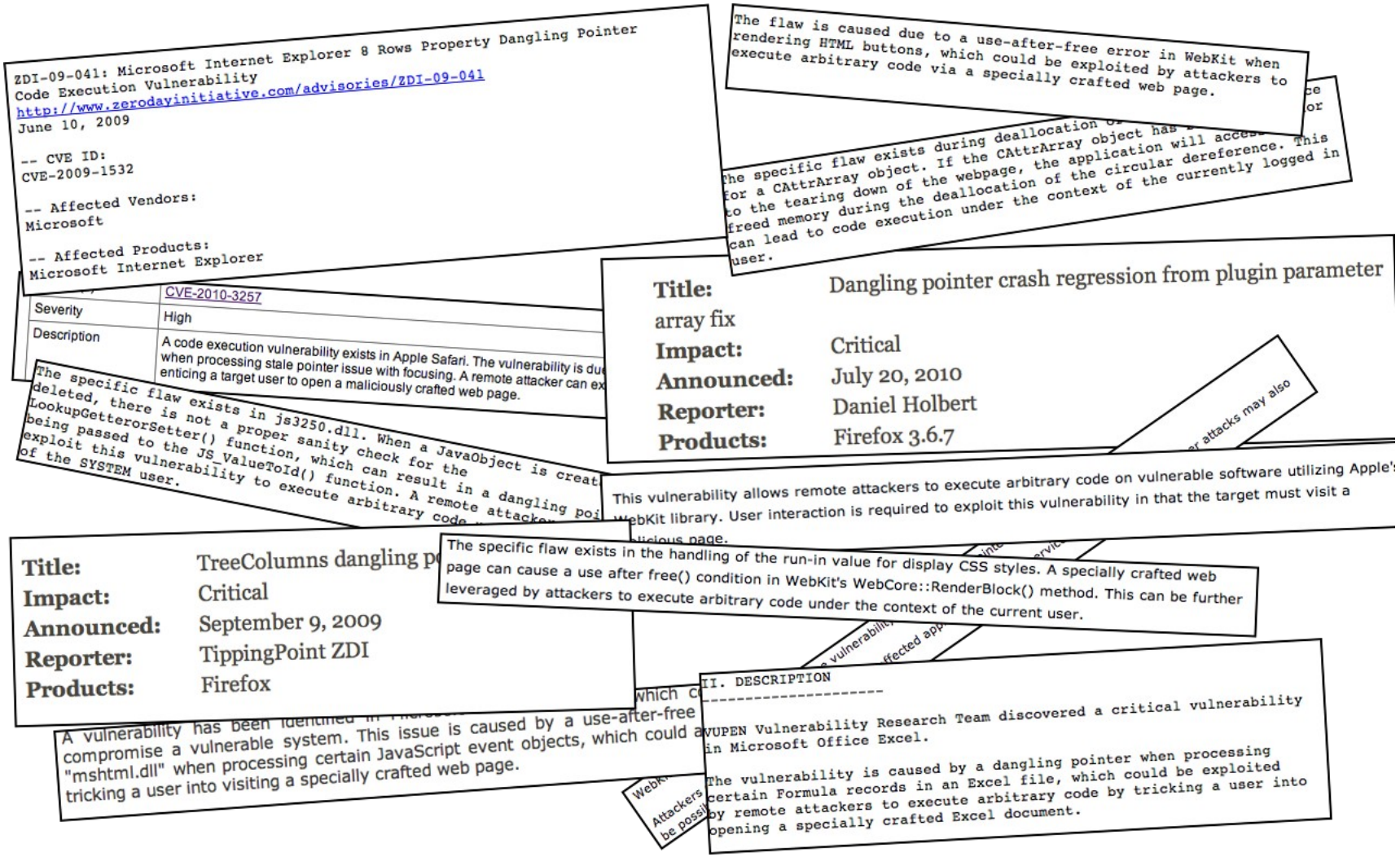
Stale pointers are the new black

Vincenzo Iozzo, Giovanni Gola
vincenzo.iozzo@zynamics.com
giovanni.gola@mail.polimi.it

Stale pointers are the new black



Motivations

@0xcharlie
Charlie Miller

But srsly, how do you compete with @taviso? He reports all my Flash and Reader 0-days. Senseless slaughter of bugs, I'm quitting infosec.

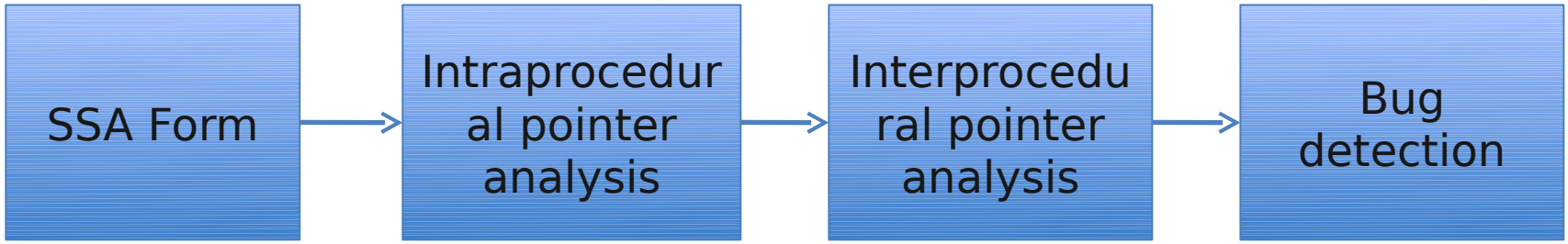
17 Aug via web ☆ Favorite ↻ Retweet ↩ Reply

Different approaches then..

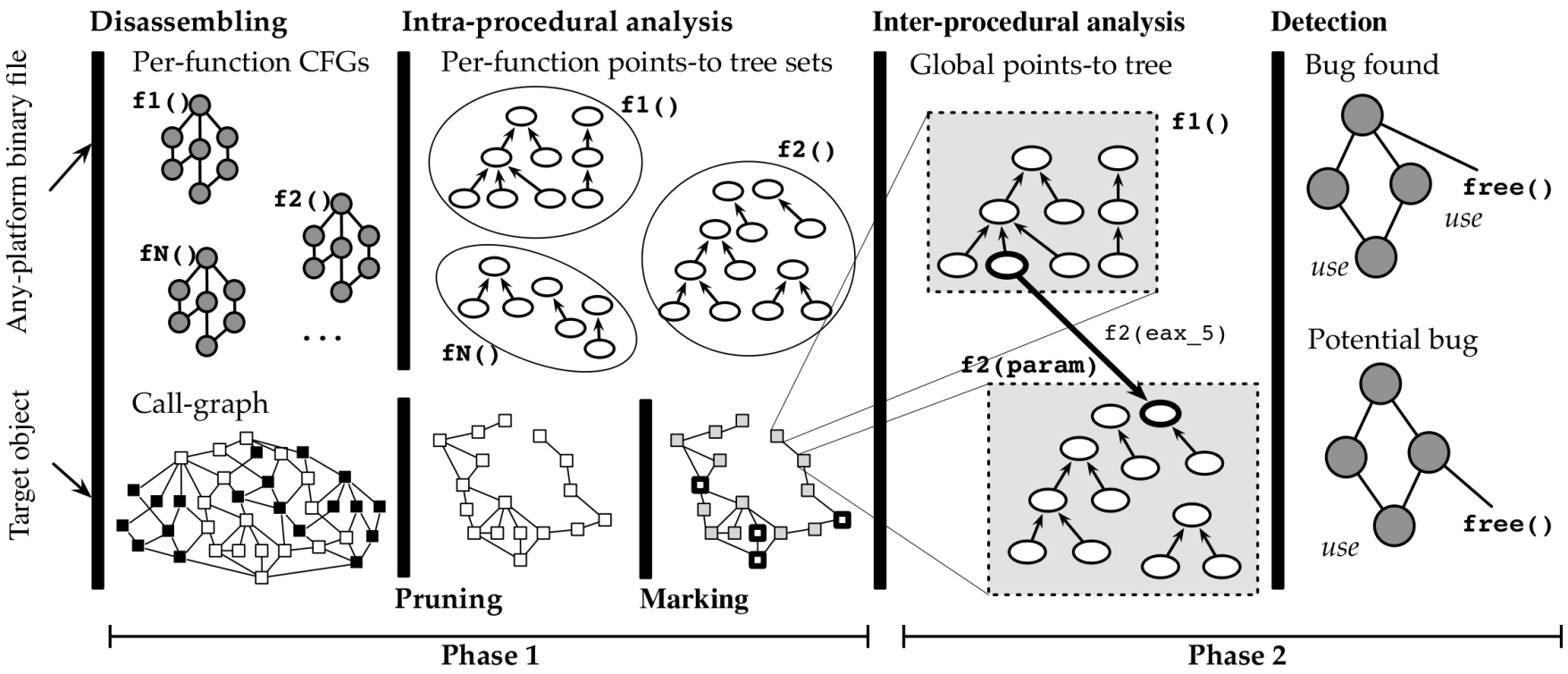
.. Or Static Analysis!

- Abstract interpretation
  - Data-flow analysis
    - **Pointer analysis**
    - Shape analysis
- Model Checking
- Theorem Proving

Our Idea


SSA Form
Intraprocedural pointer analysis
Interprocedural pointer analysis
Bug detection

How it works


Disassembling
Intra-procedural analysis
Inter-procedural analysis
Detection
Any-platform binary file
Target object
Per-function CFGs
f1()
f2()
fN()
...
Call-graph
Per-function points-to tree sets
f1()
f2()
fN()
Global points-to tree
f1()
f2(eax_5)
f2(param)
Bug found
free()
use
use
Potential bug
free()
use
Pruning
Marking
Phase 1
Phase 2

Intermediate language interlude

Enter REIL

- Small RISC instruction set (17 instructions)
  - Arithmetic: ADD, SUB, MUL, DIV, MOD, BSH
  - Bitwise: AND, OR, XOR
  - Conditional: BISZ, JCC
  - Data transfer: LDM, STM, STR
  - Others: NOP, UNDEF, UNKN
- Register machine
- Unlimited number of temp registers
- Side effect free
- No exceptions, floating point, 64Bit, ..

Example: assembly x86 → REIL

```
00001EB0   WebCore.idb::__ZNK7WebCore16AbstractDatabase14securityOriginEv
00001EB0   push        ebp
00001EB1   mov         ebp, esp
00001EB3   mov         eax, ss:[ebp+arg_0]
00001EB6   mov         eax, ds:[eax+0xC]
00001EB9   leave
00001EBA   retn
```

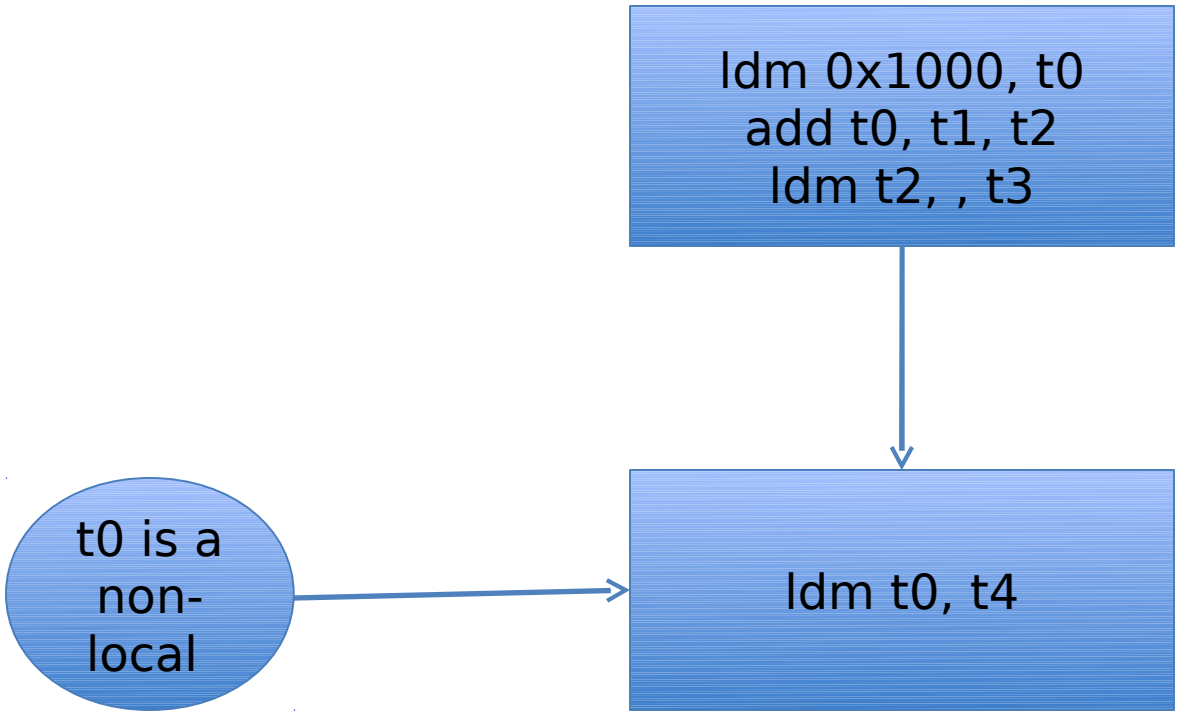
```
001EB000   sub         esp, 0x4, qword t0              // 00001EB0 push ebp
001EB001   and         qword t0, 0xFFFFFFFF, esp
001EB002   stm         ebp,  , esp
001EB100   str         esp,  , ebp                     // 00001EB1 mov ebp, esp
001EB300   add         0x8, ebp, qword t0              // 00001EB3 mov eax, ss: [ebp + arg_0]
001EB301   and         qword t0, 0xFFFFFFFF, t1
001EB302   ldm         t1,  , t2
001EB303   str         t2,  , eax
001EB600   add         0xC, eax, qword t0              // 00001EB6 mov eax, ds: [eax + 12]
001EB601   and         qword t0, 0xFFFFFFFF, t1
001EB602   ldm         t1,  , t2
001EB603   str         t2,  , eax
001EB900   str         ebp,  , esp                     // 00001EB9 leave
001EB901   ldm         esp,  , ebp
001EB902   add         esp, 0x4, qword t0
001EB903   and         qword t0, 0xFFFFFFFF, esp
001EBA00   ldm         esp,  , t0                      // 00001EBA retn
001EBA01   add         esp, 0x4, qword t1
001EBA02   and         qword t1, qword 0xFFFFFFFF, esp
001EBA03   jcc         0x1,  , t0
```

