I Know what your compiler did:

Optimization Effects on Power Side-Channel Leakage for RISC-V

Durba Chatterjee, Asmita Adhikary, Abraham Basurto, Senna van Hoek, Eloi Sanfelix, Lejla Batina, Ileana Buhan

Real World Crypto Symposium 2025

Radboud University

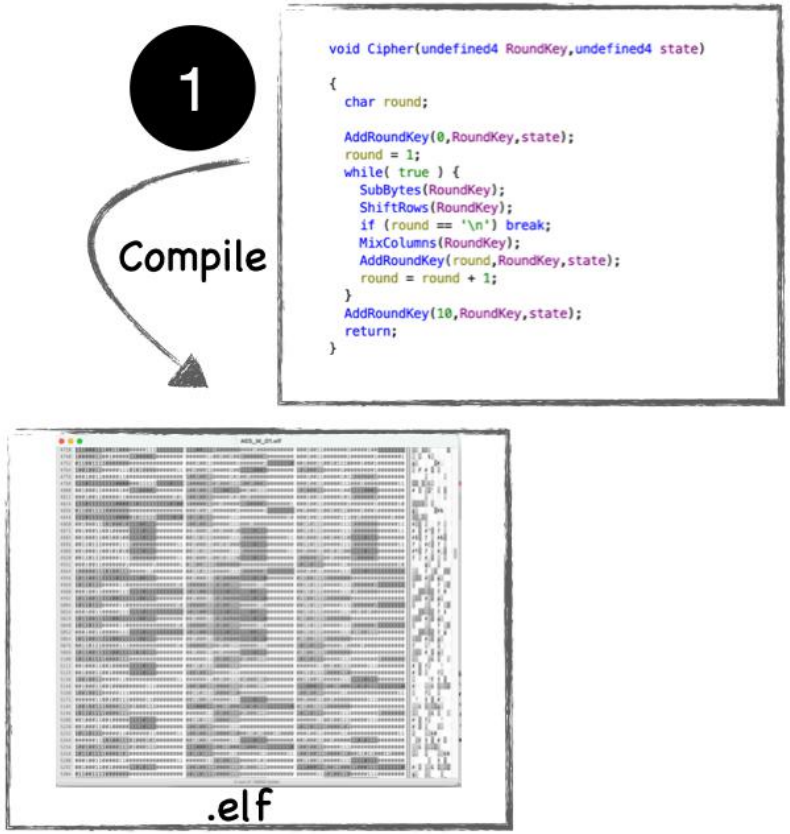
Power Side-Channel Analysis

```
void Cipher(undefined4 RoundKey,undefined4 state)

{
  char round;

  AddRoundKey(0,RoundKey,state);
  round = 1;
  while( true ) {
    SubBytes(RoundKey);
    ShiftRows(RoundKey);
    if (round == '\n') break;
    MixColumns(RoundKey);
    AddRoundKey(round,RoundKey,state);
    round = round + 1;
  }
  AddRoundKey(10,RoundKey,state);
  return;
}
```

Cryptographic implementation (software)

