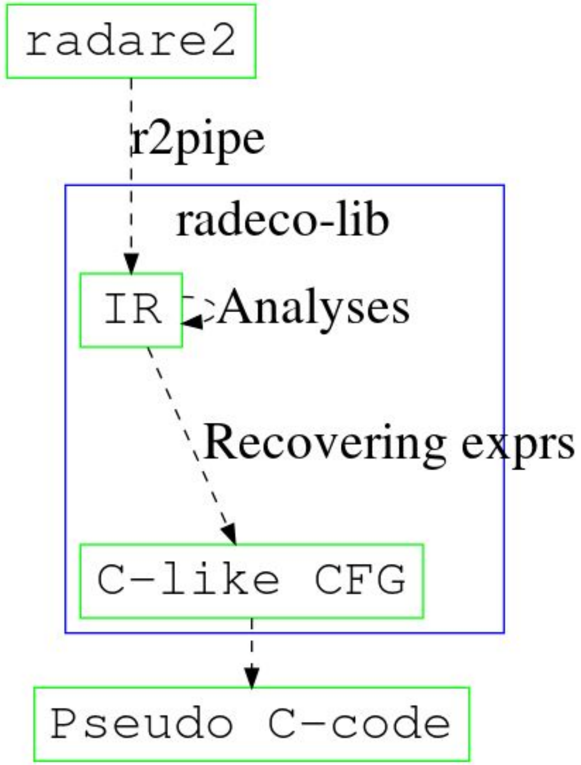


Works



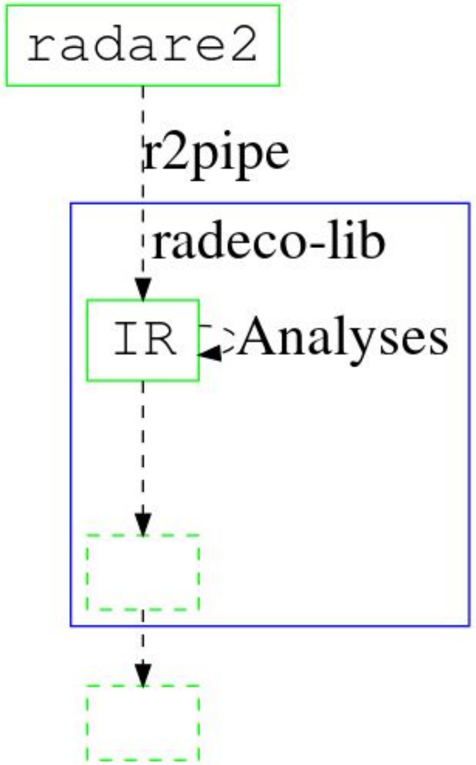
Before GSoC



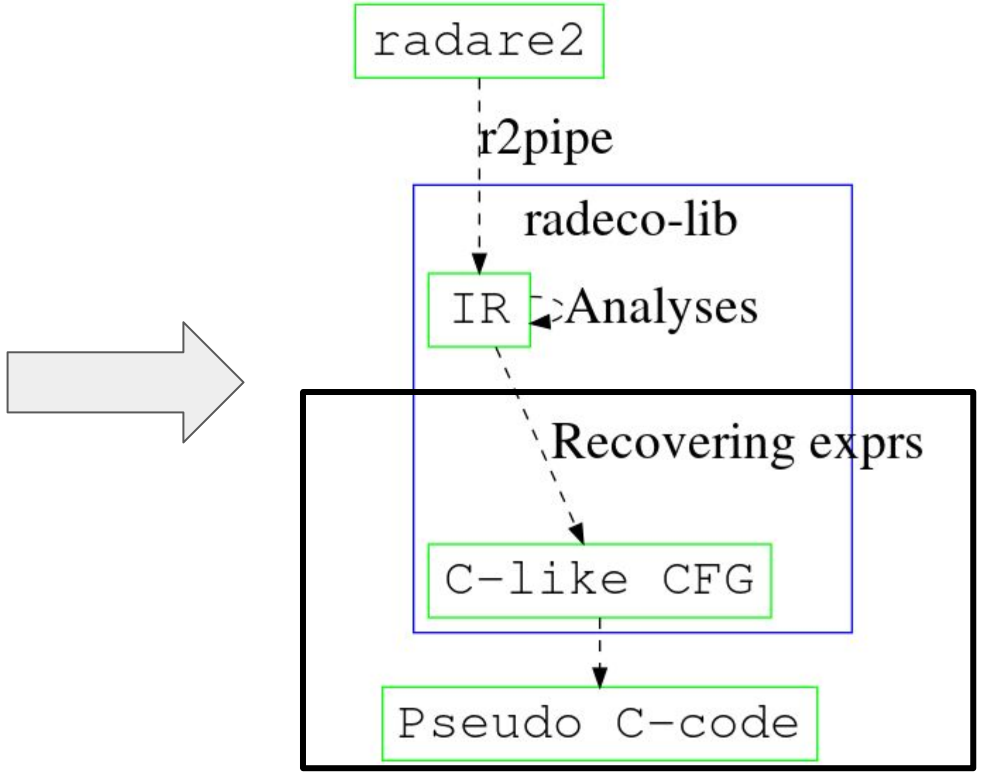


After GSoC

Works



Before GSoC

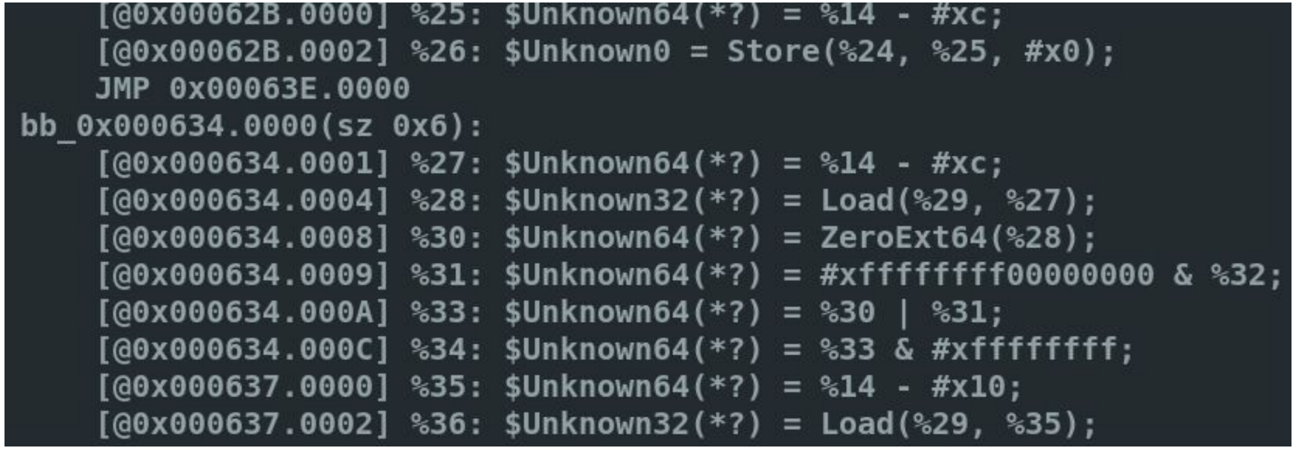


After GSoC

IR

Assembly-like intermidiate representation

- Generated from ESIL
- SSA form

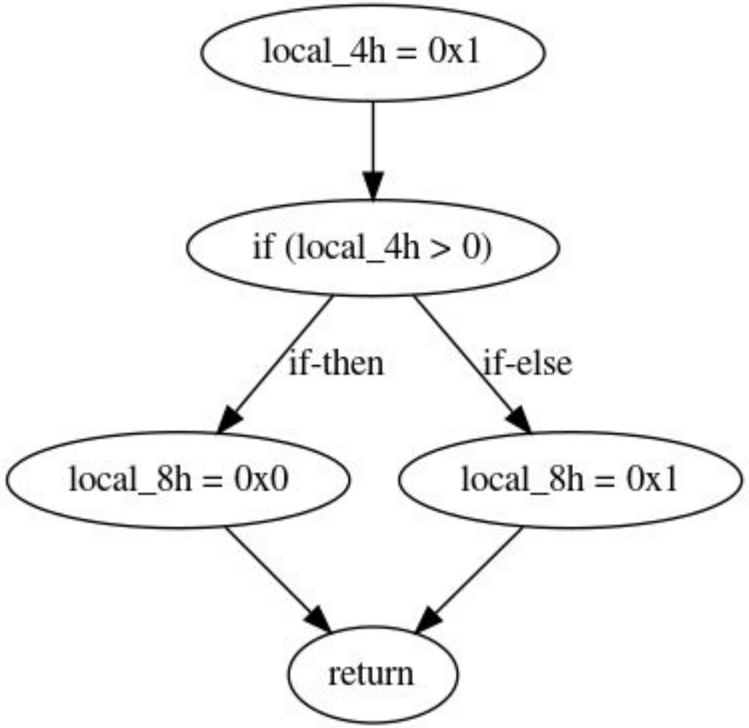


```
    [@0x00062B.0000] %25: $Unknown64(*?) = %14 - #xc;
    [@0x00062B.0002] %26: $Unknown0 = Store(%24, %25, #x0);
    JMP 0x00063E.0000
bb_0x000634.0000(sz 0x6):
    [@0x000634.0001] %27: $Unknown64(*?) = %14 - #xc;
    [@0x000634.0004] %28: $Unknown32(*?) = Load(%29, %27);
    [@0x000634.0008] %30: $Unknown64(*?) = ZeroExt64(%28);
    [@0x000634.0009] %31: $Unknown64(*?) = #xffffffff00000000 & %32;
    [@0x000634.000A] %33: $Unknown64(*?) = %30 | %31;
    [@0x000634.000C] %34: $Unknown64(*?) = %33 & #xffffffff;
    [@0x000637.0000] %35: $Unknown64(*?) = %14 - #x10;
    [@0x000637.0002] %36: $Unknown32(*?) = Load(%29, %35);
```