Single Static Assignment Form

Flavours

- Non-pruned
- Semi-pruned
- Pruned

Non-locals



Algorithm

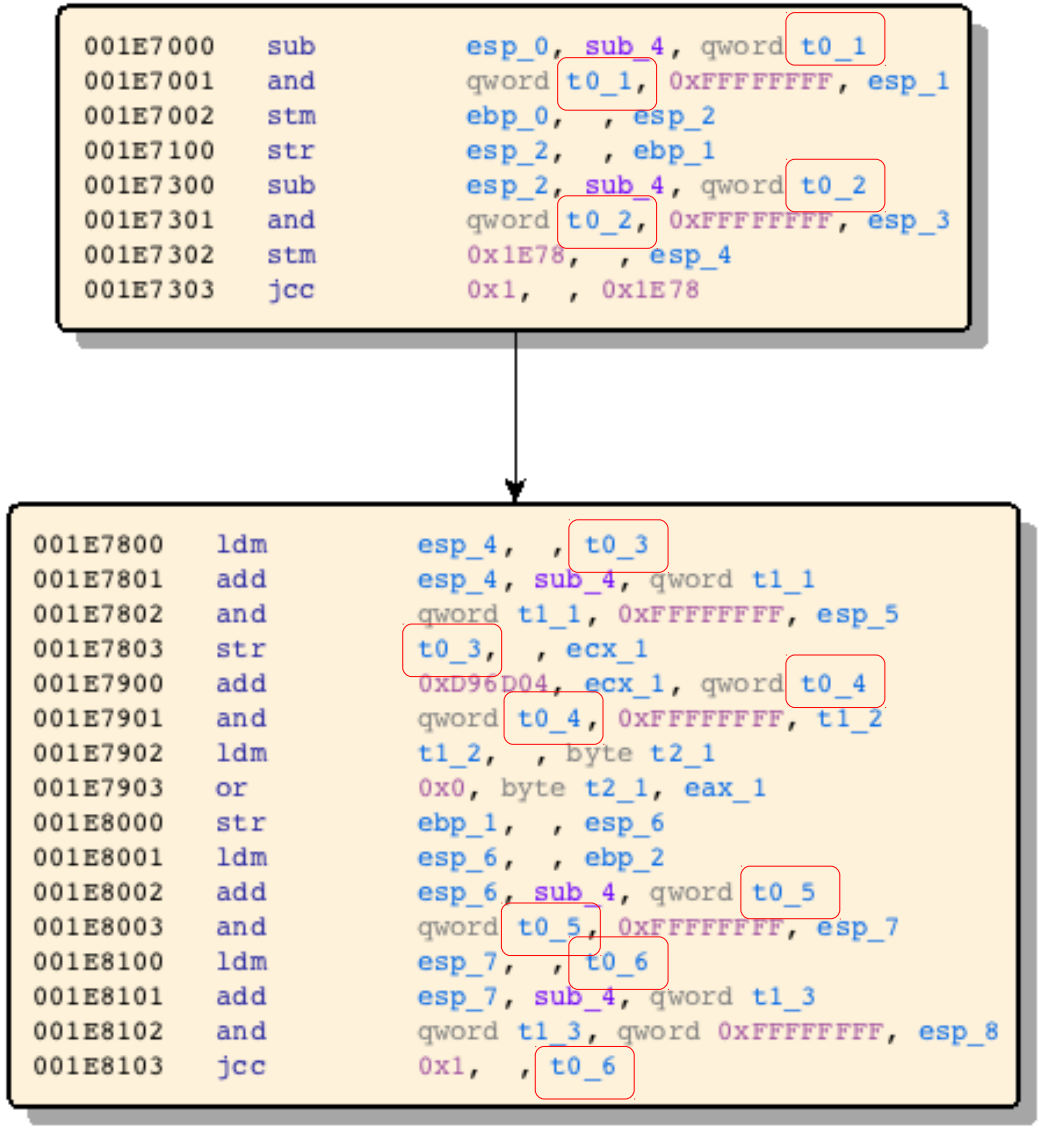
- Find non-locals
- Place phi-functions
- Recursively rename variables

A function

```
001E7000   sub        esp, 0x4, qword t0          // 00001E70 push ebp
001E7001   and        qword t0, 0xFFFFFFFF, esp
001E7002   stm        ebp,  , esp
001E7100   str        esp,  , ebp                 // 00001E71 mov ebp, esp
001E7300   sub        esp, 0x4, qword t0          // 00001E73 call cs: 7800
001E7301   and        qword t0, 0xFFFFFFFF, esp
001E7302   stm        0x1E78,  , esp
001E7303   jcc        0x1,  , 0x1E78
```

```
001E7800   ldm        esp,  , t0                  // 00001E78 pop ecx
001E7801   add        esp, 0x4, qword t1
001E7802   and        qword t1, 0xFFFFFFFF, esp
001E7803   str        t0,  , ecx
001E7900   add        0xD96D04, ecx, qword t0     // 00001E79 movzx eax, byte ds: [ecx + 14249220]
001E7901   and        qword t0, 0xFFFFFFFF, t1
001E7902   ldm        t1,  , byte t2
001E7903   or         0x0, byte t2, eax
001E8000   str        ebp,  , esp                 // 00001E80 leave
001E8001   ldm        esp,  , ebp
001E8002   add        esp, 0x4, qword t0
001E8003   and        qword t0, 0xFFFFFFFF, esp
001E8100   ldm        esp,  , t0                  // 00001E81 retn
001E8101   add        esp, 0x4, qword t1
001E8102   and        qword t1, qword 0xFFFFFFFF, esp
001E8103   jcc        0x1,  , t0
```

In SSA Form

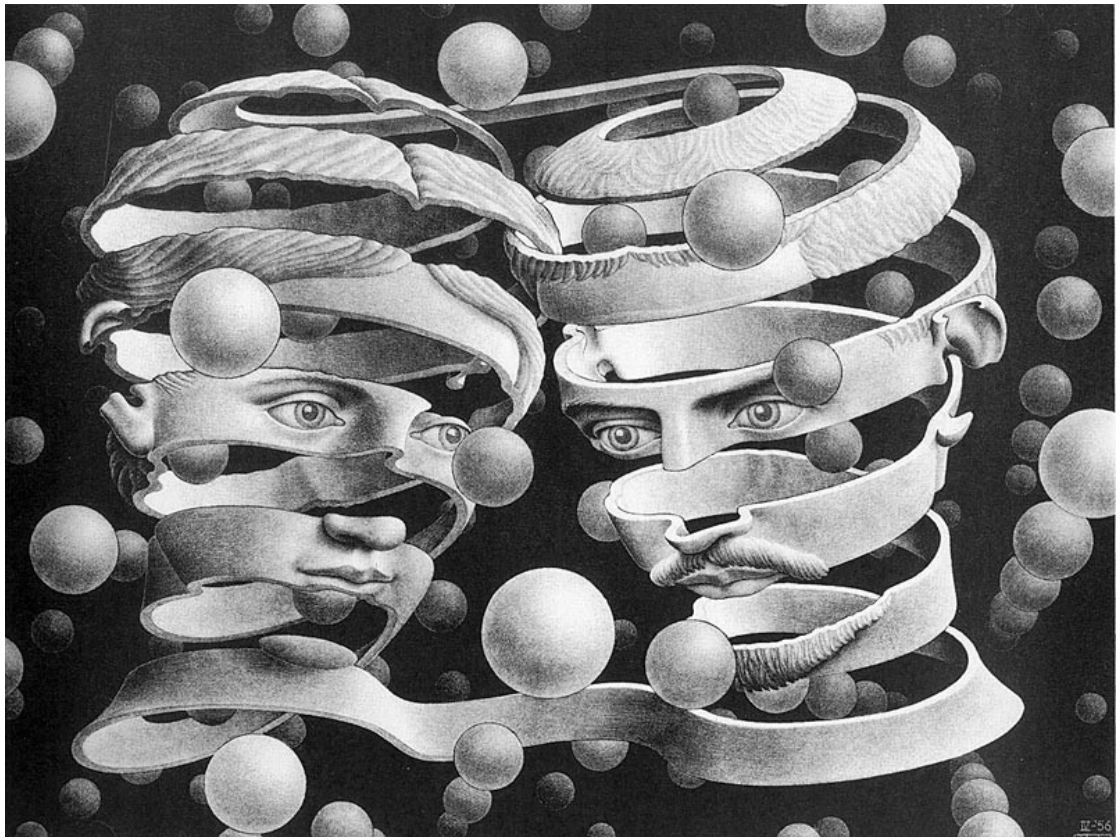

001E7000 sub esp_0, sub_4, qword t0_1
001E7001 and qword t0_1, 0xFFFFFFFF, esp_1
001E7002 stm ebp_0, , esp_2
001E7100 str esp_2, , ebp_1
001E7300 sub esp_2, sub_4, qword t0_2
001E7301 and qword t0_2, 0xFFFFFFFF, esp_3
001E7302 stm 0x1E78, , esp_4
001E7303 jcc 0x1, , 0x1E78
001E7800 ldm esp_4, , t0_3
001E7801 add esp_4, sub_4, qword t1_1
001E7802 and qword t1_1, 0xFFFFFFFF, esp_5
001E7803 str t0_3, , ecx_1
001E7900 add 0xD96D04, ecx_1, qword t0_4
001E7901 and qword t0_4, 0xFFFFFFFF, t1_2
001E7902 ldm t1_2, , byte t2_1
001E7903 or 0x0, byte t2_1, eax_1
001E8000 str ebp_1, , esp_6
001E8001 ldm esp_6, , ebp_2
001E8002 add esp_6, sub_4, qword t0_5
001E8003 and qword t0_5, 0xFFFFFFFF, esp_7
001E8100 ldm esp_7, , t0_6
001E8101 add esp_7, sub_4, qword t1_3
001E8102 and qword t1_3, qword 0xFFFFFFFF, esp_8
001E8103 jcc 0x1, , t0_6

Φ function


00321800 stm esi_18, , esp_152
00321B00 sub esp_152, sub_4, qword t0_210
00321B01 and qword t0_210, 0xFFFFFFFF, esp_153
00321B02 stm 0x3220, , esp_154
00321B03 jcc 0x1, , __ZN3WTF8fastFreeEPv
00313800 add 0xFFFFFFB4, ebp_1, qword t0_189
00313801 and qword t0_189, 0xFFFFFFFF, t1_181
00313802 ldm t1_181, , t2_53
00313803 str t2_53, , eax_44
00313B00 and eax_44, eax_44, t0_190
00313B01 and t0_190, 0x80000000, t1_182
00313B02 bsh t1_182, -31, byte SF_47
00313B03 bisz t0_190, , byte ZF_47
00313B04 str byte 0x0, , byte CF_47
00313B05 str byte 0x0, , byte OF_47
00313D00 jcc byte ZF_47, , 0x315B
00314B00 stm ebx_10, , esp_130
00314E00 sub esp_130, sub_4, qword t0_195
00314E01 and qword t0_195, 0xFFFFFFFF, esp_131
00314E02 stm 0x3153, , esp_132
00314E03 jcc 0x1, , __ZN3WTF10StringImplD1Ev
00315300 stm ebx_10, , esp_133
00315600 sub esp_133, sub_4, qword t0_196
00315601 and qword t0_196, 0xFFFFFFFF, esp_134
00315602 stm 0x315B, , esp_135
00315603 jcc 0x1, , __ZN3WTF8fastFreeEPv
00315B00 str RETURN_PHI, , esp_136
00315B01 sub esp_136, sub_4, qword t0_197
00315B02 and qword t0_197, 0xFFFFFFFF, esp_137
00315B03 stm 0x3160, , esp_138
00315B04 jcc 0x1, , 0x176CC0

Detour...Abstract interpretation

Abstract Interpretation



Abstract Interpretation..formally

Give several semantics linked by relations of abstraction

Abstract Interpretation..intuitively

Partial execution of the program which tracks only part of information about its semantics, without performing all the calculations

MonoREIL


monorail kitteh
experiencing sum technukul difukultiez
ICANHASCHEEZBURGER.COM

So what you need?

- The control flow graph of a function
- A way to walk the CFG
- The lattice
  - Its elements
  - A way to combine lattice elements
- An initial state
- REIL instructions effects on the lattice

One constraint!

The lattice has to be noetherian

Noether...