Slide Courtesy: Ileana Buhan

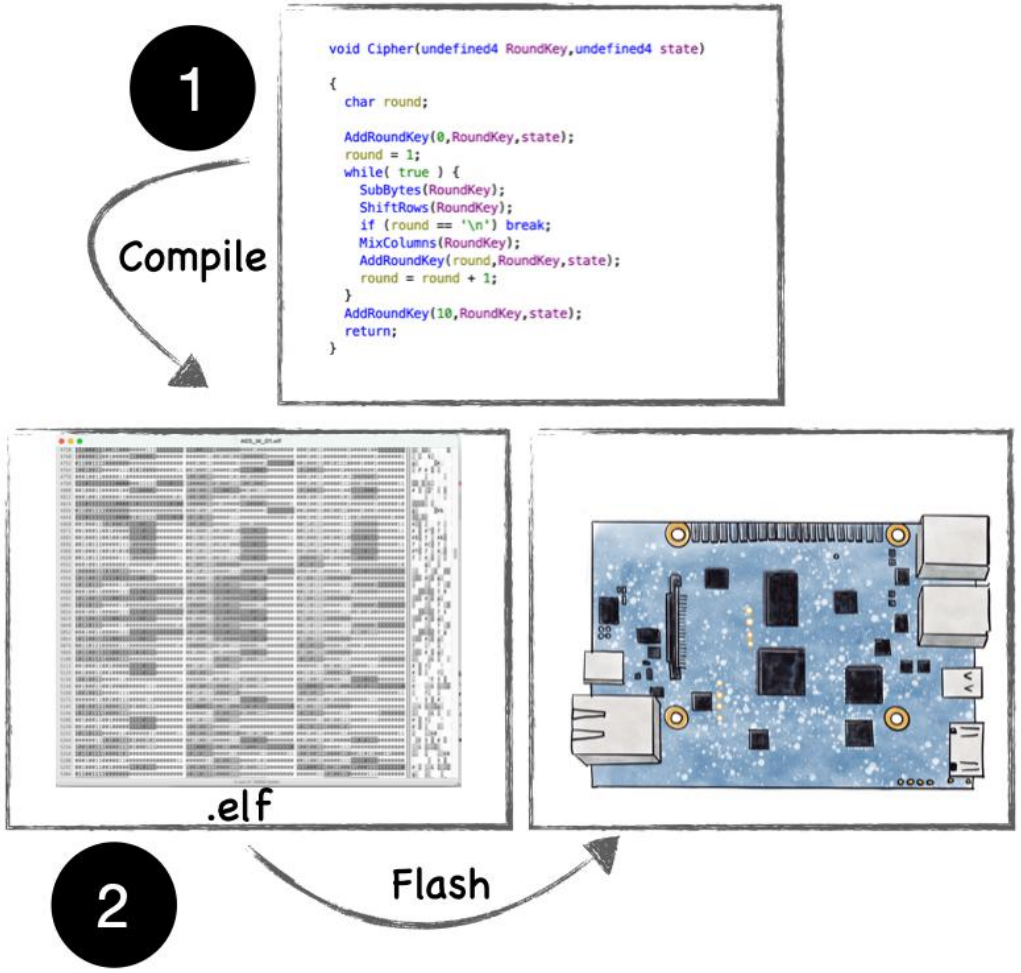
Power Side-Channel Analysis



Binary/Executable file

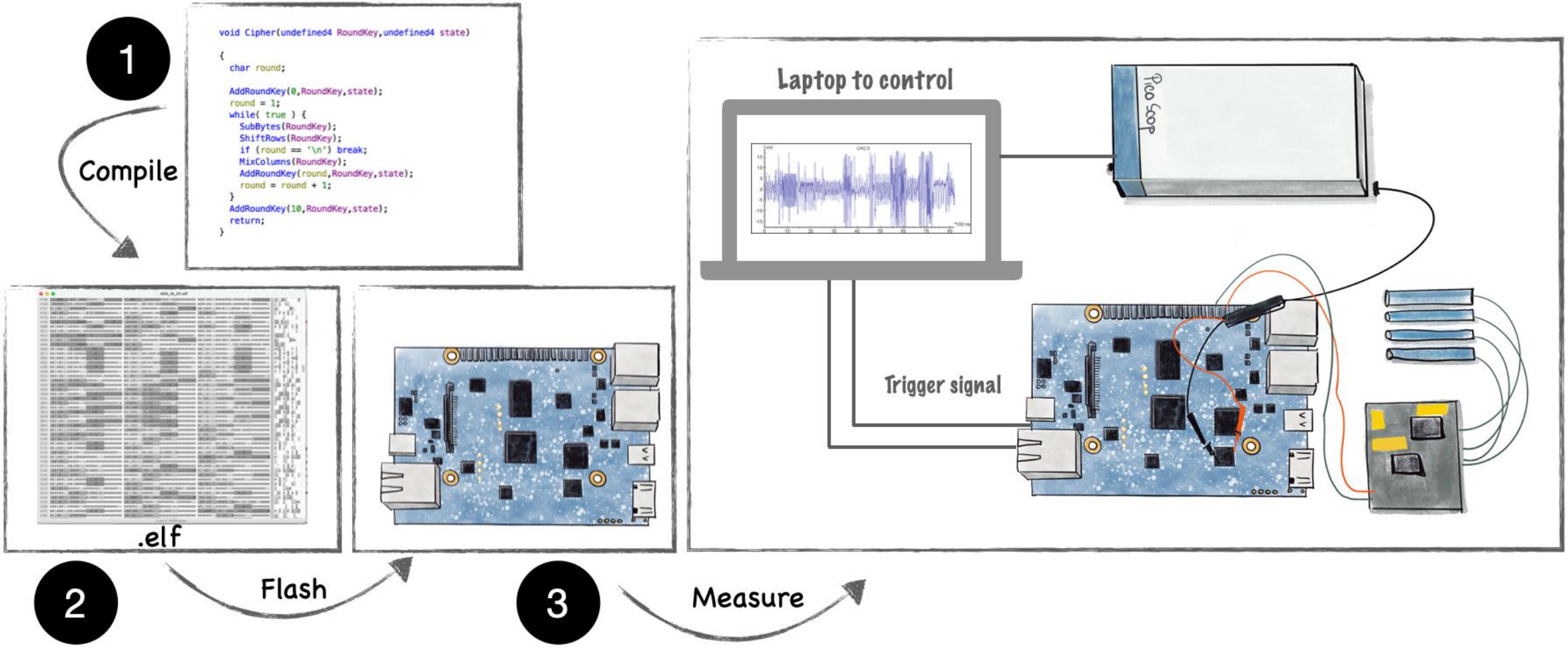
Slide Courtesy: Ileana Buhan

Power Side-Channel Analysis



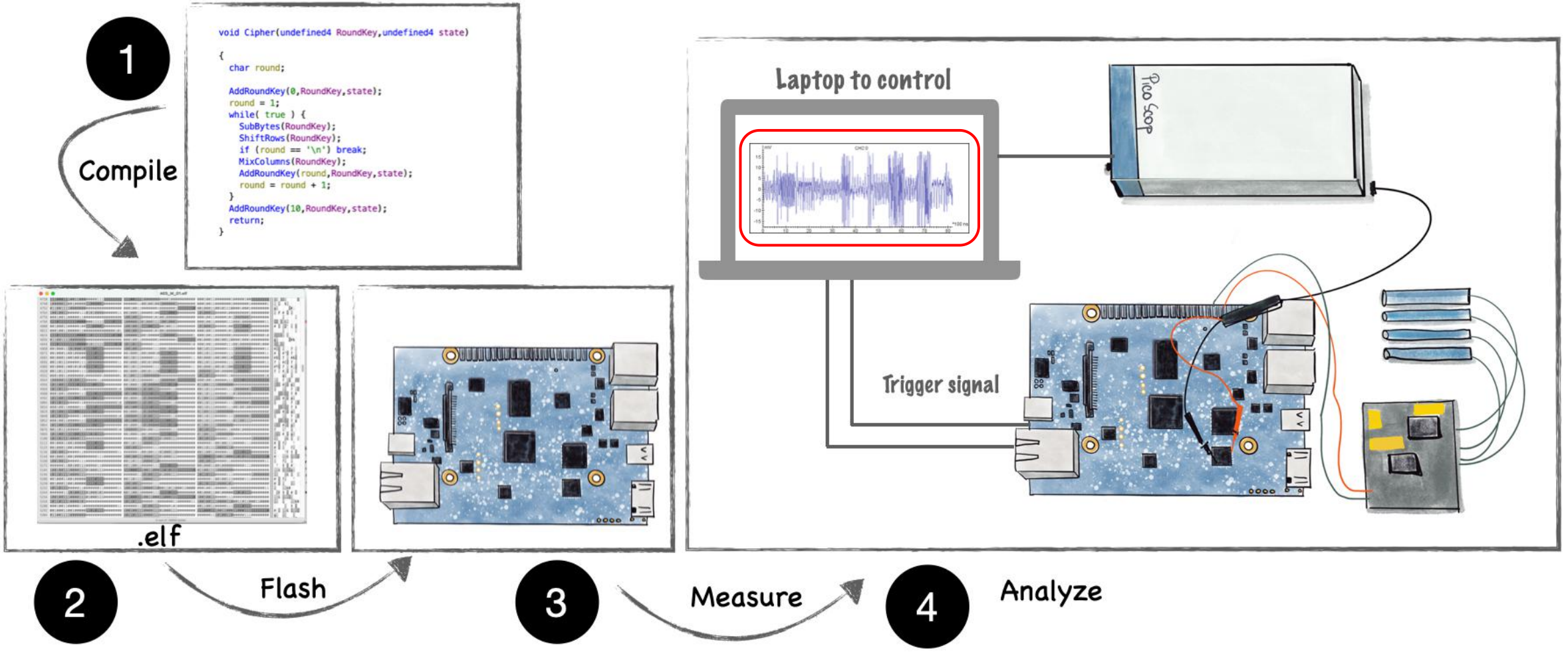
Slide Courtesy: Ileana Buhan

Power Side-Channel Analysis



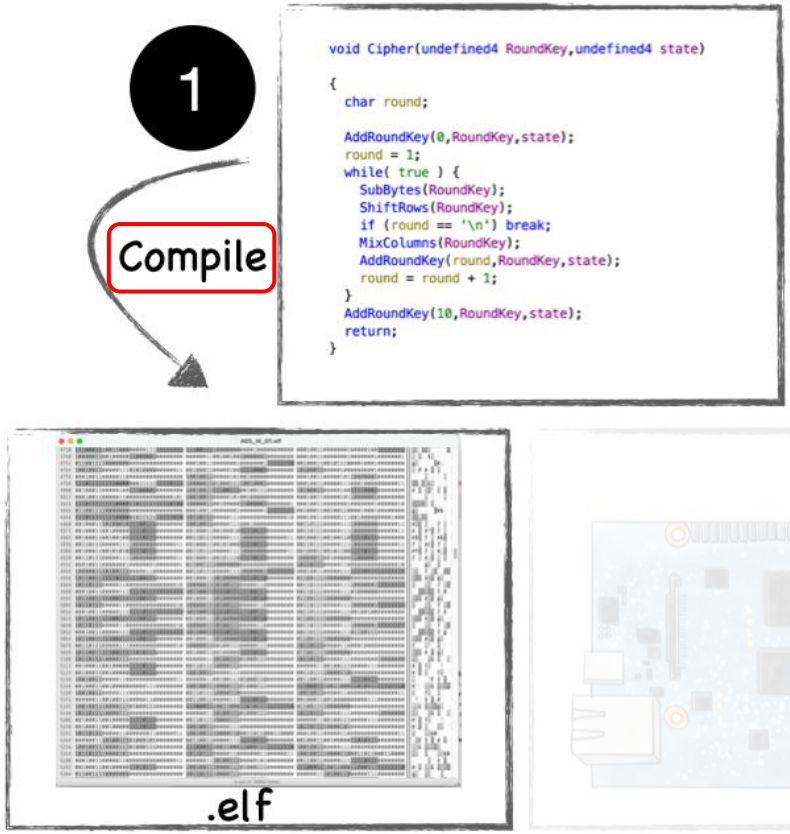
Slide Courtesy: Ileana Buhan

Power Side-Channel Analysis



Slide Courtesy: Ileana Buhan

Compilation Options

