Dynamic instrumentation techniques

Ahmad shahnejat Michel Dagenais

May, 06

OUTLINE

- INTRODUCTION TO DYNAMIC INSTRUMENTATION
- TRAP INSTRUCTION
- INSTRUCTION PUNNING TECHNIQUE
- PROPOSED COMPILER-ASSISTED TECHNIQUE 1
- PROPOSED TECHNIQUE 2
- PROPOSED TECHNIQUE 3
- CONCLUSION AND FUTURE WORK

INT3 (CC ENCODING)

Trap-based vs. jump-based probes

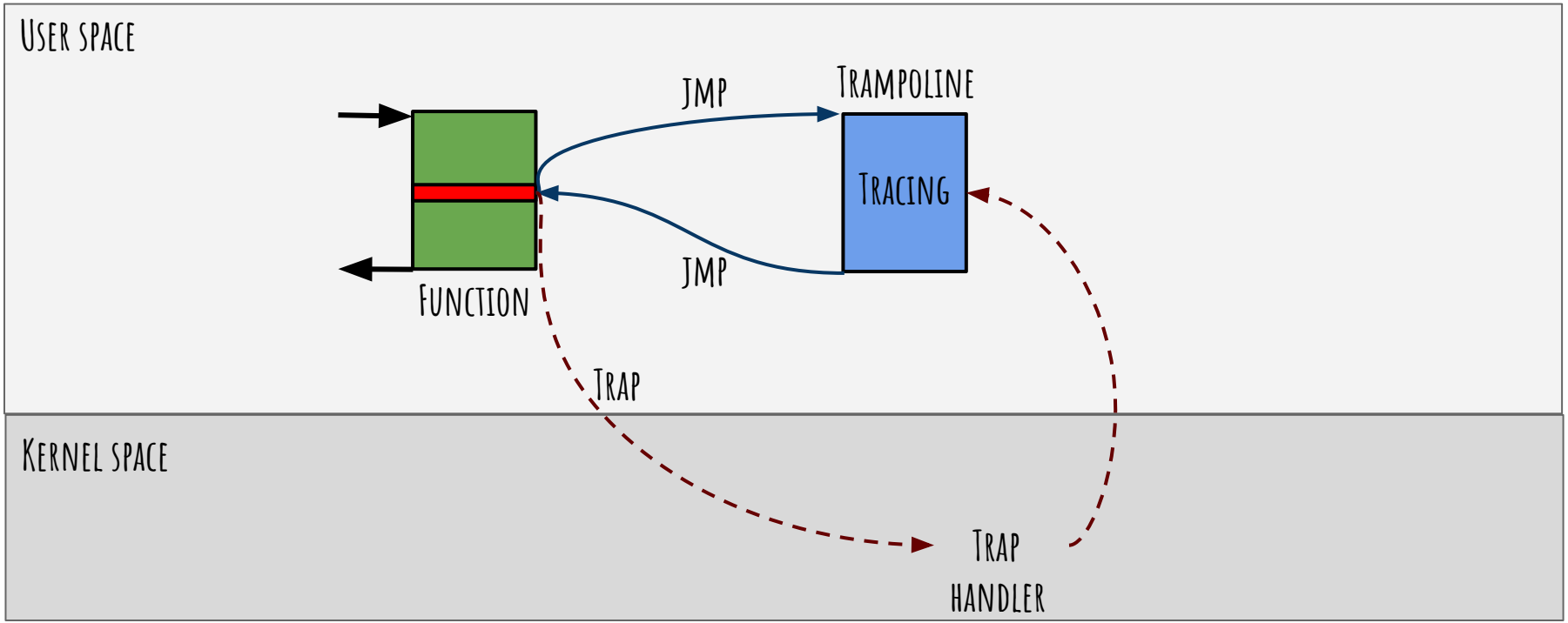
Trap-based probes:

- use an interrupt handler
- encoded with single-bytes (INT3 in the x86 instruction set) that will fit at any probe site atomically.
- substantial slow down along instrumentation (interrupt and userspace to kernel space switching)
- Trap-based probes are usually effective as a last option

jump-based probes:

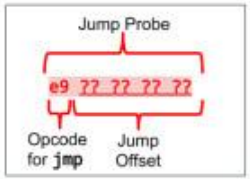
- redirects control flow directly to a trampoline rather than signal handlers.
- low invocation overhead
- Neighbor instructions will be overwritten, which is unsound If the probed instruction is smaller than the jump

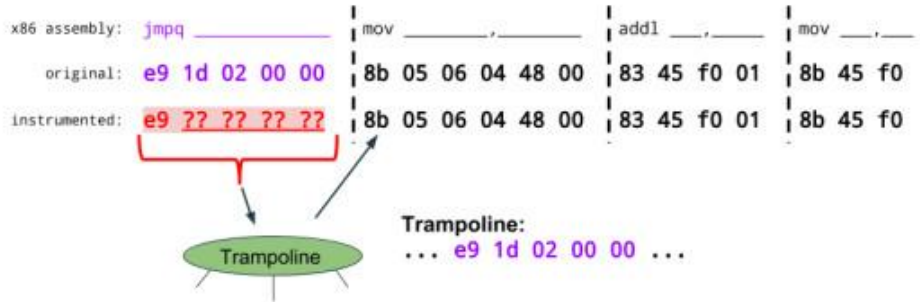
FastTP vs. new techniques



Jump-based tracepoints

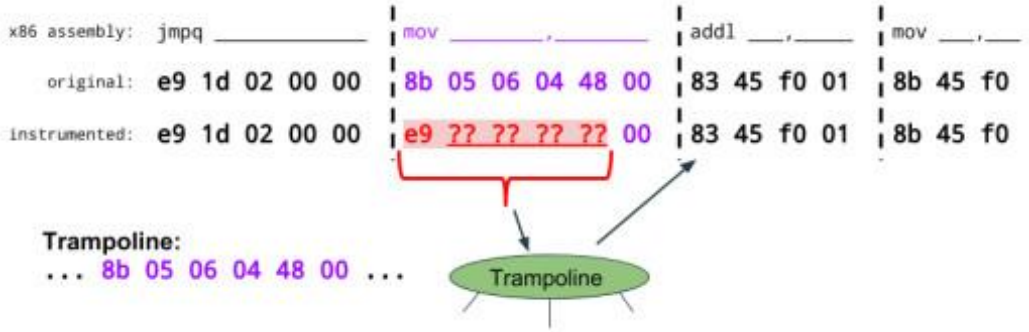
- If the probe site holds an instruction of five-bytes in length