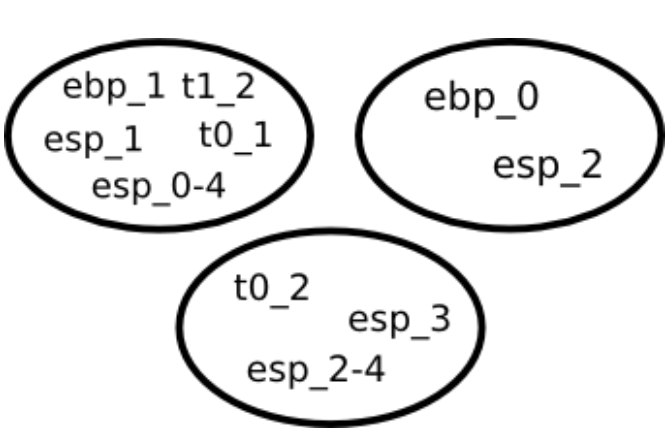
…Emmy Noether

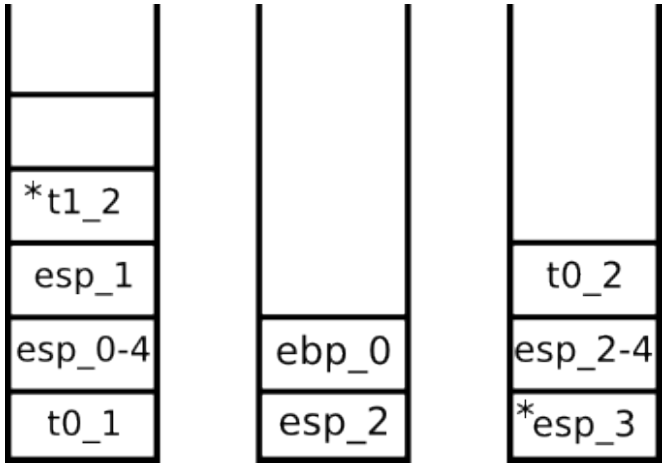
Now the analysis itself

Intraprocedural Analysis

- Pointer Analysis: Efficiency
- Shape Analysis: Precision
- Alias Set Analysis: Tradeoff between the two

Data Structures





- push() and pop() on linked lists

  30% faster

- Hash consing

  30% memory saving

Transfer Functions

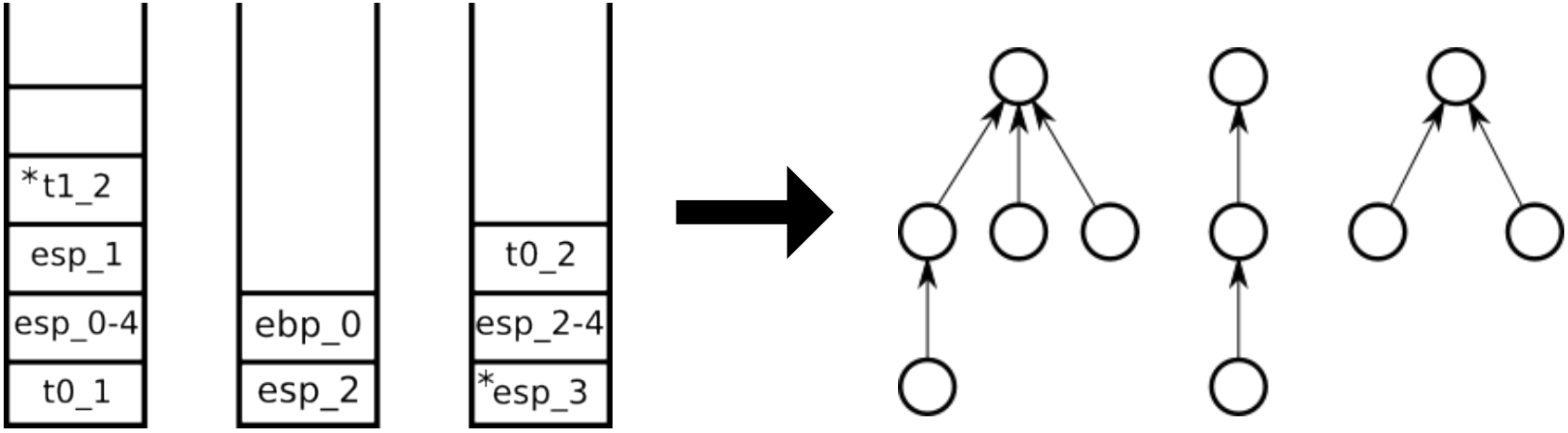
Arithmetic instructions	**Operation**
ADD x_1, x_2, y	y is added to the alias set of $x_1 + x_2$
SUB x_1, x_2, y	y is added to the alias set of $x_1 - x_2$
MUL x_1, x_2, y	y is added to the alias set of $x_1 \cdot x_2$
DIV x_1, x_2, y	y is added to the alias set of $\left\lfloor \frac{x_1}{x_2} \right\rfloor$
MOD x_1, x_2, y	y is added to the alias set of $x_1 \bmod x_2$
BSH x_1, x_2, y	y is added to the alias set of $\begin{cases} x_1 \cdot 2^{x_2} & \text{if } x_2 \geq 0 \\ \left\lfloor \frac{x_1}{2^{-x_2}} \right\rfloor & \text{if } x_2 < 0 \end{cases}$
Bitwise instructions	**Operation**
AND x_1, x_2, y	y is added to the alias set of $x_1 \& x_2$
OR x_1, x_2, y	y is added to the alias set of $x_1 \mid x_2$
XOR x_1, x_2, y	y is added to the alias set of $x_1 \oplus x_2$
Logical instructions	**Operation**
BISZ x_1, ε, y	y is removed from all alias sets
JCC x_1, ε, y	does not affect alias sets
Data transfer instructions	**Operation**
LDM x_1, ε, y	y is added to the alias set of mem$[x_1]$
STM x_1, ε, y	mem$[y]$ is added to the alias set of x_1
STR x_1, ε, y	y is added to the alias set of x_1
Other instructions	**Operation**
NOP $\varepsilon, \varepsilon, \varepsilon$	does not affect alias sets
UNDEF $\varepsilon, \varepsilon, y$	y is removed from all alias sets
UNKN $\varepsilon, \varepsilon, \varepsilon$	does not affect alias sets

combine()

- Filter out non-live variables from each alias list:
  - live-out(inst) ⊆ vars(dom(inst))
  - Alias list ∩ vars(sdom(Φ)):
    - pop() from the list until

      top(alias list) ∈ vars(sdom(Φ))
- Add aliases defined by Φ functions
- Unite the sets of lists

Data Structures Again



Alias lists

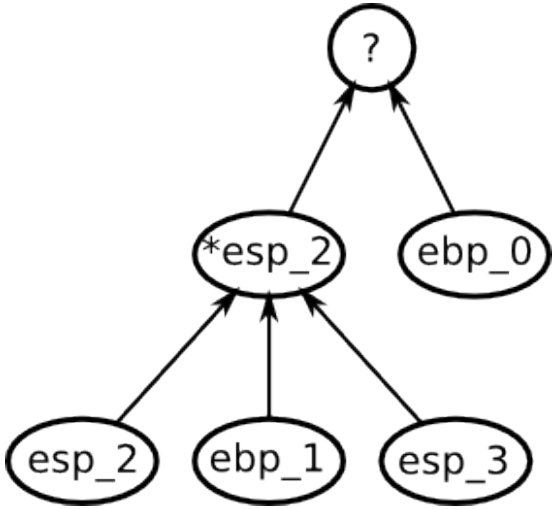
Points-to trees

Example: intraprocedural analysis

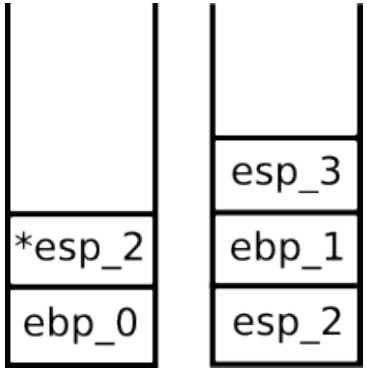
```
001ED002    stm         ebp_0,  , esp_2
001ED100    str         esp_2,  , ebp_1

001ED900    str         ebp_1,  , esp_3
```

REIL in SSA Form



Points-to trees



Alias lists

Data structures and C++ peculiarities

Heap Modeling and Aggregate Modeling:

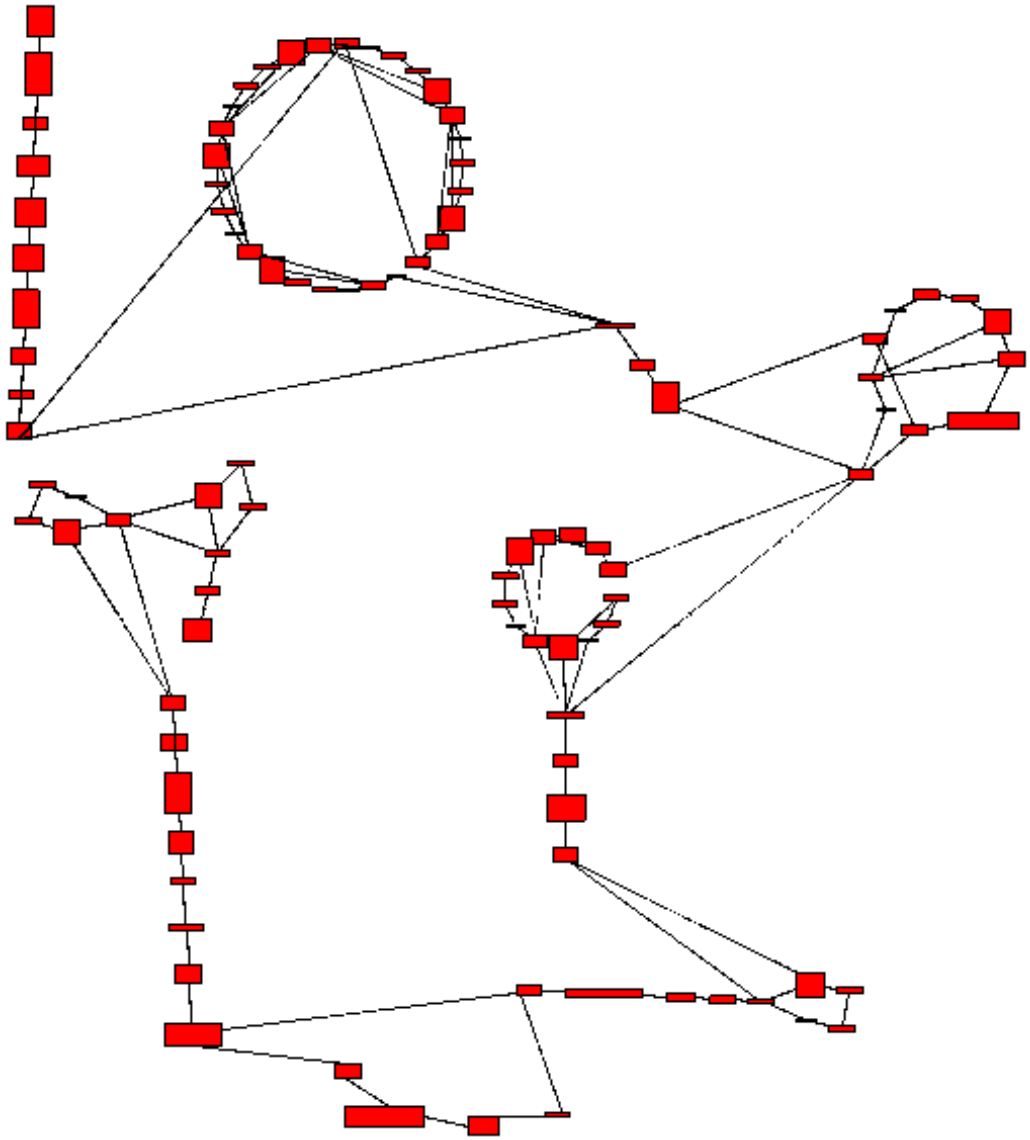
Too slow OR too imprecise

Identify constructors and destructors and consequent Size Tracking

+

Define access to abstract data structures

Interprocedural Analysis



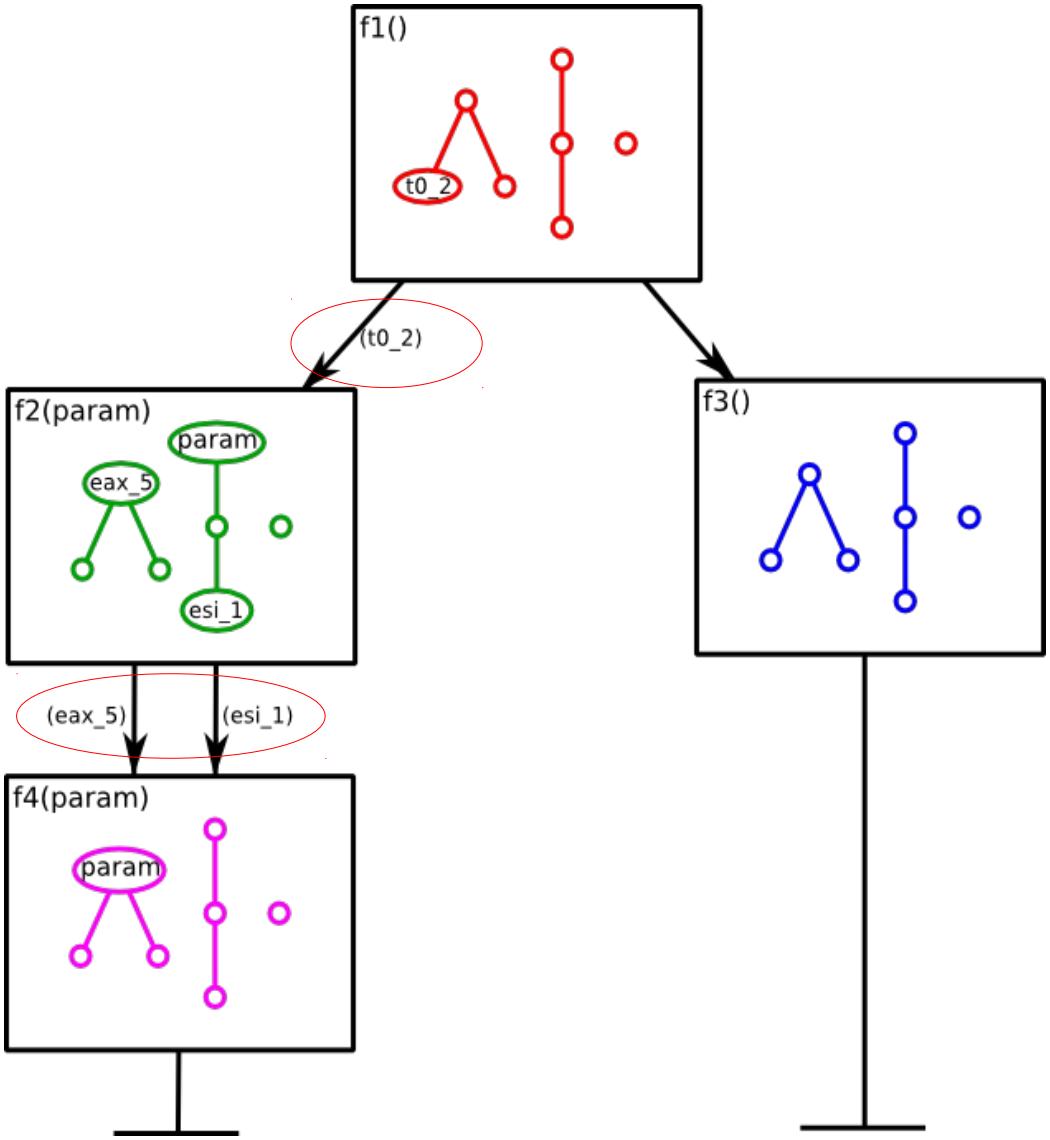
Tracking parameters and return

- IDA effectively tracks parameters
- return is identified by guessing the calling convention

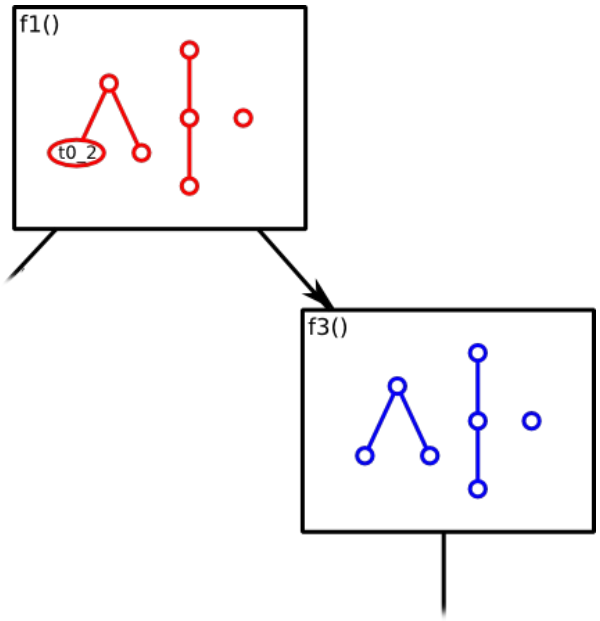
Algorithm

- Flow-insensitive
- Context-sensitive
- Fixed point analysis:
  - walks on the PCG

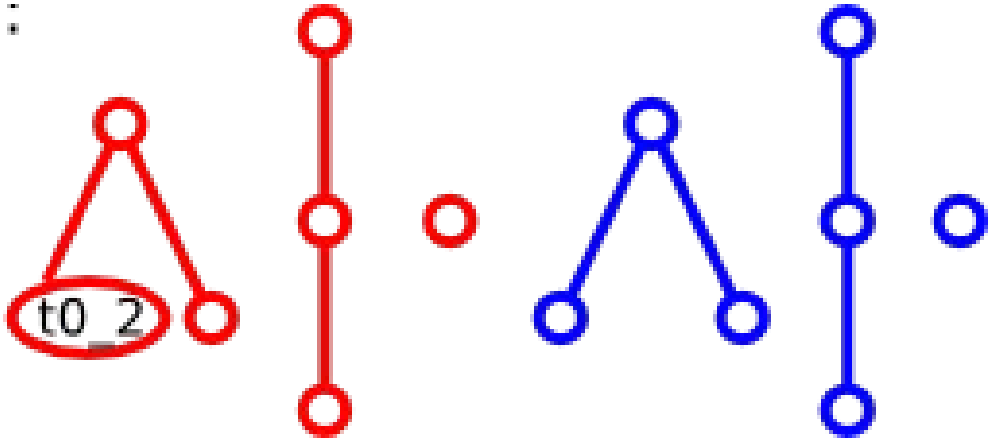
Procedure Call Graph



Example: interprocedural analysis

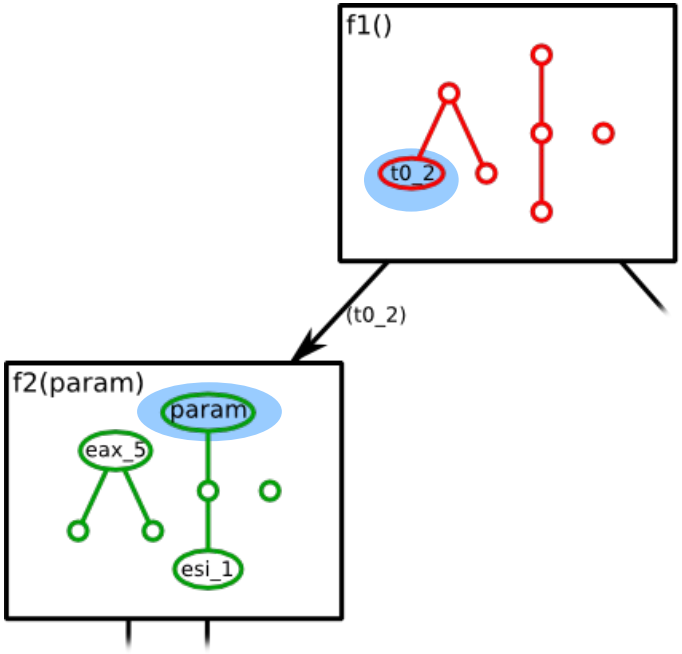


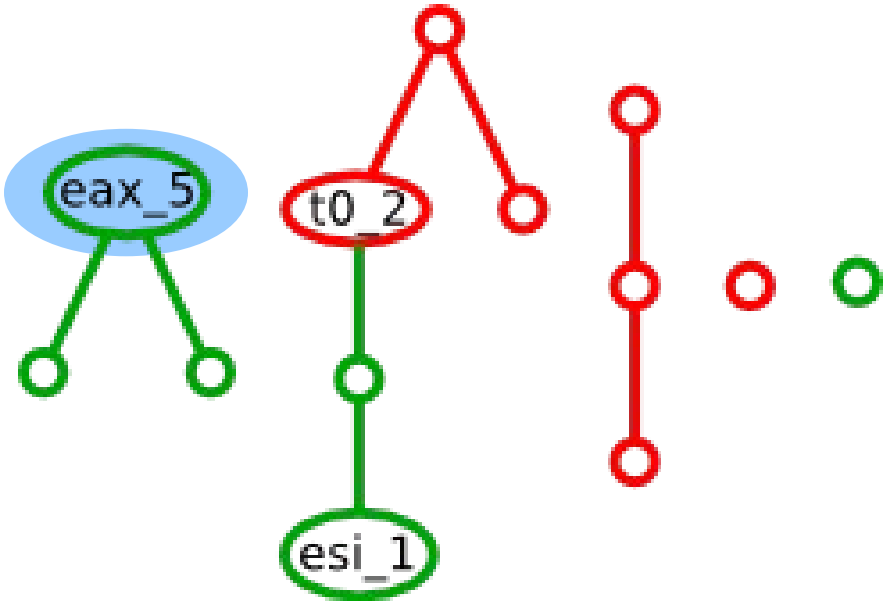
PCG (detail)