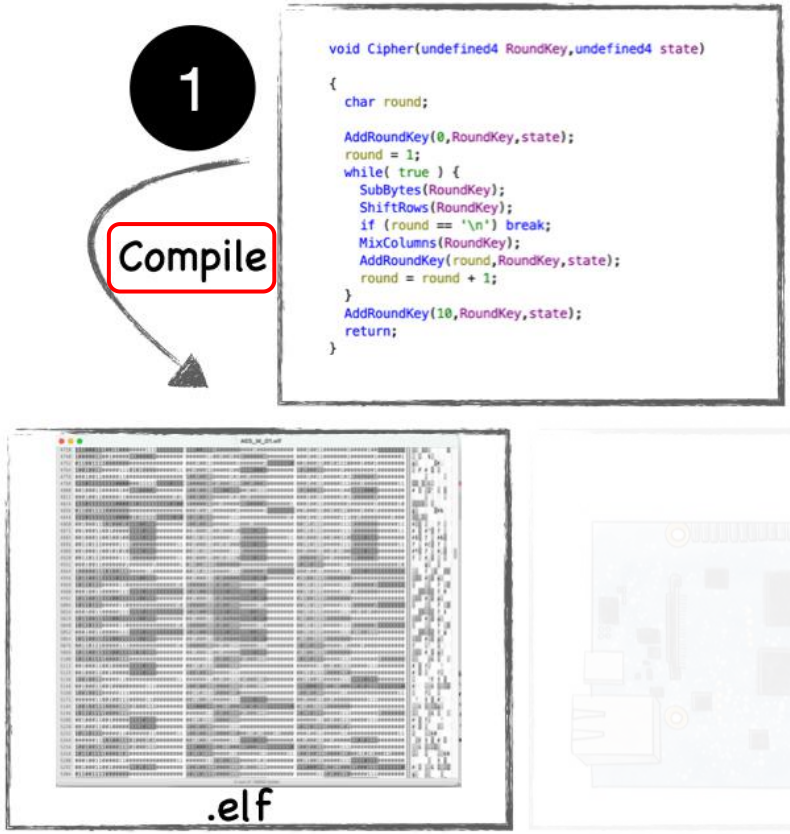


Compilation (+options):

riscv64-unknown-elf-gcc -march=rv32im -mabi=ilp32 -Os -o output input

Optimization levels: -O0 (no optimization),
-O1, -O2, -O3, (optimize for execution time)
-Os (optimize for size)

Compilation Options



Compilation (+options):

riscv64-unknown-elf-gcc -march=rv32im -mabi=ilp32 -Os -o output input

Optimization levels: -O0 (no optimization),
-O1, -O2, -O3, (optimize for execution time)
-Os (optimize for size)