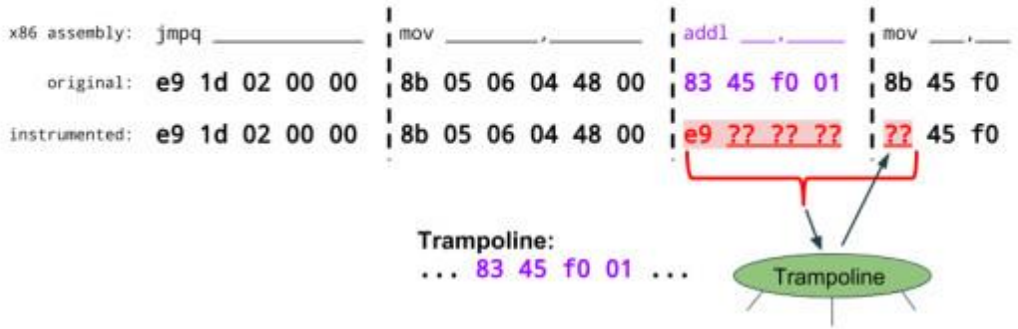
Jump-based tracepoints

- If the probe site holds a five-byte plus instruction



Jump-based tracepoints

- If the probe site holds an instruction shorter than 5 bytes



Instruction punning technique

- By injecting a jump instruction, the relative offset of the jump serves simultaneously both as data and as a sequence of instruction(s).

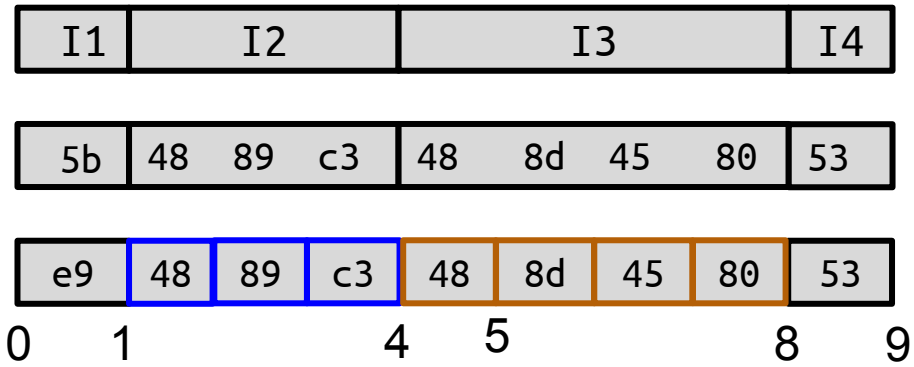
pun[1]
/pən/

noun

1. a joke exploiting the different possible meanings of a word or the fact that there are words which sound alike but have different meanings.
   "the pigs were a squeal (if you'll forgive the pun)"
   synonyms: play on words, wordplay, double entendre, double meaning, innuendo, witticism, quip; More

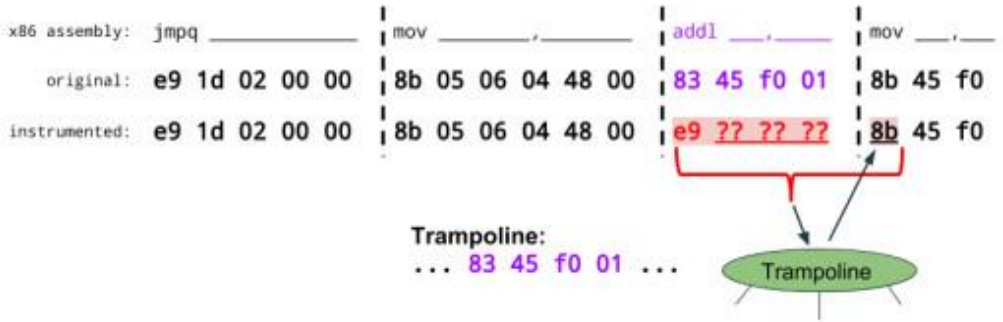
verb

1. make a joke exploiting the different possible meanings of a word.
   "his first puzzle punned on composers, with answers like “Handel with care” and “Haydn go seek”"