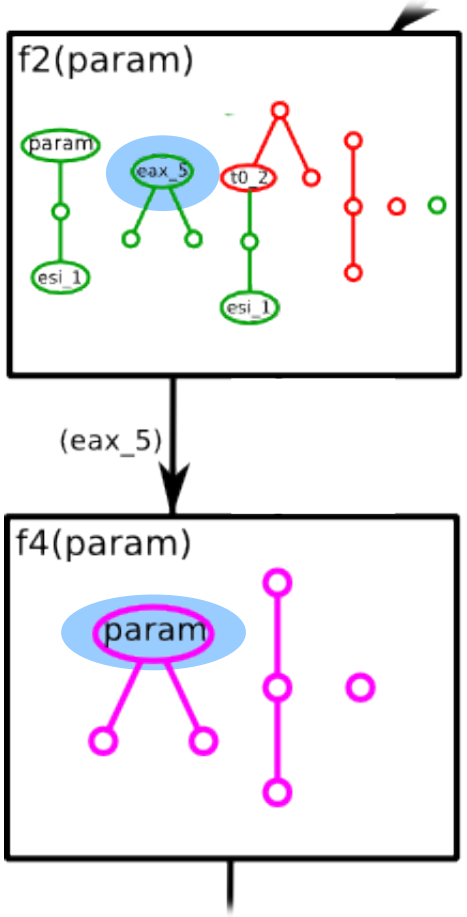
Analysis result

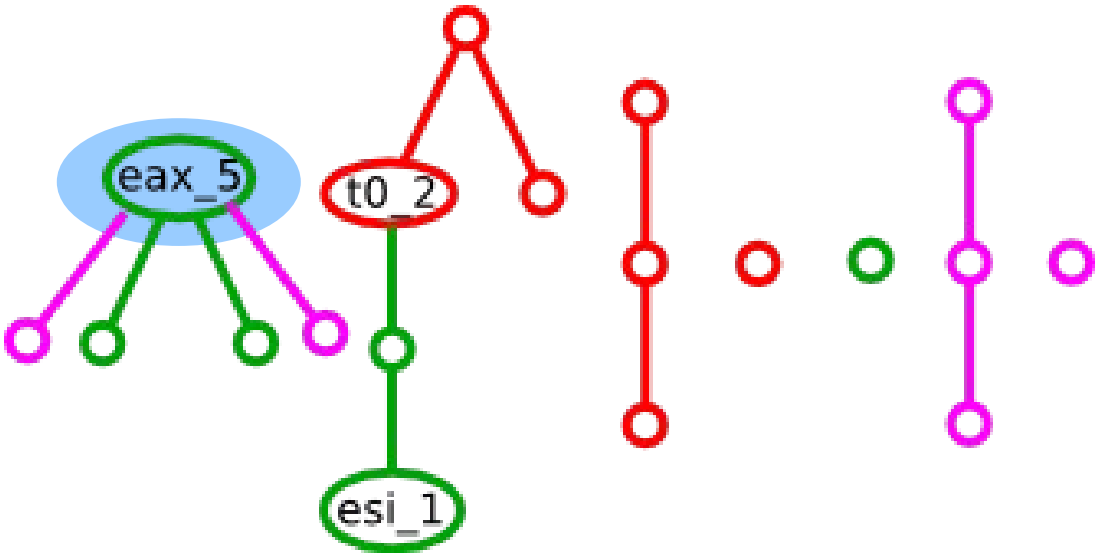
Example: interprocedural analysis



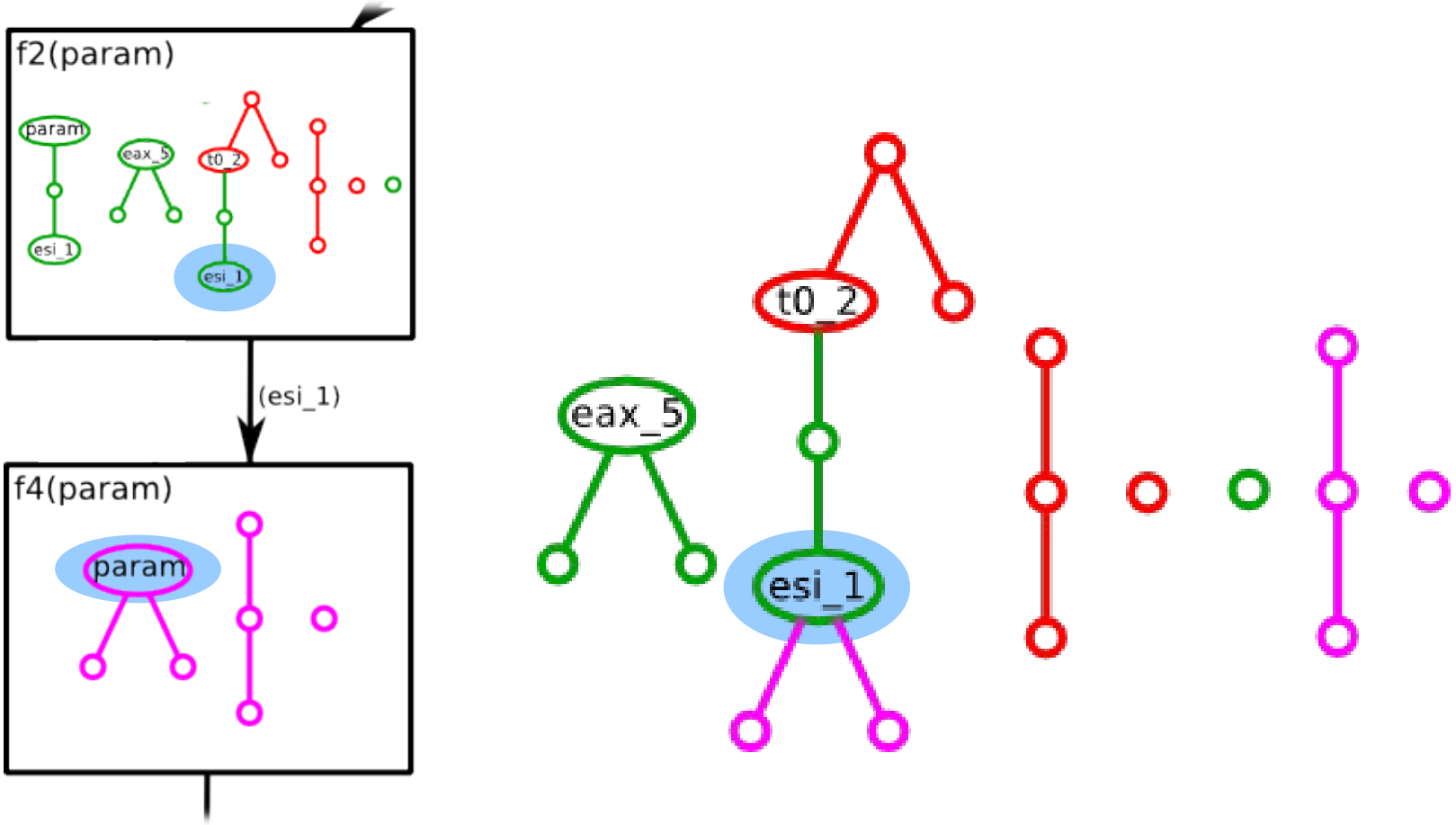


Example: interprocedural analysis

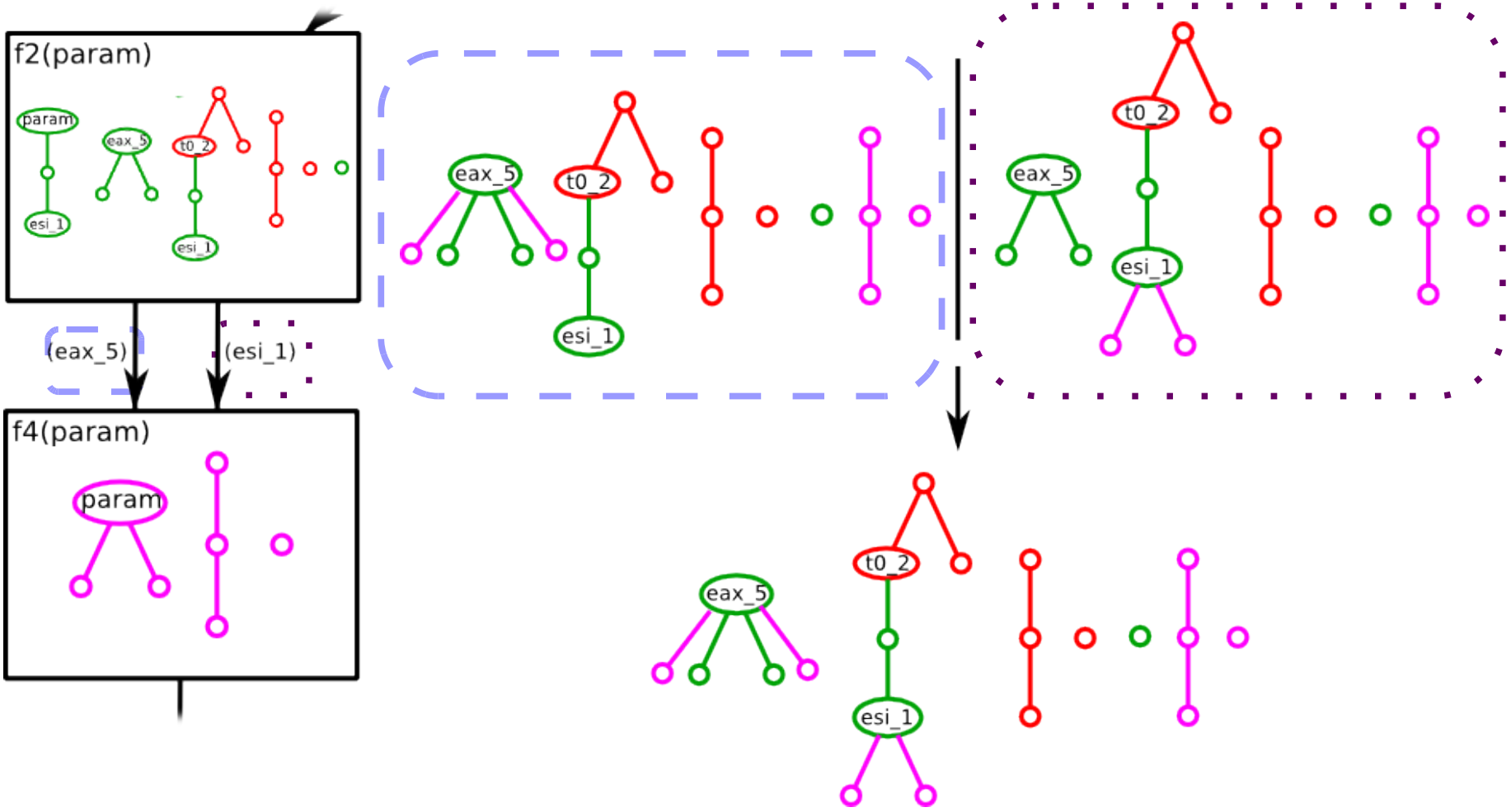




Example: interprocedural analysis

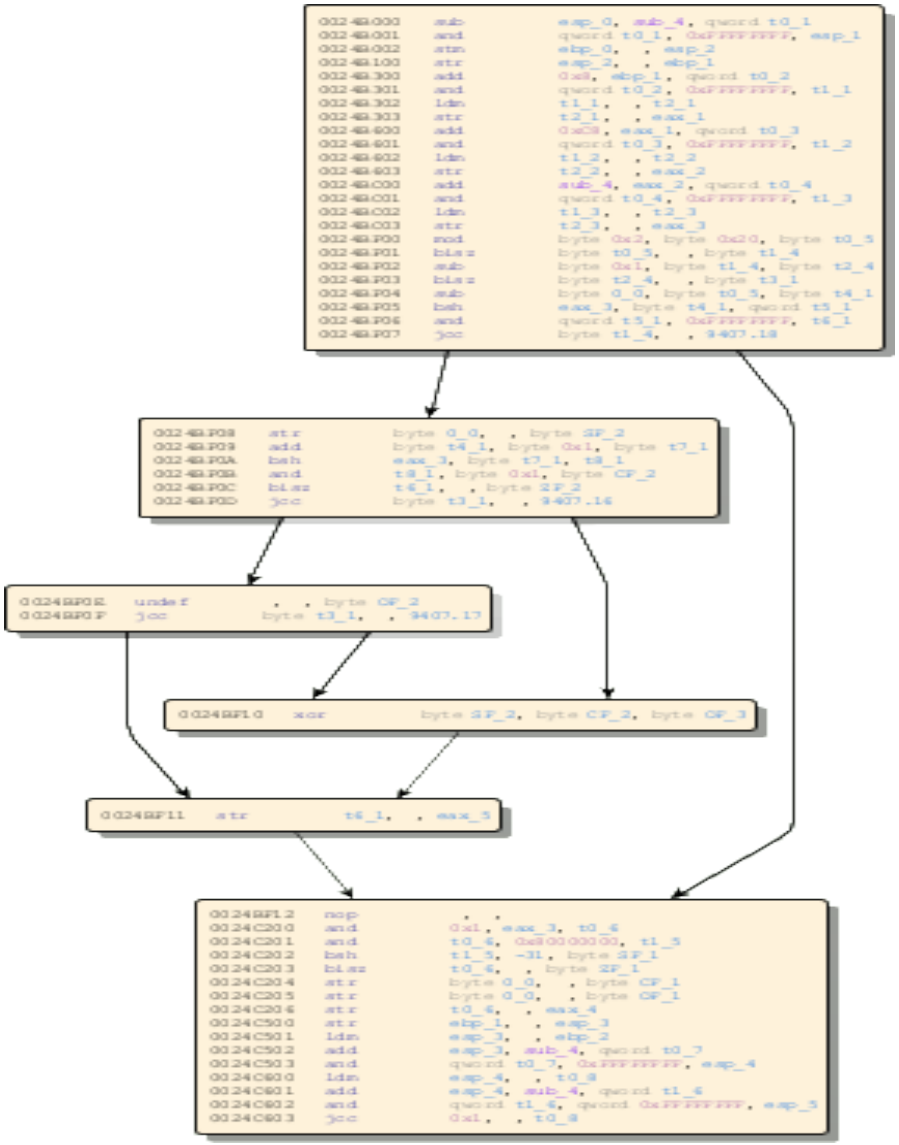


Example: interprocedural analysis

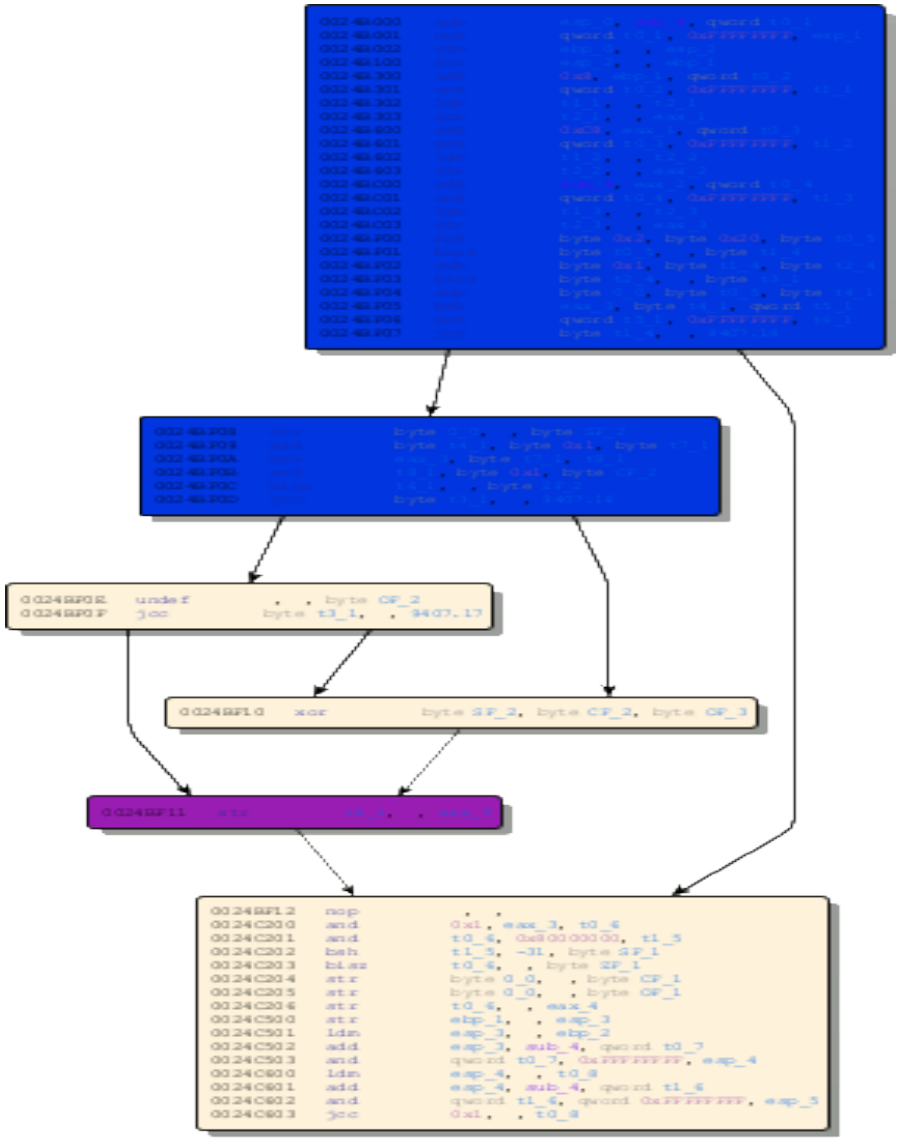


Detour...Graph Theory for Dummies