Do compiler optimizations impact power side-channel leakage?

Do compiler optimizations impact power SCA?

What are the effects of compiler optimization?

RQ1

Do compiler optimizations impact power SCA?

What are the effects of compiler optimization?

Can we isolate the impact of these optimizations?

Do compiler optimizations impact power SCA?

What are the effects of compiler optimization?

How do these impact power side-channel leakage?

Can we isolate the impact of these optimizations?

Do compiler optimizations impact power SCA?

What are the effects of compiler optimization?

How do these impact power side-channel leakage?

Can we isolate the impact of these optimizations?

Do these effects propagate to real traces?

Do compiler optimizations impact power SCA?

What are the effects of compiler optimization?

How do these impact power side-channel leakage?

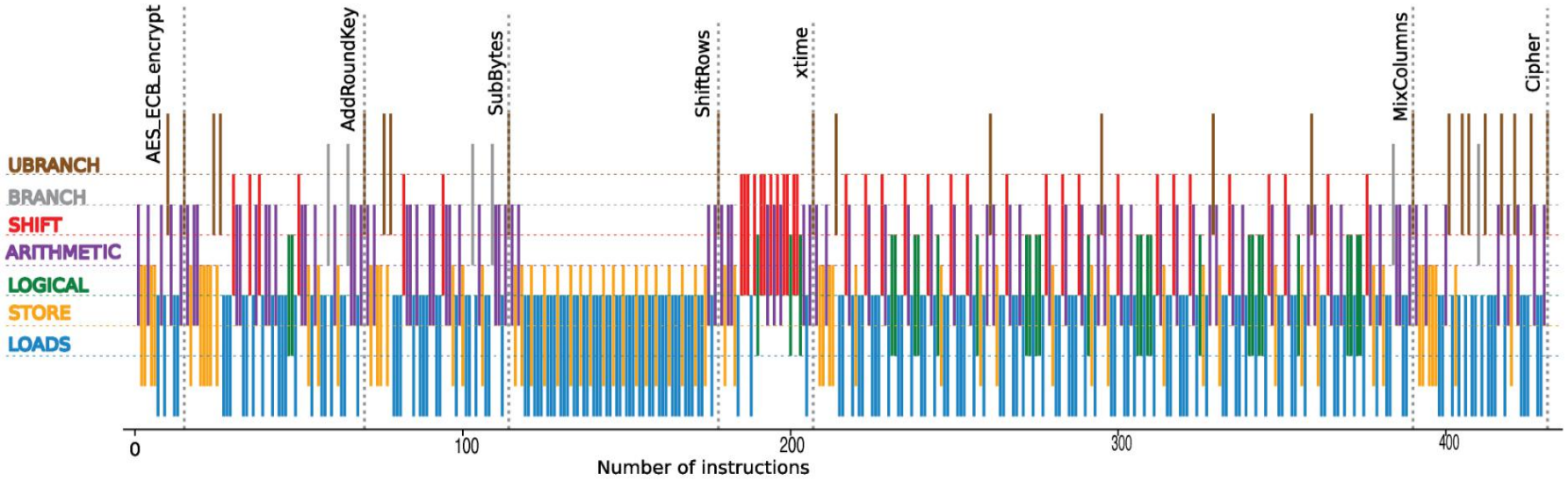
Can we predict these leaks?