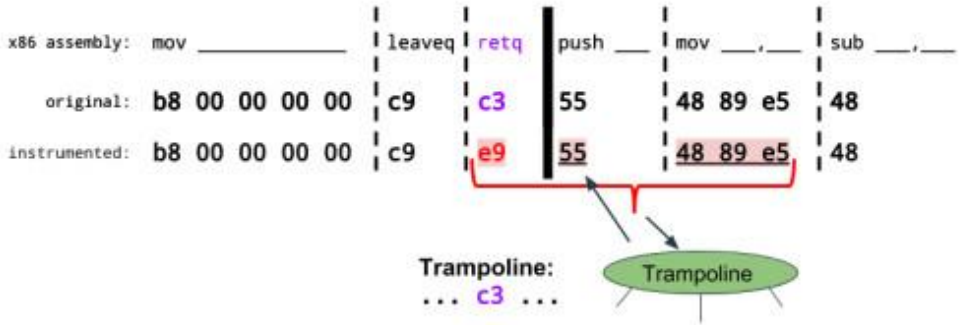
Instruction punning technique

- If the probe site holds an instruction shorter than 5 bytes

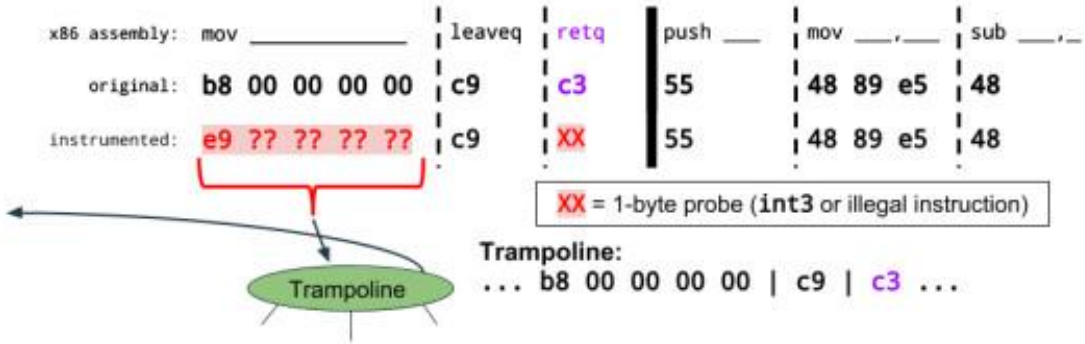


Instruction punning technique

- only one pun is available for the jump probe

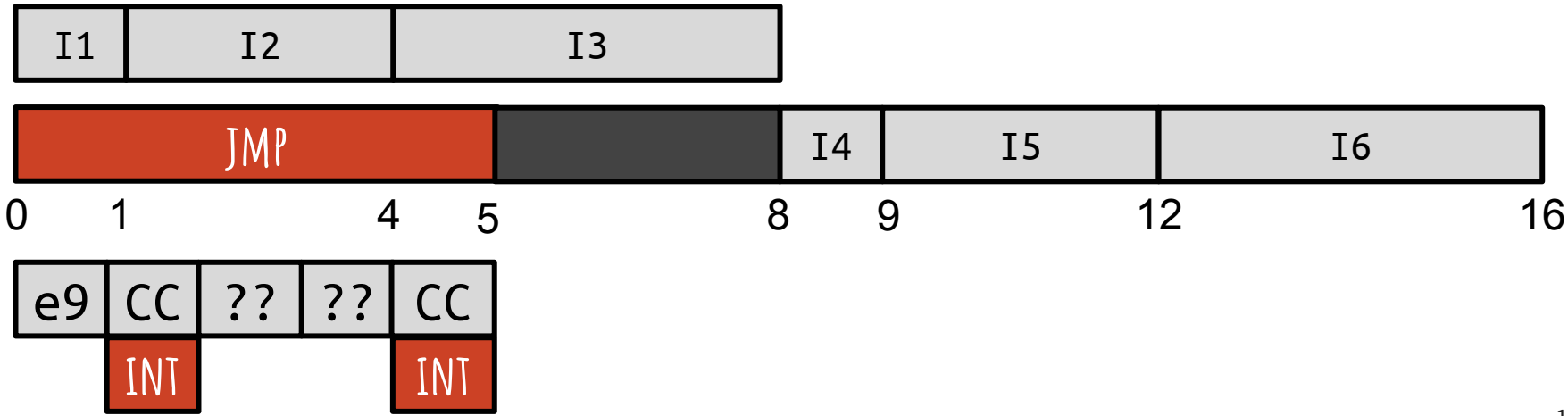


Instruction punning technique



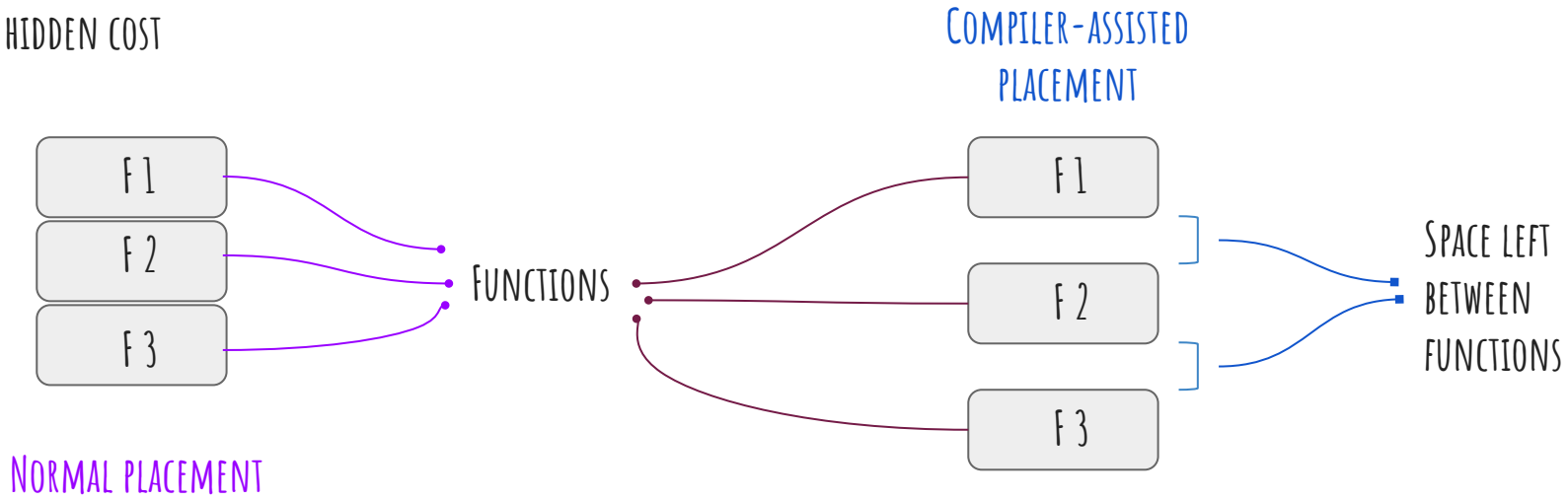
FastTP technique

- Max usage of trap instructions



Compiler-assisted Technique 1

- Forcing the compiler to leave space between functions
- have a hidden cost



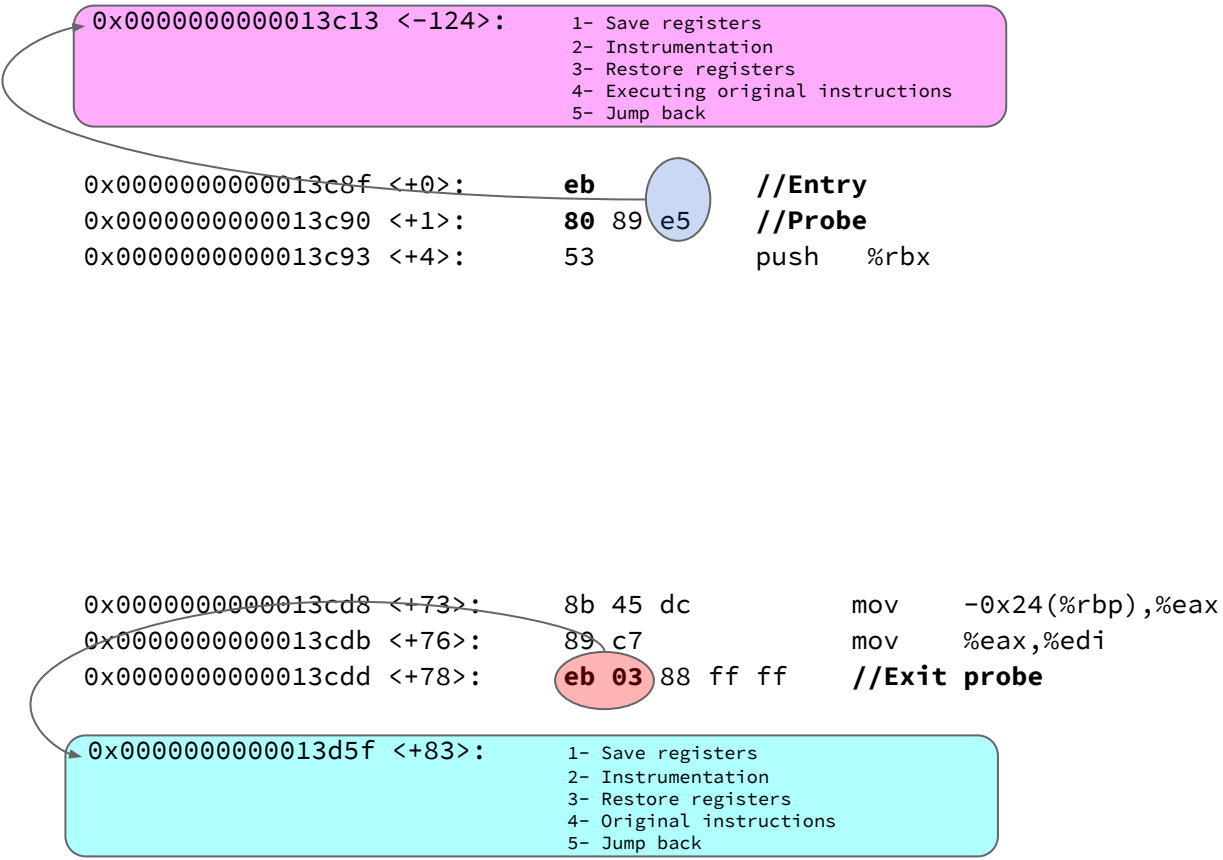
```
                                    1- Save registers
                                    2- Instrumentation
                                    3- Restore registers
                                    4- Executing original instructions
                                    5- Jump back
0x0000000000013c8f <+0>:     55      push   %rbp
0x0000000000013c90 <+1>:     48 89 e5        mov    %rsp,%rbp
0x0000000000013c93 <+4>:     53      push   %rbx
0x0000000000013c94 <+5>:     48 83 ec 28     sub    $0x28,%rsp
0x0000000000013c98 <+9>:     48 89 f8        mov    %rdi,%rax
0x0000000000013c9b <+12>:    48 89 c1        mov    %rax,%rcx
0x0000000000013c9e <+15>:    48 89 d3        mov    %rdx,%rbx
0x0000000000013ca1 <+18>:    48 89 f3        mov    %rsi,%rbx
0x0000000000013ca4 <+21>:    48 89 4d e0     mov    %rcx,-0x20(%rbp)
0x0000000000013ca8 <+25>:    48 89 5d e8     mov    %rbx,-0x18(%rbp)
0x0000000000013cac <+29>:    89 55 dc        mov    %edx,-0x24(%rbp)
0x0000000000013caf <+32>:    48 8b 55 e0     mov    -0x20(%rbp),%rdx
0x0000000000013cb3 <+36>:    48 8b 45 e8     mov    -0x18(%rbp),%rax
0x0000000000013cb7 <+40>:    48 89 d6        mov    %rdx,%rsi
0x0000000000013cba <+43>:    48 89 c2        mov    %rax,%rdx
0x0000000000013cbd <+46>:    48 8d 3d 3c d7 21 00    lea    0x21d73c(%rip),%rdi        # 0x231400 <_ZSt4cerr@@GLIBCXX_3.4>
0x0000000000013cc4 <+53>:    e8 38 47 00 00  callq  0x18401 <_ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St17basic_string_viewIS3_S4_E>