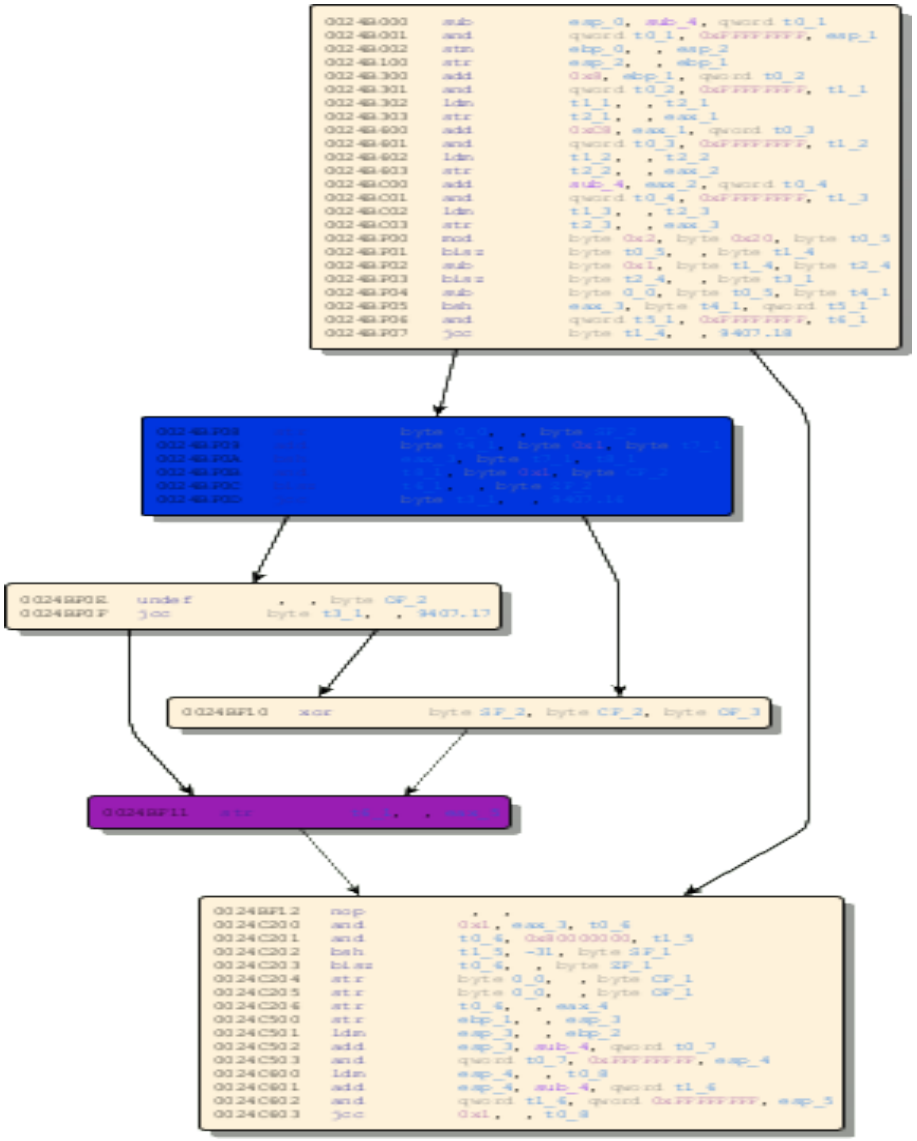
A control flow graph



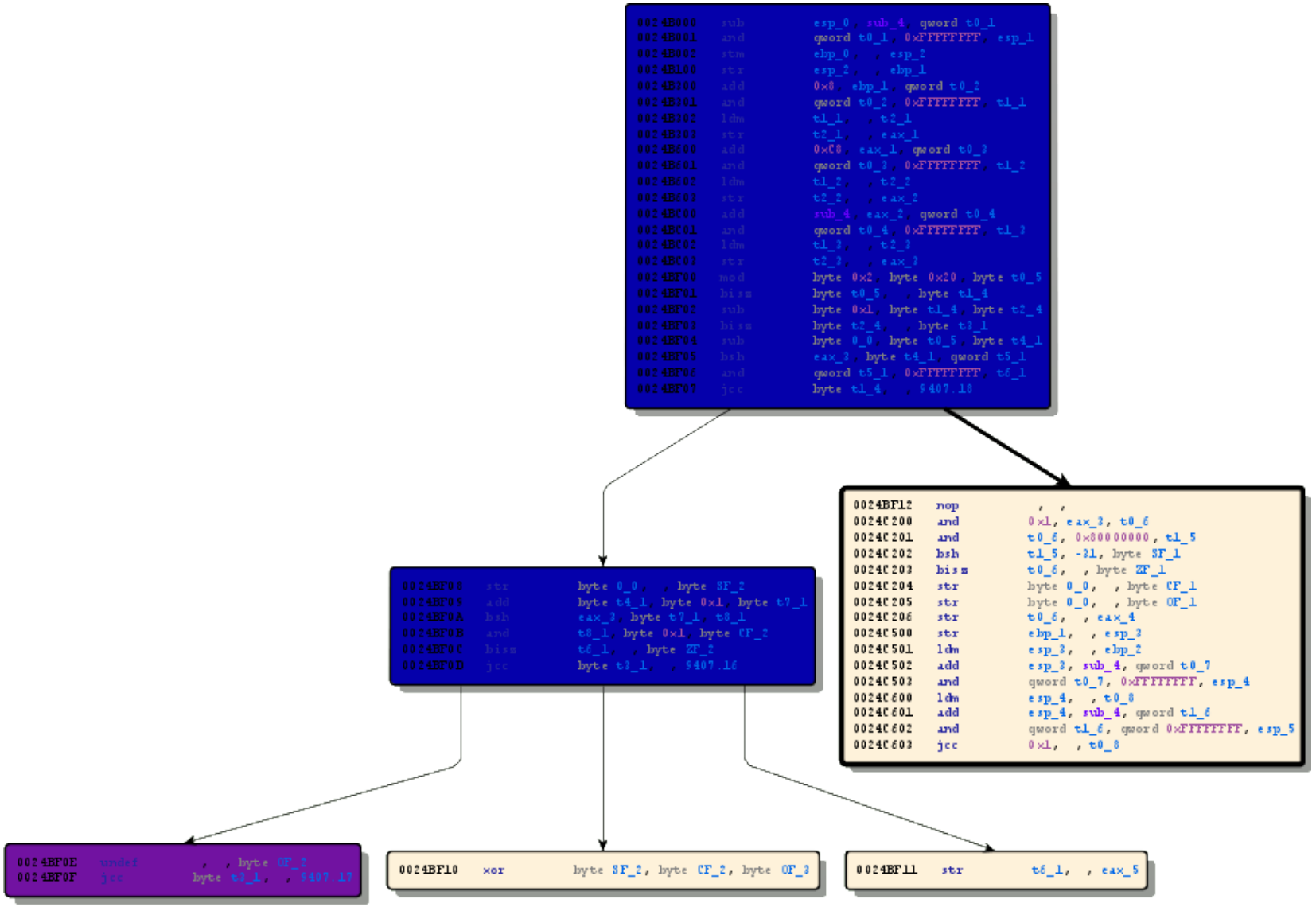
Dominators



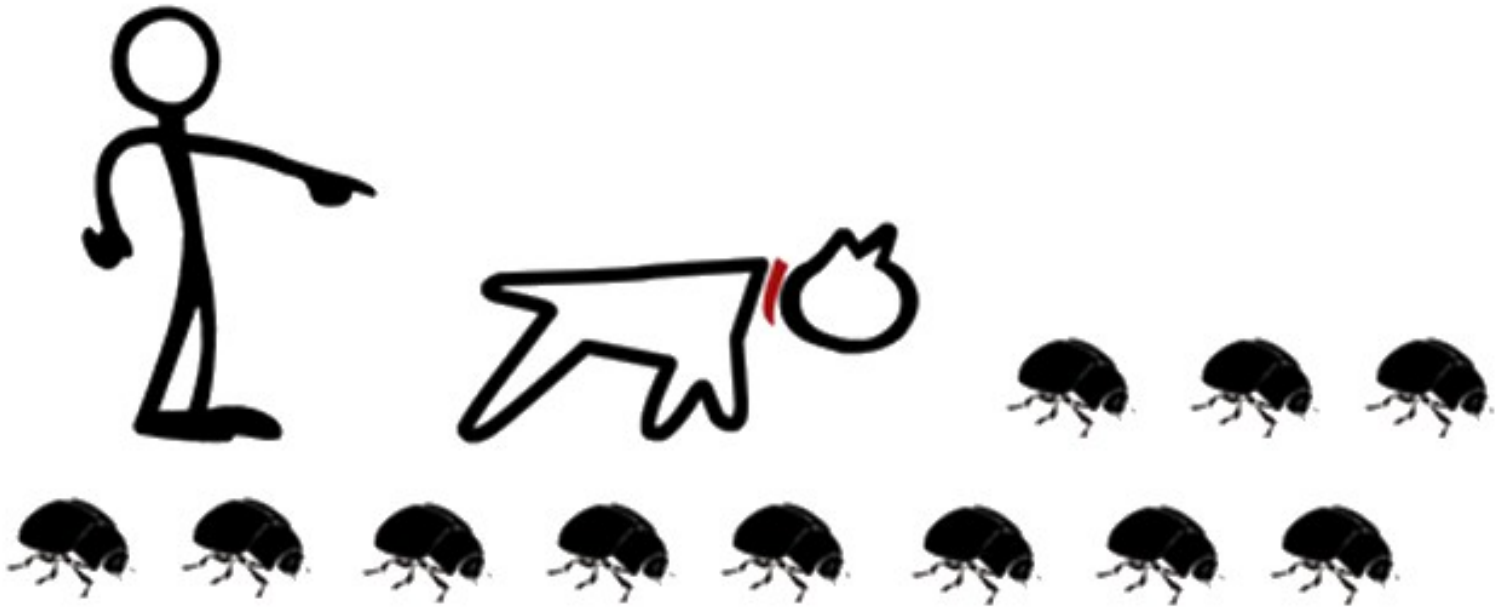
Immediate dominator



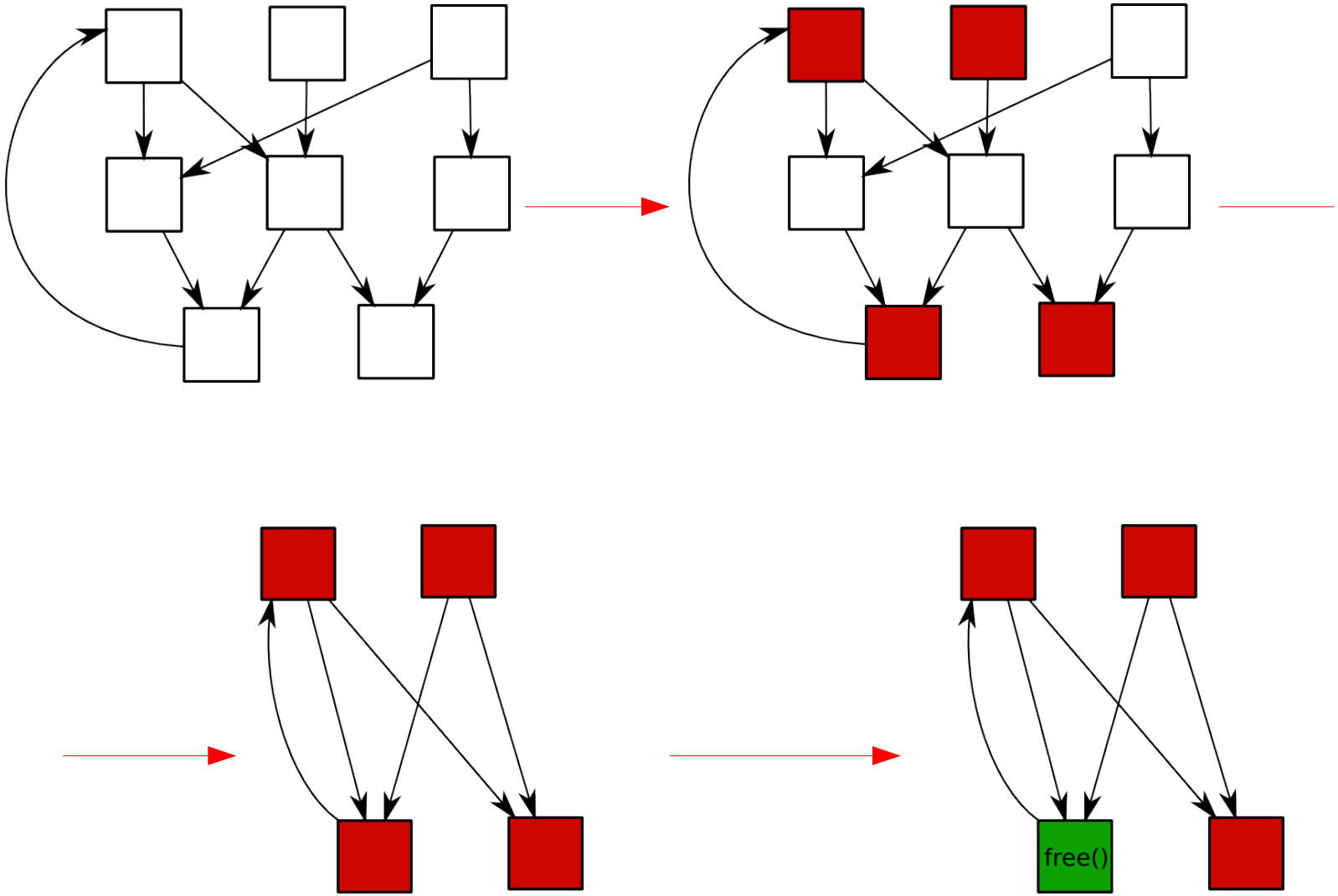
Dominator tree



Bug Detection



Callgraph pruning



Algorithm

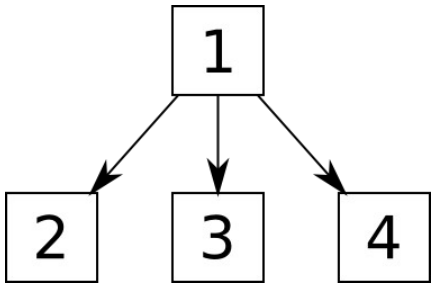
v: tracked alias
X: basic block of f() that calls the destructor
B: basic block of f() that accesses v or calls a function that accesses v