Can we isolate the impact of these optimizations?

Do these effects propagate to real traces?

Compiled binary of unprotected AES

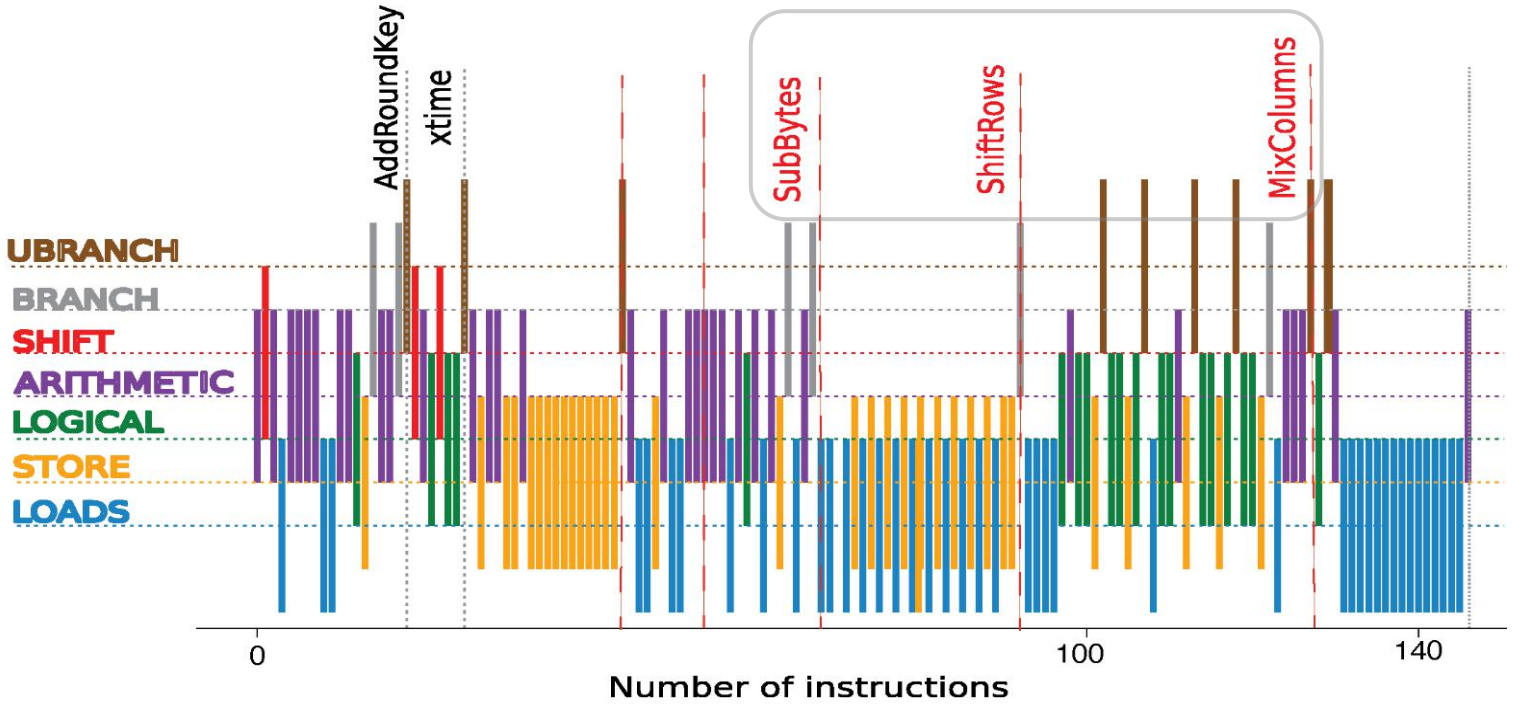
```
                    Cipher
000029a8 13 01 01 fd     addi       sp,sp,-0x30
000029ac 23 26 11 02     sw         ra,0x2c(sp)
000029b0 23 24 81 02     sw         s0,0x28(sp)
000029b4 13 04 01 03     addi       s0,sp,0x30
000029b8 23 2e a4 fc     sw         state,-0x24=>local_24(s0)
000029bc 23 2c b4 fc     sw         key,-0x28=>local_28(s0)
000029c0 a3 07 04 fe     sb         zero,-0x11=>roundVar(s0)
000029c4 03 26 84 fd     lw         a2,-0x28=>local_28(s0)
000029c8 83 25 c4 fd     lw         key,-0x24=>local_24(s0)
000029cc 13 05 00 00     mv         state,zero
000029d0 ef e0 1f d9     jal        ra,AddRoundKey
000029d4 93 07 10 00     li         a5,1
000029d8 a3 07 f4 fe     sb         a5,-0x11=>roundVar(s0)

                    LAB_000029dc
000029dc 03 25 c4 fd     lw         state,-0x24=>local_24(s0)
000029e0 ef e0 df e5     jal        ra,SubBytes
000029e4 03 25 c4 fd     lw         state,-0x24=>local_24(s0)
000029e8 ef e0 5f f0     jal        ra,ShiftRows
000029ec 03 47 f4 fe     lbu        a4,-0x11=>roundVar(s0)
000029f0 93 07 a0 00     li         a5,10
000029f4 63 08 f7 02     beq        a4,a5,LAB_00002a24
000029f8 03 25 c4 fd     lw         state,-0x24=>local_24(s0)
000029fc ef f0 4f 86     jal        ra,MixColumns