0x0000000000013cc9 <+58>:    48 8d 35 31 04 01 00    lea    0x10431(%rip),%rsi        # 0x24101
0x0000000000013cd0 <+65>:    48 89 c7        mov    %rax,%rdi
0x0000000000013cd3 <+68>:    e8 18 87 ff ff  callq  0xc3f0 <_ZStlsISt11char_traitsIcEERSt13basic_ostreamIcT_ES5_PKc@plt>
0x0000000000013cd8 <+73>:    8b 45 dc        mov    -0x24(%rbp),%eax
0x0000000000013cdb <+76>:    89 c7   mov    %eax,%edi
0x0000000000013cdd <+78>:    e8 2e 88 ff ff  callq  0xc510 <exit@plt>
                                    1- Save registers
                                    2- Instrumentation
                                    3- Restore registers
                                    4- Executing original instructions
                                    5- Jump back
```

```
1- Save registers
2- Instrumentation
3- Restore registers
4- Executing original instructions
5- Jump back
```

```
0x0000000000013c8f <+0>:     55          push   %rbp
0x0000000000013c90 <+1>:     48 89 e5    mov    %rsp,%rbp
0x0000000000013c93 <+4>:     53          push   %rbx
```

```
0x0000000000013cd8 <+73>:    8b 45 dc          mov    -0x24(%rbp),%eax
0x0000000000013cdb <+76>:    89 c7             mov    %eax,%edi
0x0000000000013cdd <+78>:    e8 2e 88 ff ff    callq  0xc510 <exit@plt>
```

```
1- Save registers
2- Instrumentation
3- Restore registers
4- Original instructions
5- Jump back
```


0x0000000000013c13 <-124>:
1- Save registers
2- Instrumentation
3- Restore registers
4- Executing original instructions
5- Jump back
0x0000000000013c8f <+0>: eb e5 //Entry
0x0000000000013c90 <+1>: 80 89 //Probe
0x0000000000013c93 <+4>: 53 push %rbx
0x0000000000013cd8 <+73>: 8b 45 dc mov -0x24(%rbp),%eax
0x0000000000013cdb <+76>: 89 c7 mov %eax,%edi
0x0000000000013cdd <+78>: eb 03 88 ff ff //Exit probe
0x0000000000013d5f <+83>:
1- Save registers
2- Instrumentation
3- Restore registers
4- Original instructions
5- Jump back

Technique 2

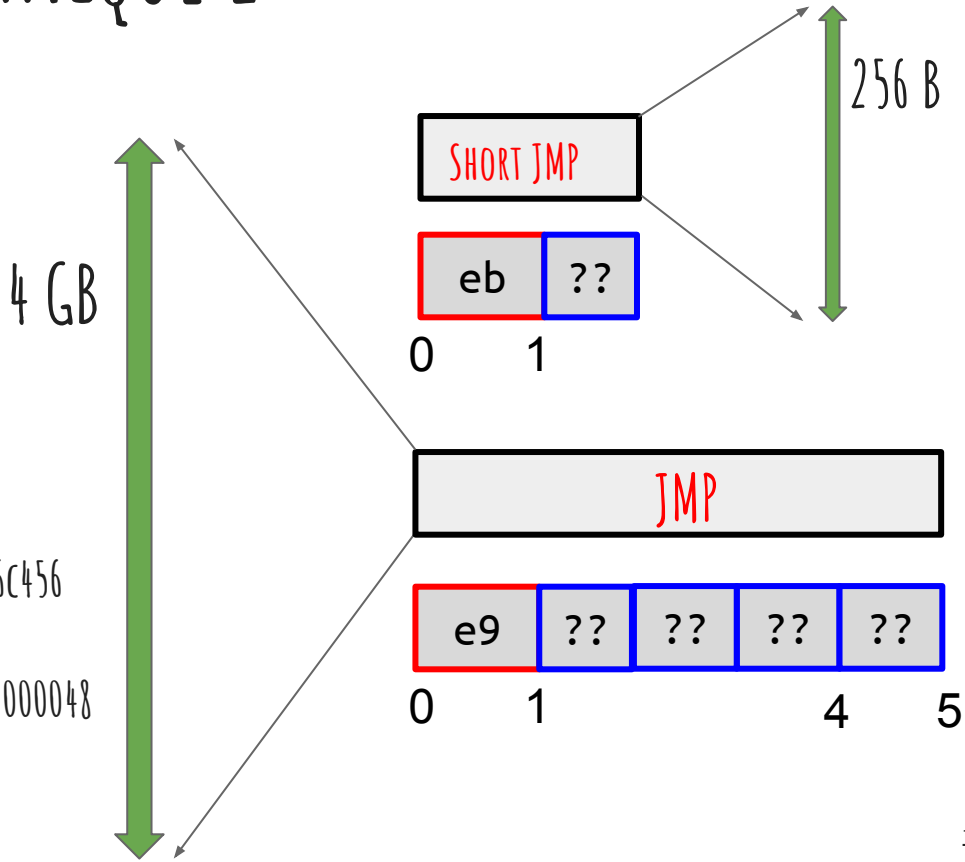
- Binary overlapping
- Why not using 2-byte short jump?
- How far the range of a jump could be?
- Landing on another jump/Call