For every couple (X, B) in f(), check the following:

	Bug	*Warning*
Stale pointer	$B \in dom(X)$	$B \notin dom(X) \wedge B \in succ(X)$
Memory leak	$X \notin dom(B) \wedge X \notin succ(B)$	$X \notin dom(B) \wedge X \in succ(B)$

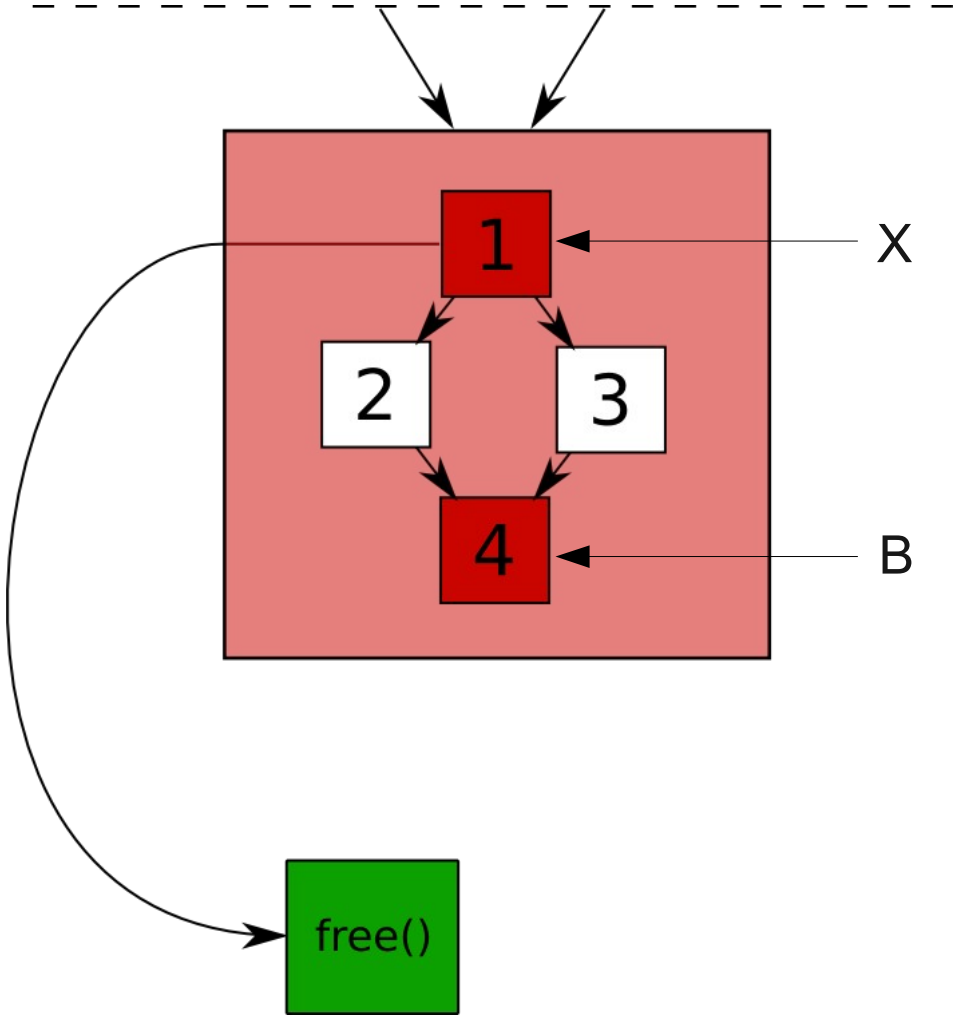
Iterate substituting f() with each of its callers

Example: use after free bug

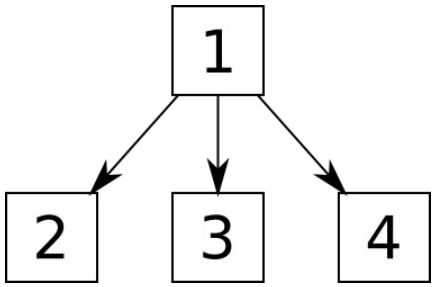


Dominator tree

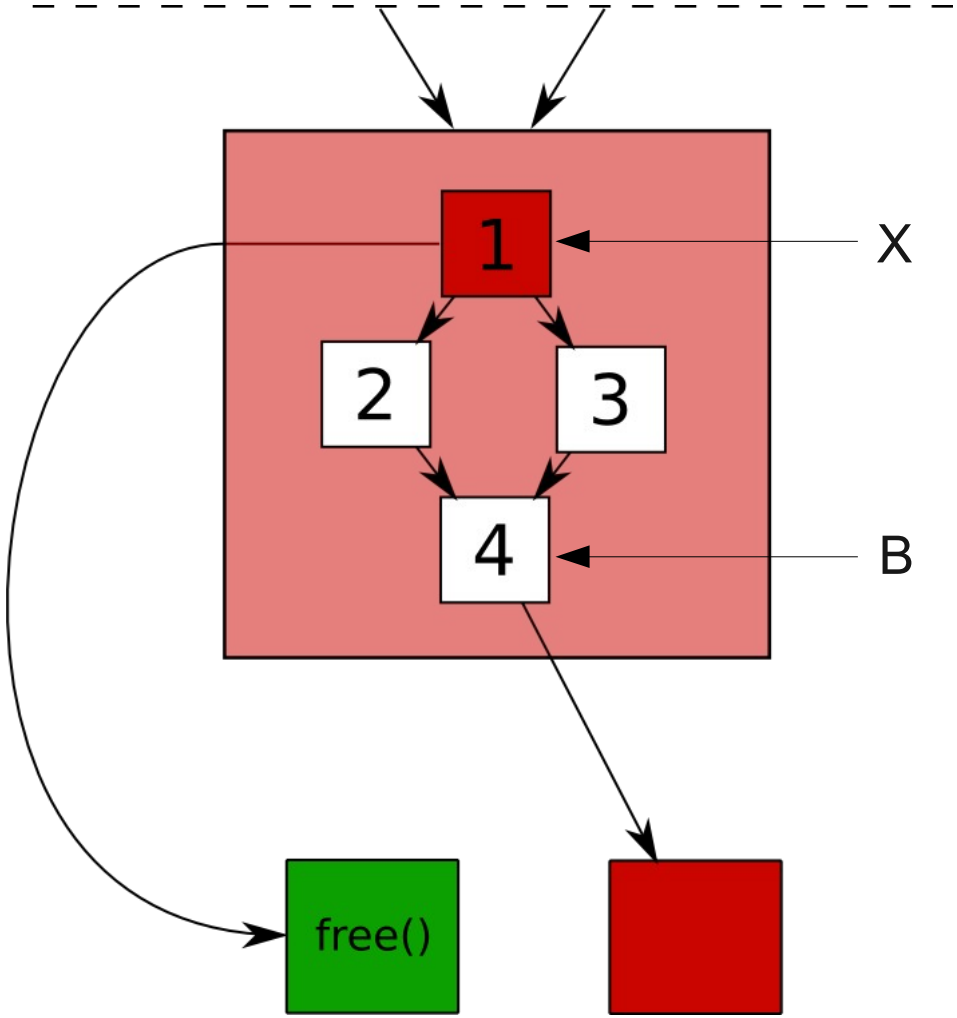
$B \in dom(X)$



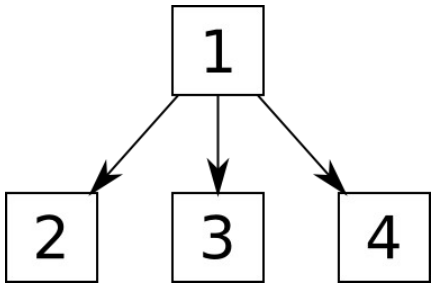
Example: use after free bug

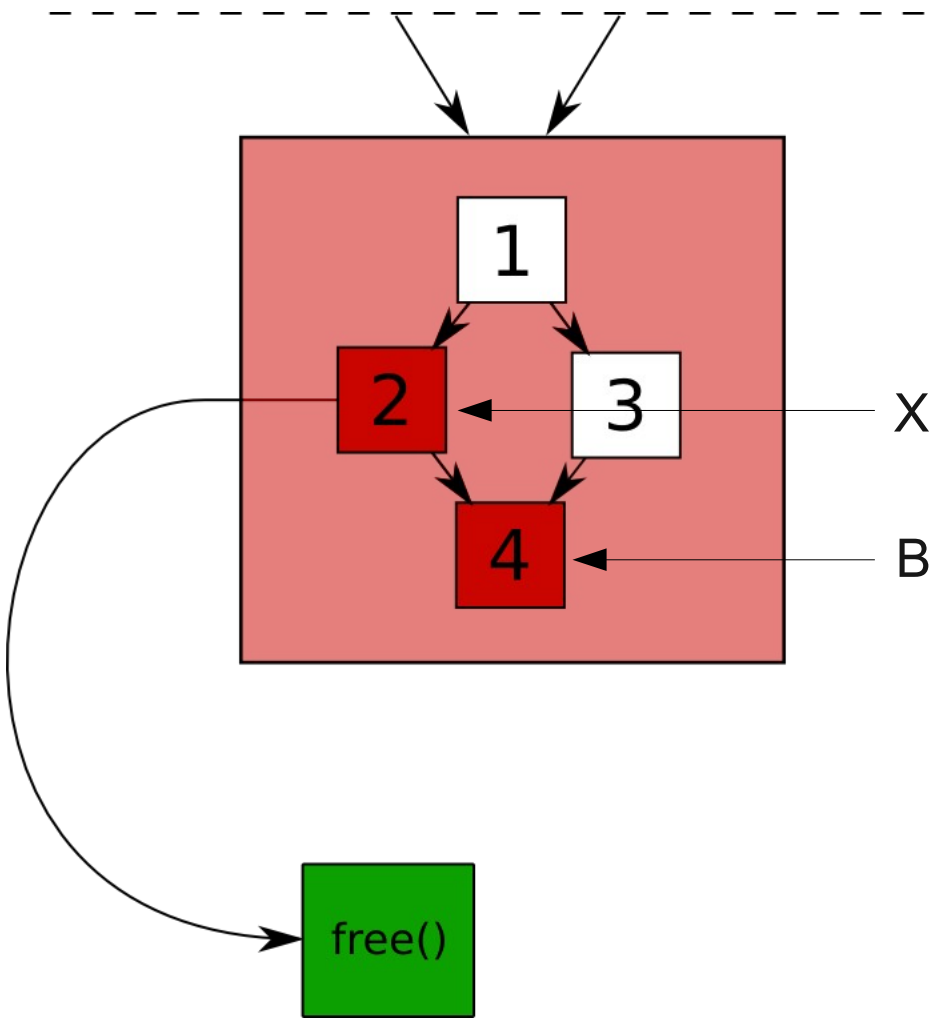


$B \in dom(X)$



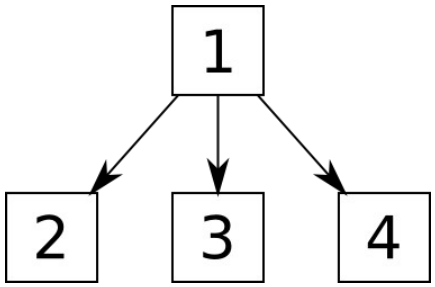
Example: use after free bug (warning)