Example IR

C-like CFG

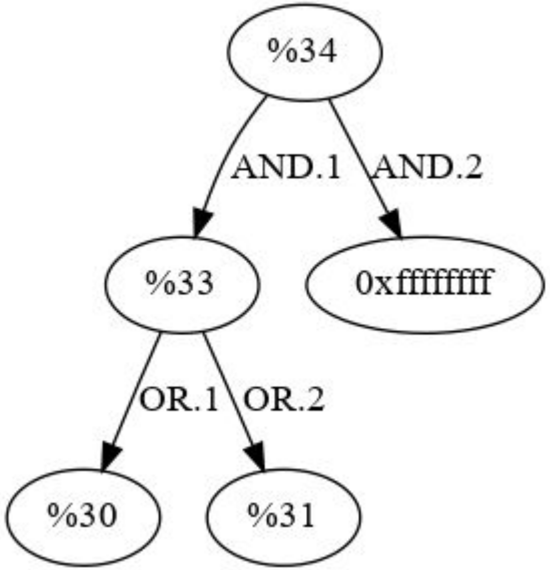
- CFG with C expressions



IR to C-like CFG

- CFG is copied from the one of IR No for/if/while
- Expressions are recovered by expr tree

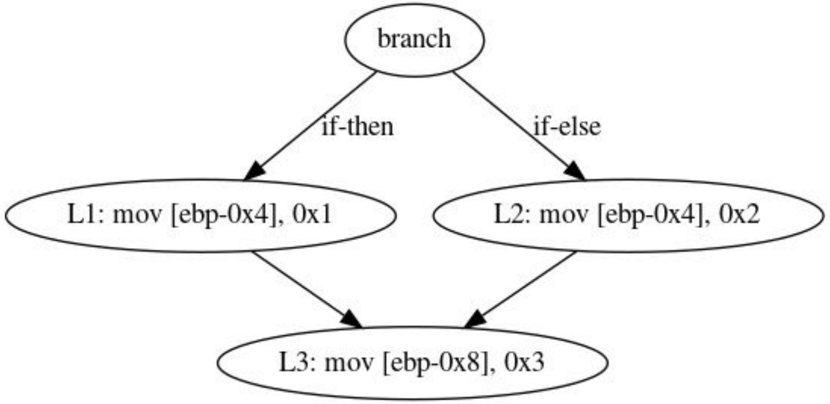
%34 = (%30 | %31) & 0xffffffff

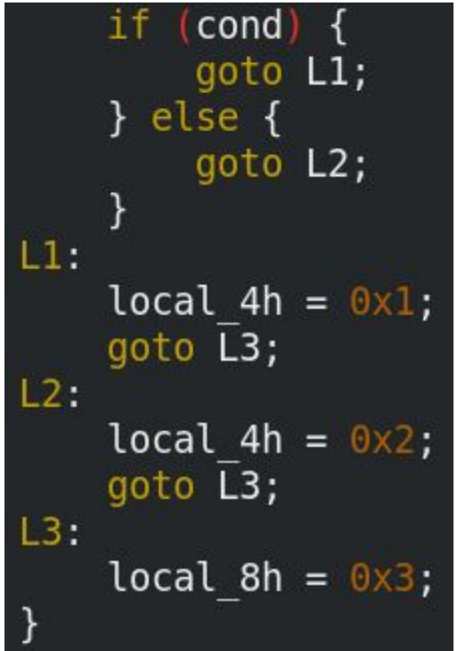


```
%33 = %30 | %31
%34 = %33 & 0xffffffff
```