e8	e9	43	00	00

Callq 0x5555556c456

e9	43	00	00	48

Jmp 0x48000048



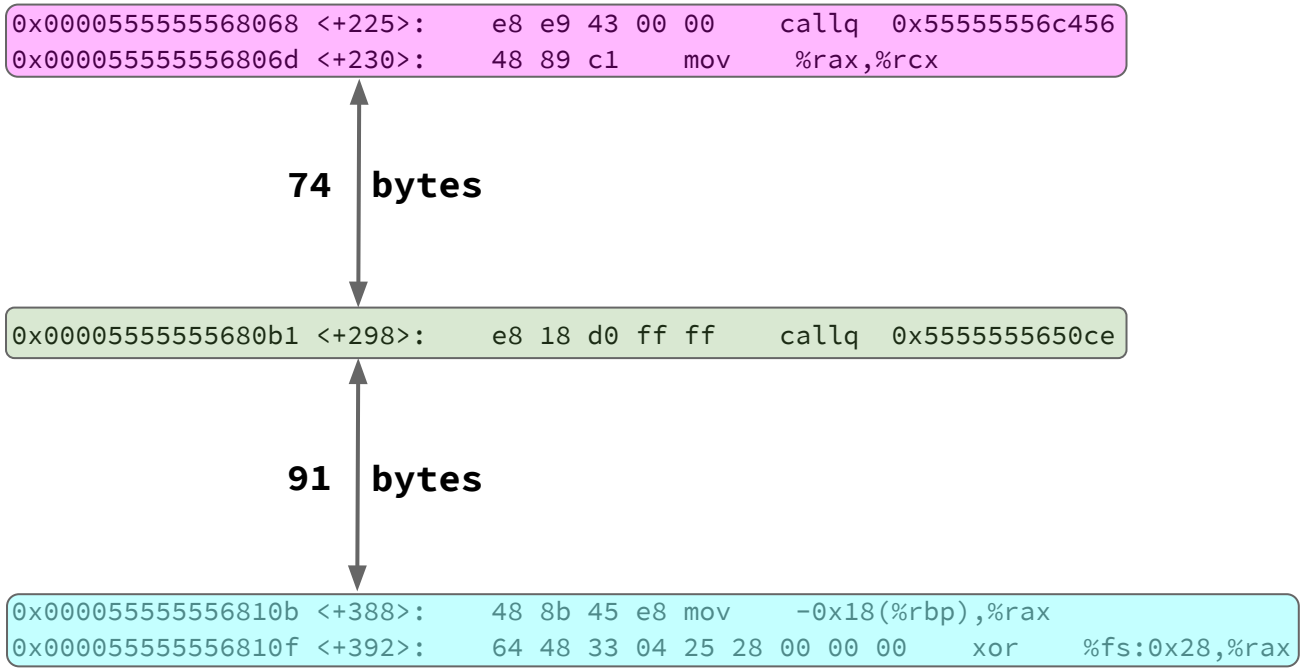
```
0x0000555555568068 <+225>:   e8 e9 43 00 00  callq  0x55555556c456 <_ZNKSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEcvSt17basic_string_viewIcS2_EEv>
0x000055555556806d <+230>:   48 89 c1        mov    %rax,%rcx
0x0000555555568070 <+233>:   48 89 d3        mov    %rdx,%rbx
0x0000555555568073 <+236>:   48 89 d0        mov    %rdx,%rax
0x0000555555568076 <+239>:   ba 01 00 00 00  mov    $0x1,%edx
0x000055555556807b <+244>:   48 89 cf        mov    %rcx,%rdi
0x000055555556807e <+247>:   48 89 c6        mov    %rax,%rsi
0x0000555555568081 <+250>:   e8 09 fc ff ff  callq  0x555555567c8f <_Z7do_exitSt17basic_string_viewIcSt11char_traitsIcEEi>
0x0000555555568086 <+255>:   e8 45 85 ff ff  callq  0x5555555605d0 <getpid@plt>
0x000055555556808b <+260>:   89 c2   mov    %eax,%edx
0x000055555556808d <+262>:   48 8d 45 80     lea    -0x80(%rbp),%rax
0x0000555555568091 <+266>:   89 d6   mov    %edx,%esi
0x0000555555568093 <+268>:   48 89 c7        mov    %rax,%rdi
0x0000555555568096 <+271>:   e8 05 99 ff ff  callq  0x5555555619a0 <_ZNSt7__cxx119to_stringEi>
0x000055555556809b <+276>:   48 8d 45 80     lea    -0x80(%rbp),%rax
0x000055555556809f <+280>:   48 89 c7        mov    %rax,%rdi
0x00005555555680a2 <+283>:   e8 35 ae ff ff  callq  0x555555562edc <_ZNKSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE4sizeEv>
0x00005555555680a7 <+288>:   48 89 c3        mov    %rax,%rbx
0x00005555555680aa <+291>:   48 8d 45 80     lea    -0x80(%rbp),%rax
0x00005555555680ae <+295>:   48 89 c7        mov    %rax,%rdi
0x00005555555680b1 <+298>:   e8 18 d0 ff ff  callq  0x5555555650ce <_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE4dataEv>
0x00005555555680b6 <+303>:   48 89 c1        mov    %rax,%rcx
0x00005555555680b9 <+306>:   8b 85 5c ff ff ff       mov    -0xa4(%rbp),%eax
0x00005555555680bf <+312>:   48 89 da        mov    %rbx,%rdx
0x00005555555680c2 <+315>:   48 89 ce        mov    %rcx,%rsi
0x00005555555680c5 <+318>:   89 c7   mov    %eax,%edi
0x00005555555680c7 <+320>:   e8 44 80 ff ff  callq  0x555555560110 <write@plt>
0x00005555555680cc <+325>:   8b 85 5c ff ff ff       mov    -0xa4(%rbp),%eax
0x00005555555680d2 <+331>:   89 c7   mov    %eax,%edi
0x00005555555680d4 <+333>:   e8 f7 86 ff ff  callq  0x5555555607d0 <close@plt>
0x00005555555680d9 <+338>:   48 8d 45 a0     lea    -0x60(%rbp),%rax
0x00005555555680dd <+342>:   48 89 c7        mov    %rax,%rdi
0x00005555555680e0 <+345>:   e8 21 2e 00 00  callq  0x55555556af06 <_ZNKSt12experimental10filesystem2v17__cxx114path5c_strEv>
0x00005555555680e5 <+350>:   be a4 01 00 00  mov    $0x1a4,%esi
0x00005555555680ea <+355>:   48 89 c7        mov    %rax,%rdi
0x00005555555680ed <+358>:   e8 8e 82 ff ff  callq  0x555555560380 <chmod@plt>
0x00005555555680f2 <+363>:   48 8d 45 80     lea    -0x80(%rbp),%rax
0x00005555555680f6 <+367>:   48 89 c7        mov    %rax,%rdi
0x00005555555680f9 <+370>:   e8 ec ab ff ff  callq  0x555555562cea <_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED2Ev>
0x00005555555680fe <+375>:   48 8d 45 a0     lea    -0x60(%rbp),%rax
0x0000555555568102 <+379>:   48 89 c7        mov    %rax,%rdi
0x0000555555568105 <+382>:   e8 d0 2d 00 00  callq  0x55555556aeda <_ZNSt12experimental10filesystem2v17__cxx114pathD2Ev>
0x000055555556810a <+387>:   90      nop
0x000055555556810b <+388>:   48 8b 45 e8     mov    -0x18(%rbp),%rax
0x000055555556810f <+392>:   64 48 33 04 25 28 00 00 00      xor    %fs:0x28,%rax
0x0000555555568118 <+401>:   74 57   je     0x555555568171 <_Z11lock_daemonPvi+490>
```