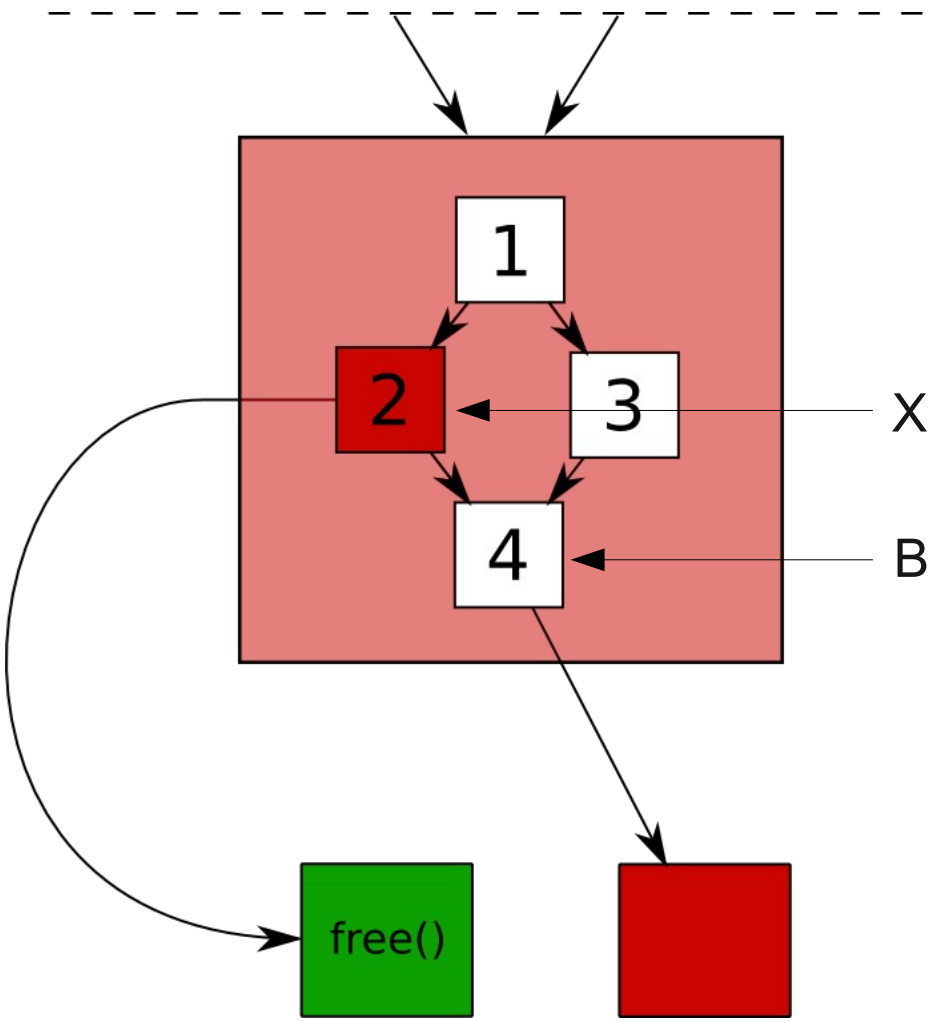




$$B \notin dom(X) \wedge B \in succ(X)$$

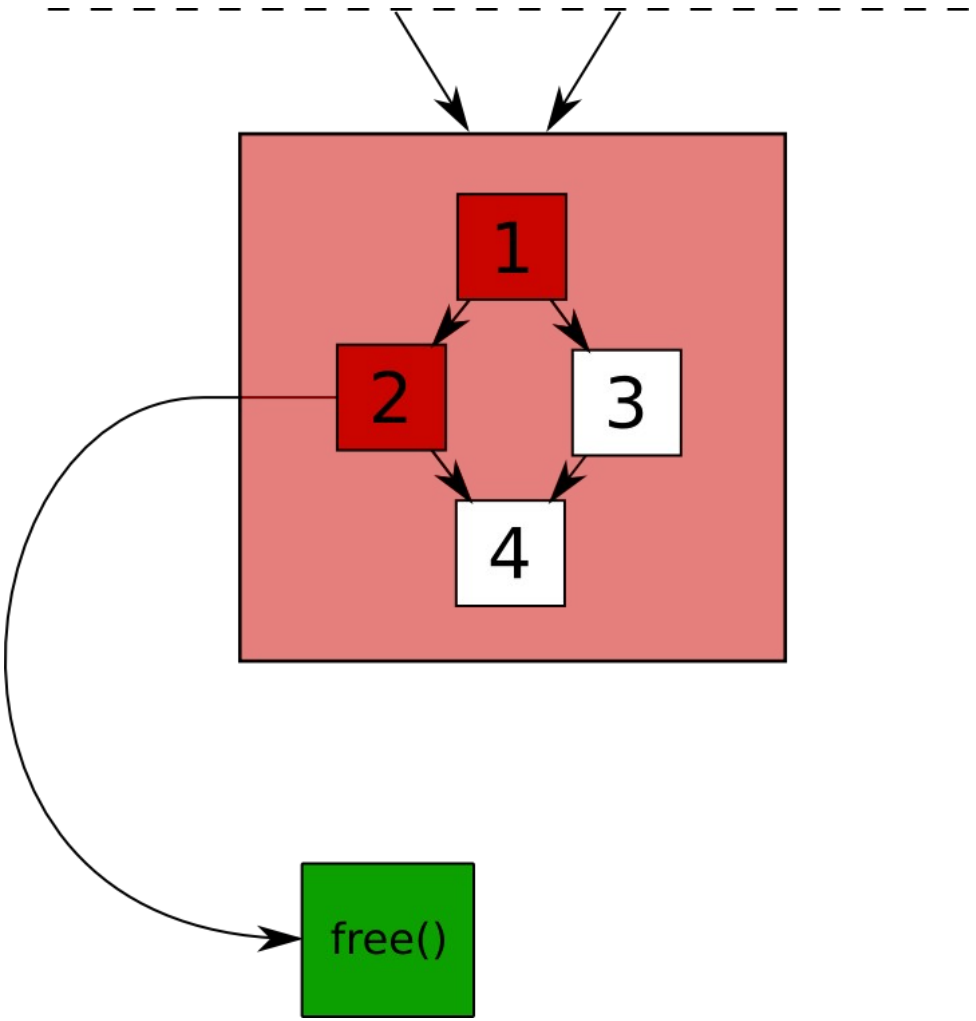
Example: use after free bug (warning)



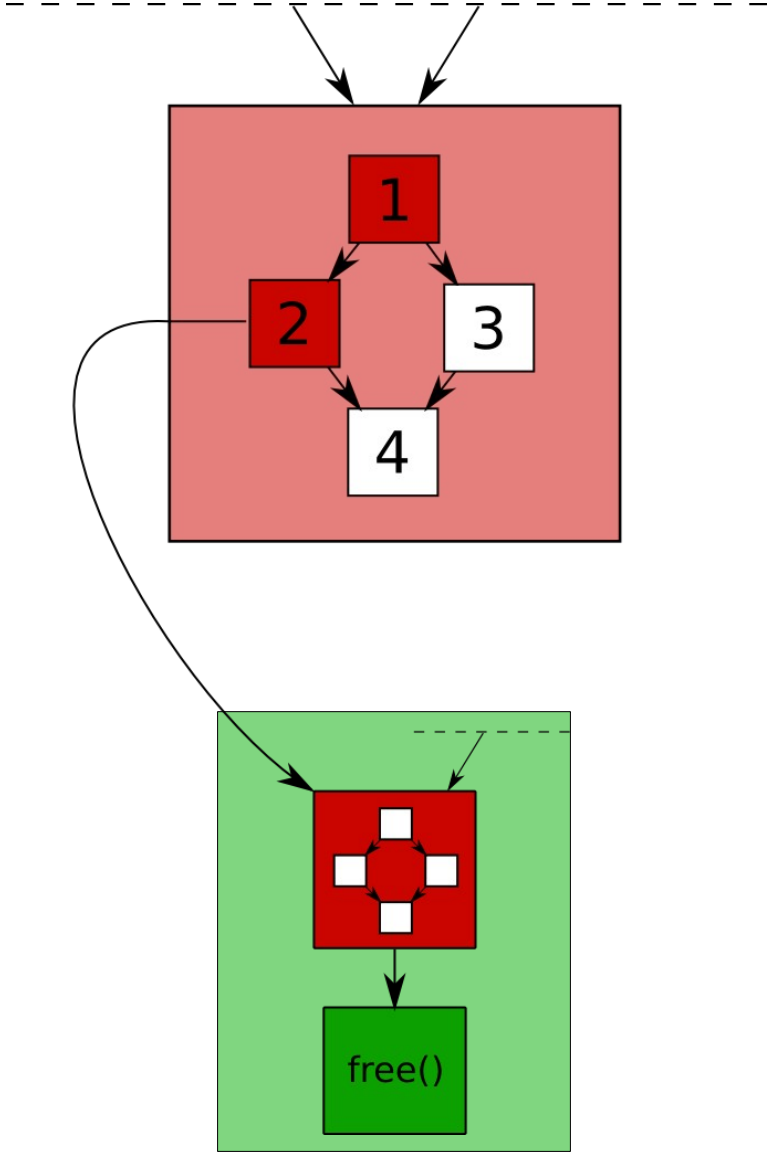


B !∈ dom(X)
∧
B ∈ succ(X)

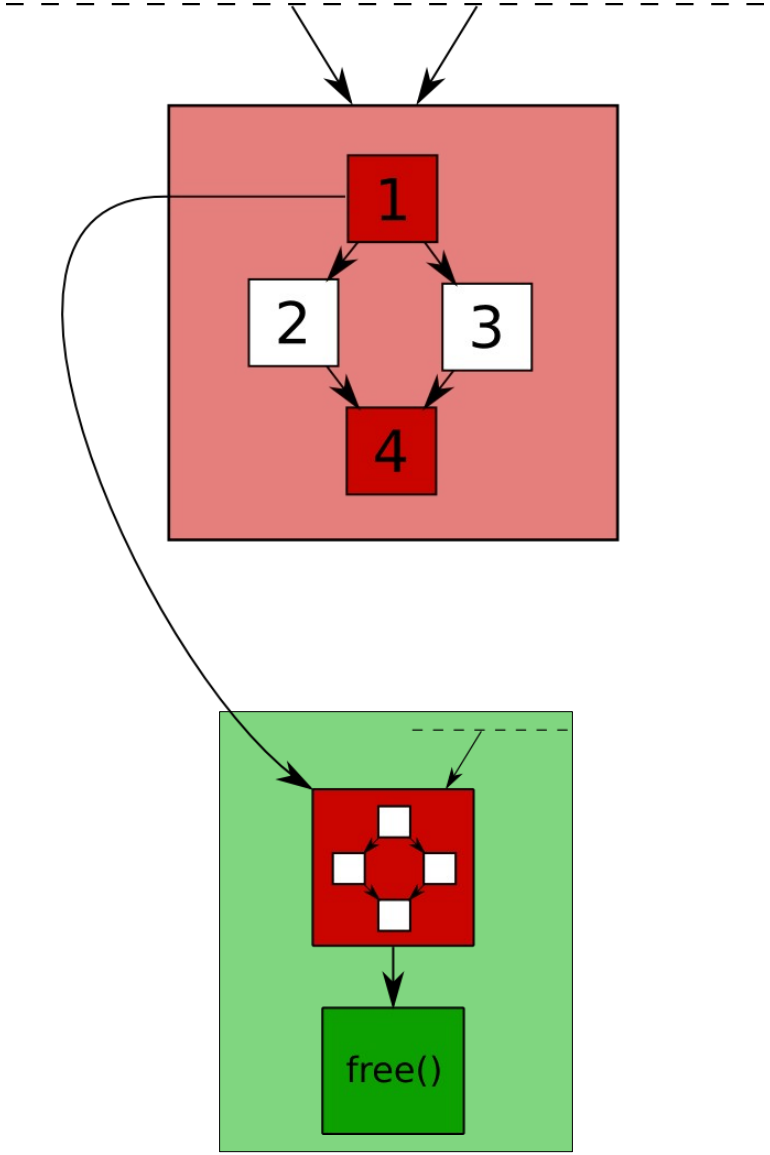
Example: no bugs



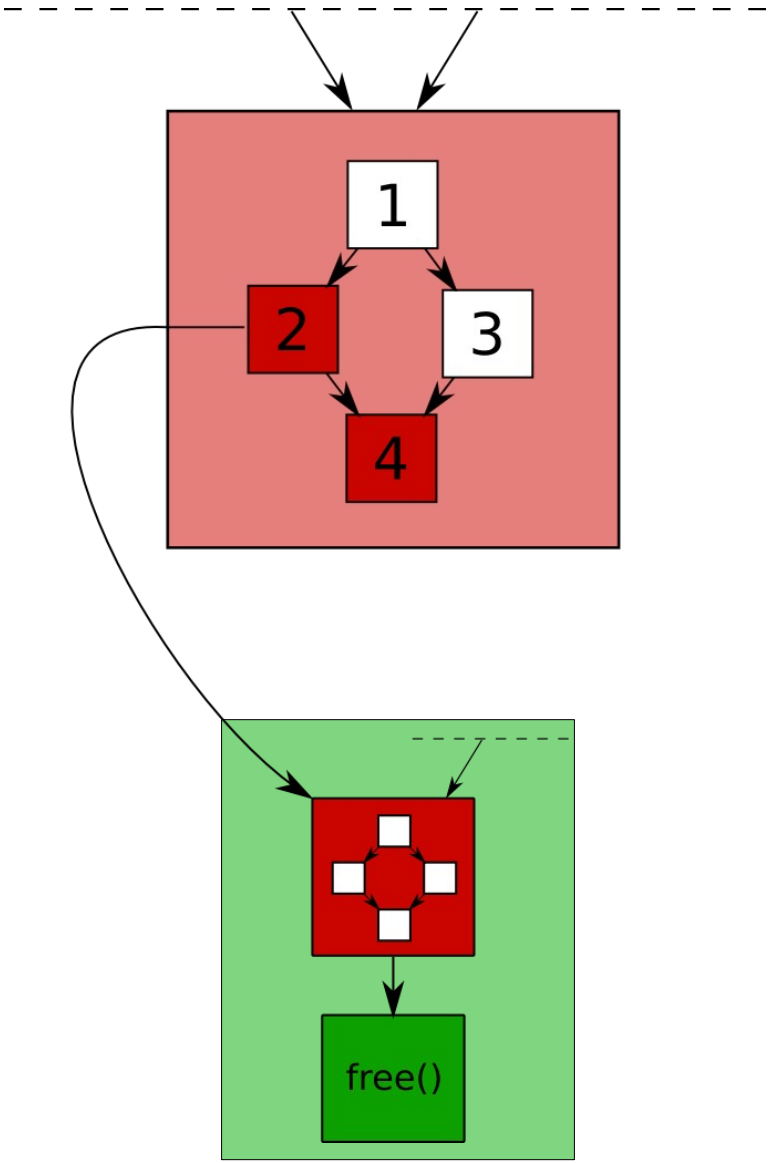
Example: no bugs



Example: use after free bug



Example: use after free bug (warning)



What’s the catch

- We cannot handle all data structures
- We cannot handle function pointers
- We have false positives
- We have false negatives
- Some “smart pointers”-like interfaces might not be covered
- The best use is for C++ life-span issues

Future

- Increase the number of covered data structure
- Use a solver to reduce false positives
- Import dynamic analysis data to mitigate the function pointers problem to reduce false negatives

That’s all folks