C-like CFG to Pseudo C code

- Recover C code from CFG (with GOTOs)
- Only recover assignments with memory reference
  - Recovered: mov [ebp - 0x1c], 0x10
  - Ignored: mov eax, 0x10





```
    if (cond) {
        goto L1;
    } else {
        goto L2;
    }
L1:
    local_4h = 0x1;
    goto L3;
L2:
    local_4h = 0x2;
    goto L3;
L3:
    local_8h = 0x3;
}
```

Challenges

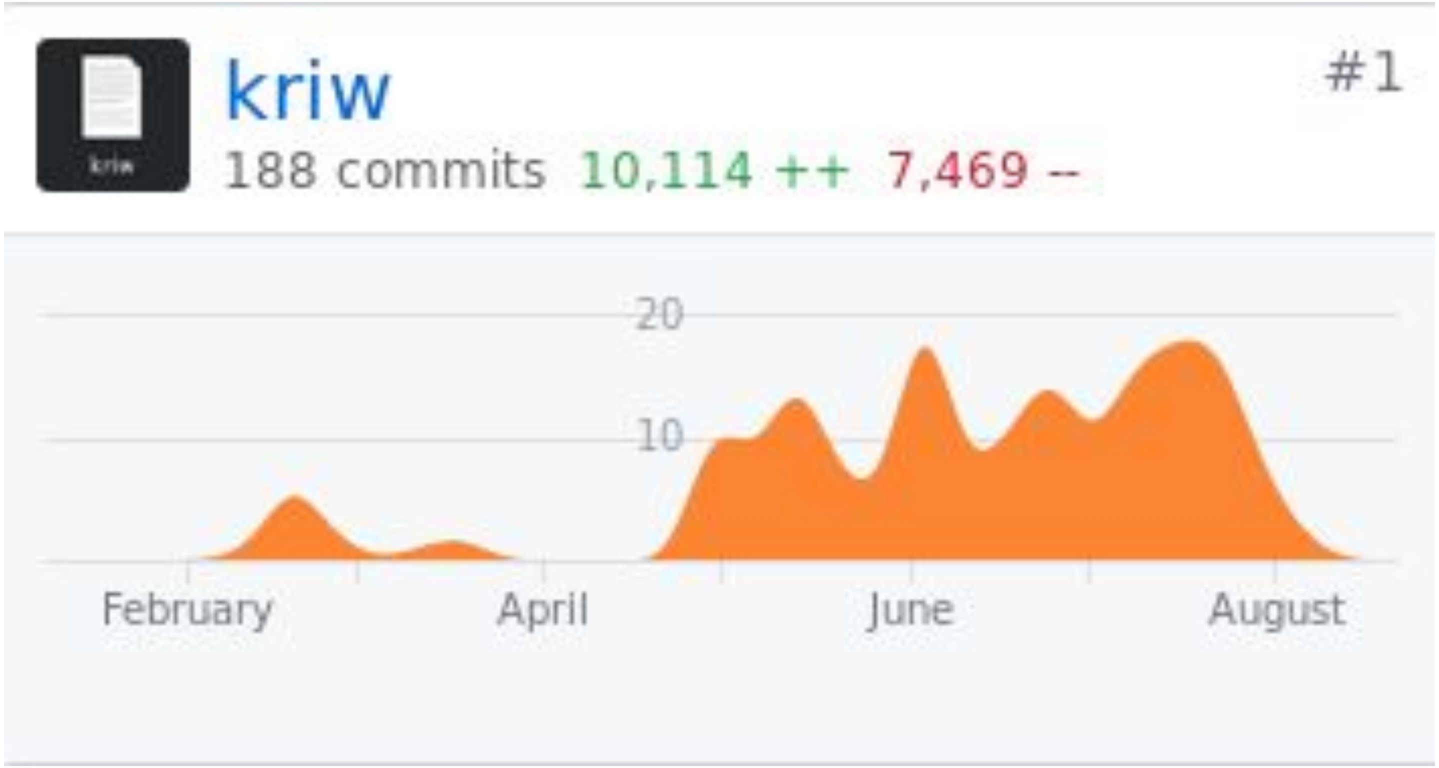
- Required knowledge of program/binary analysis
- Sharing tasks with HMPerson1
- Few information about decompiler

TODOs

Milestone for Radeco-0.1

- Bug fixes
- API stabilization (radeco-lib)
- Documentation
- etc

Commits



kriw

188 commits 10,114 ++ 7,469 --

#1

Thank you