e9 43 00 00 48 = jmp 0x48000048

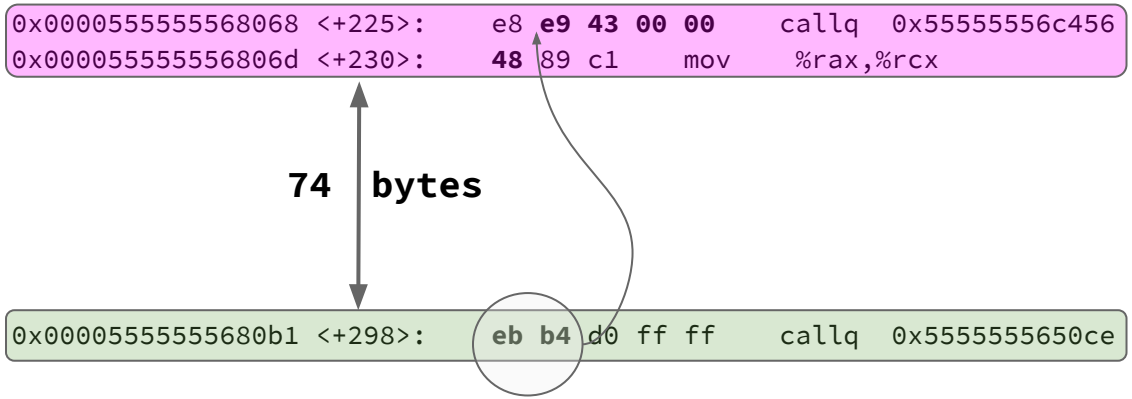
e8 64 48 33 04 = call 0x4334869


0x0000555555568068 <+225>: e8 e9 43 00 00 callq 0x55555556c456 <_ZNKSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEEcvSt17basic_string_viewIcS2_EEv>
0x000055555556806d <+230>: 48 89 c1 mov %rax,%rcx
0x0000555555568070 <+233>: 48 89 d3 mov %rdx,%rbx
0x0000555555568073 <+236>: 48 89 d0 mov %rdx,%rax
0x0000555555568076 <+239>: ba 01 00 00 00 mov $0x1,%edx
0x000055555556807b <+244>: 48 89 cf mov %rcx,%rdi
0x000055555556807e <+247>: 48 89 c6 mov %rax,%rsi
0x0000555555568081 <+250>: e8 09 fc ff ff callq 0x555555567c8f <_Z7do_exitSt17basic_string_viewIcSt11char_traitsIcEEi>
0x0000555555568086 <+255>: e8 45 85 ff ff callq 0x5555555605d0 <getpid@plt>
0x000055555556808b <+260>: 89 c2 mov %eax,%edx
0x000055555556808d <+262>: 48 8d 45 80 lea -0x80(%rbp),%rax
0x0000555555568091 <+266>: 89 d6 mov %edx,%esi
0x0000555555568093 <+268>: 48 89 c7 mov %rax,%rdi
0x0000555555568096 <+271>: e8 05 99 ff ff callq 0x5555555619a0 <_ZNSt7__cxx119to_stringEi>
0x000055555556809b <+276>: 48 8d 45 80 lea -0x80(%rbp),%rax
0x000055555556809f <+280>: 48 89 c7 mov %rax,%rdi
0x00005555555680a2 <+283>: e8 35 ae ff ff callq 0x555555562edc <_ZNKSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE4sizeEv>
0x00005555555680a7 <+288>: 48 89 c3 mov %rax,%rbx
0x00005555555680aa <+291>: 48 8d 45 80 lea -0x80(%rbp),%rax
0x00005555555680ae <+295>: 48 89 c7 mov %rax,%rdi
0x00005555555680b1 <+298>: e8 18 d0 ff ff callq 0x5555555650ce <_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEE4dataEv>
0x00005555555680b6 <+303>: 48 89 c1 mov %rax,%rcx
0x00005555555680b9 <+306>: 8b 85 5c ff ff ff mov -0xa4(%rbp),%eax
0x00005555555680bf <+312>: 48 89 da mov %rbx,%rdx
0x00005555555680c2 <+315>: 48 89 ce mov %rcx,%rsi
0x00005555555680c5 <+318>: 89 c7 mov %eax,%edi
0x00005555555680c7 <+320>: e8 44 80 ff ff callq 0x555555560110 <write@plt>
0x00005555555680cc <+325>: 8b 85 5c ff ff ff mov -0xa4(%rbp),%eax
0x00005555555680d2 <+331>: 89 c7 mov %eax,%edi
0x00005555555680d4 <+333>: e8 f7 86 ff ff callq 0x5555555607d0 <close@plt>
0x00005555555680d9 <+338>: 48 8d 45 a0 lea -0x60(%rbp),%rax
0x00005555555680dd <+342>: 48 89 c7 mov %rax,%rdi
0x00005555555680e0 <+345>: e8 21 2e 00 00 callq 0x55555556af06 <_ZNKSt12experimental10filesystem2v17__cxx114path5c_strEv>
0x00005555555680e5 <+350>: be a4 01 00 00 mov $0x1a4,%esi
0x00005555555680ea <+355>: 48 89 c7 mov %rax,%rdi
0x00005555555680ed <+358>: e8 8e 82 ff ff callq 0x555555560380 <chmod@plt>
0x00005555555680f2 <+363>: 48 8d 45 80 lea -0x80(%rbp),%rax
0x00005555555680f6 <+367>: 48 89 c7 mov %rax,%rdi
0x00005555555680f9 <+370>: e8 ec ab ff ff callq 0x555555562cea <_ZNSt7__cxx1112basic_stringIcSt11char_traitsIcESaIcEED2Ev>
0x00005555555680fe <+375>: 48 8d 45 a0 lea -0x60(%rbp),%rax
0x0000555555568102 <+379>: 48 89 c7 mov %rax,%rdi
0x0000555555568105 <+382>: e8 d0 2d 00 00 callq 0x55555556aeda <_ZNSt12experimental10filesystem2v17__cxx114pathD2Ev>
0x000055555556810a <+387>: 90 nop
0x000055555556810b <+388>: 48 8b 45 e8 mov -0x18(%rbp),%rax
0x000055555556810f <+392>: 64 48 33 04 25 28 00 00 00 xor %fs:0x28,%rax
0x0000555555568118 <+401>: 74 57 je 0x555555568171 <_Z11lock_daemonv+490>
e9 43 00 00 48 = jmp 0x48000048
e8 64 48 33 04 = call 0x4334869

Technique 2


0x0000555555568068 <+225>: e8 e9 43 00 00 callq 0x55555556c456
0x000055555556806d <+230>: 48 89 c1 mov %rax,%rcx
74 bytes
0x00005555555680b1 <+298>: e8 18 d0 ff ff callq 0x5555555650ce
91 bytes
0x000055555556810b <+388>: 48 8b 45 e8 mov -0x18(%rbp),%rax
0x000055555556810f <+392>: 64 48 33 04 25 28 00 00 00 xor %fs:0x28,%rax

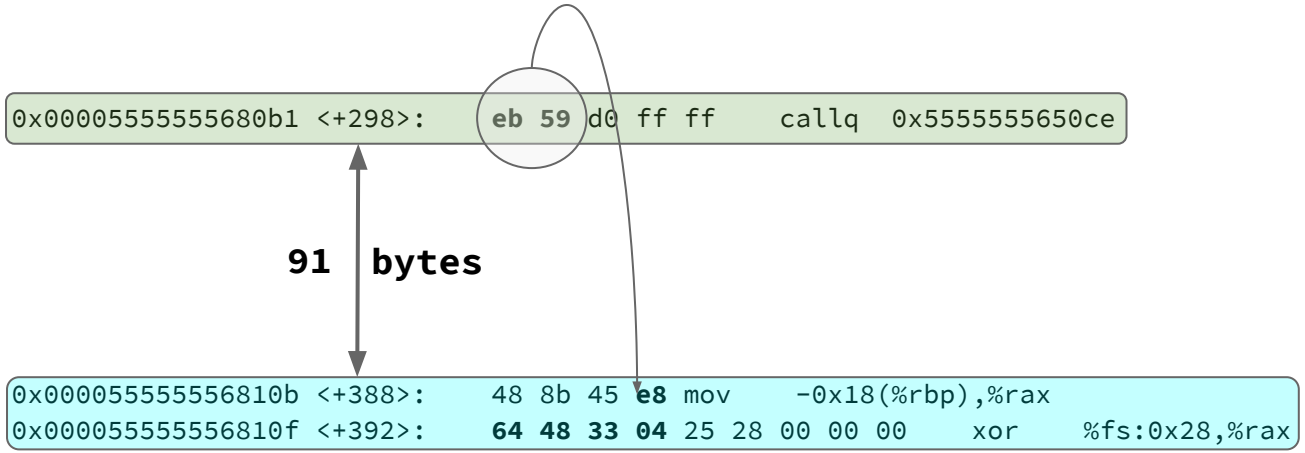
Technique 2




```
0x000055555556810b <+388>:    48 8b 45 e8 mov    -0x18(%rbp),%rax
0x000055555556810f <+392>:    64 48 33 04 25 28 00 00 00    xor    %fs:0x28,%rax
```

Technique 2

```
0x0000555555568068 <+225>:    e8 e9 43 00 00    callq  0x55555556c456
0x000055555556806d <+230>:    48 89 c1    mov    %rax,%rcx
```

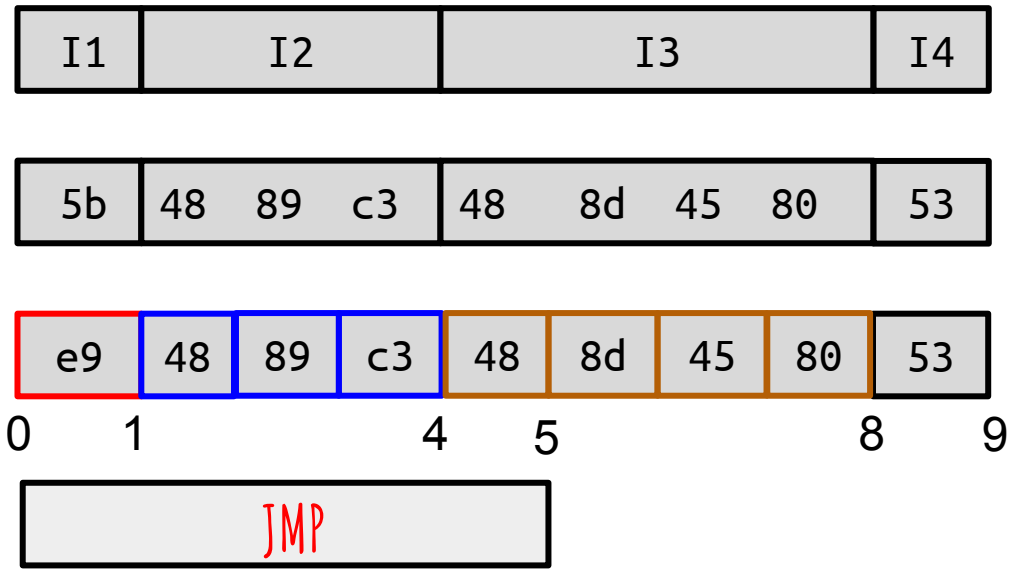


Technique 3

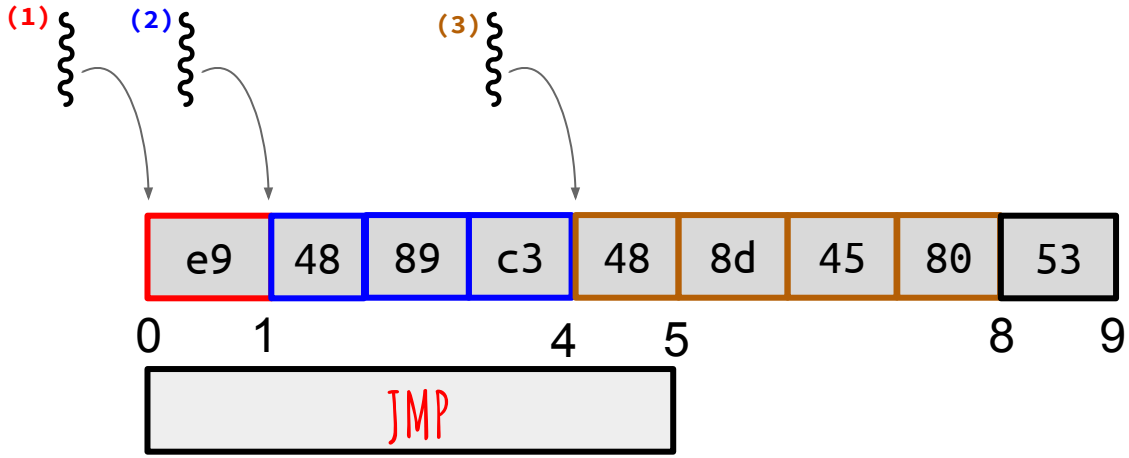
- Instrumentation of a five-byte location with multiple instructions.

- Reusing the suffix of an instruction as a distinct instruction is used mainly in code obfuscation.

- 1st: instruction punning
  2nd: ?



Technique 3



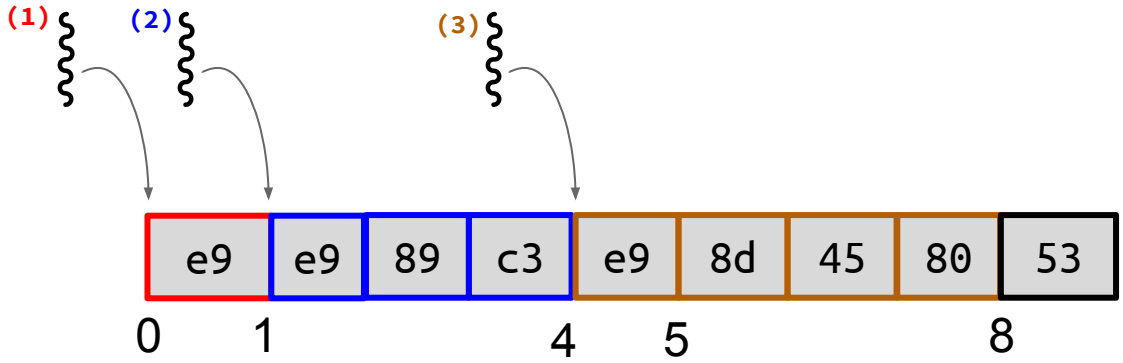
(1): E9 48 89 c3 48 = `jmp 0x48c3894d` → Need to be validated

(2): 48 89 c3 =
```
dec     eax
mov     ebx,eax
```

(3): 48 8d 45 80 =
```
dec     eax
lea     eax,[ebp-0x80]
```

→ Original instructions

Technique 3



(1): e9 e9 ?? ?? e9

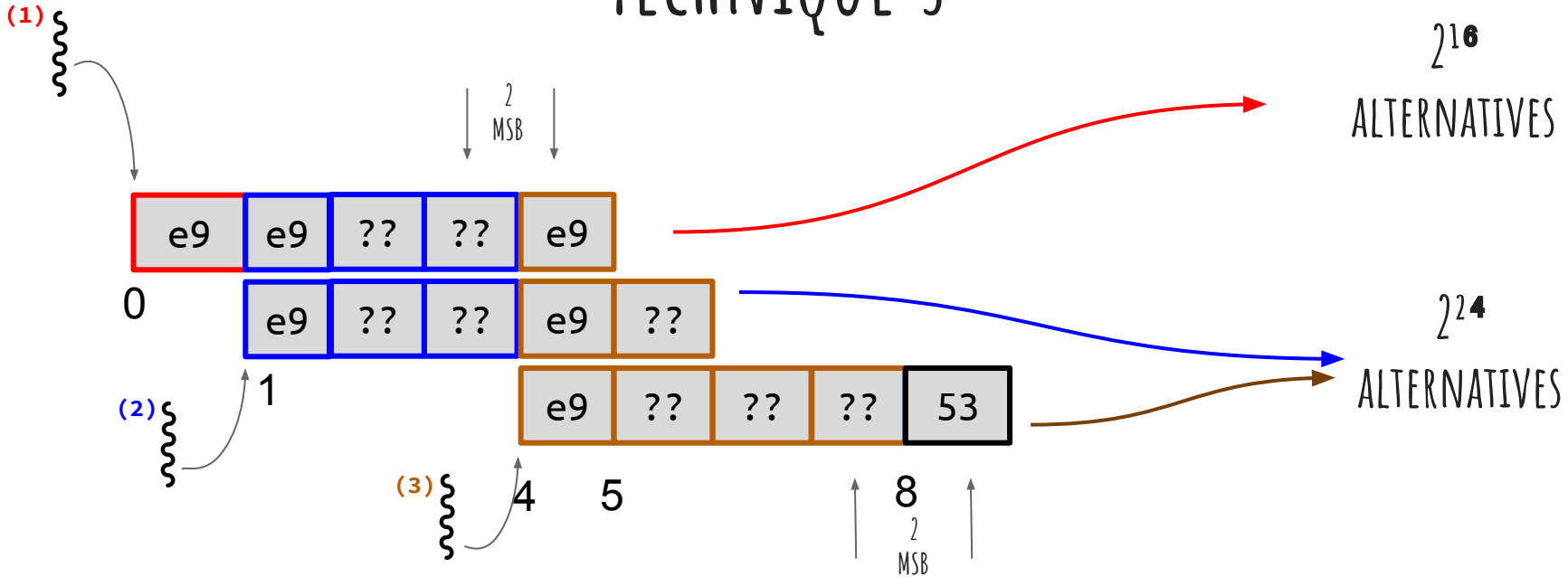
(2): e9 ?? ?? e9 ??

(3): e9 ?? ?? ?? 53

2 bytes available to manipulate

3 bytes available to manipulate

Technique 3



- In practice it typically takes no more than 7 attempts(for the two significant bytes) to map memory for a trampoline, while we have at least 256 alternatives in this cases.

Conclusion & Future work

Conclusion & Future work

- The key goal is interpreting data as code.
  this technique is called instruction punning.

- 1st approach: Instruction punning
- 2nd approach: Proposed techniques
- last approach: Trap instruction(s)
- Trampoline placement
- Prototype under development

Questions?!

:)

References

1- B. Chamith, B. J. Svensson, L. Dalessandro, and R. R. Newton. Instruction punning:
Lightweight instrumentation for x86-64. In Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation, 2017.

2- Zhao, Valerie, "Evaluation of Dynamic Binary Instrumentation Approaches: Dynamic Binary Translation vs. Dynamic Probe Injection" (2018).