00002a00 83 47 f4 fe     lbu        a5,-0x11=>roundVar(s0)
00002a04 03 26 84 fd     lw         a2,-0x28=>local_28(s0)
00002a08 83 25 c4 fd     lw         key,-0x24=>local_24(s0)
00002a0c 13 85 07 00     mv         state,a5
00002a10 ef e0 1f d5     jal        ra,AddRoundKey
00002a14 83 47 f4 fe     lbu        a5,-0x11=>roundVar(s0)
00002a18 93 87 17 00     addi       a5,a5,1
00002a1c a3 07 f4 fe     sb         a5,-0x11=>roundVar(s0)
00002a20 6f f0 df fb     j          LAB_000029dc

                    LAB_00002a24
00002a24 13 00 00 00     nop
00002a28 03 26 84 fd     lw         a2,-0x28=>local_28(s0)
00002a2c 83 25 c4 fd     lw         key,-0x24=>local_24(s0)
00002a30 13 05 a0 00     li         state,0xa
00002a34 ef e0 df d2     jal        ra,AddRoundKey
00002a38 13 00 00 00     nop
00002a3c 83 20 c1 02     lw         ra,0x2c(sp)
00002a40 03 24 81 02     lw         s0,0x28(sp)
00002a44 13 01 01 03     addi       sp,sp,0x30
00002a48 67 80 00 00     ret
```



Optimization level: -O0

Compiled binary of unprotected AES

```
                        LAB_00000b4c
00000b4c 03 46 07 00    lbu         a2,0x0(a4)
00000b50 93 87 17 00    addi        row,row,0x1
00000b54 93 f7 f7 0f    andi        row,row,0xff
00000b58 33 86 ca 00    add         a2,s5,a2
00000b5c 03 46 06 00    lbu         a2,0x0(a2=>sbox)
00000b60 13 07 47 00    addi        a4,a4,0x4
00000b64 23 0e c7 fe    sb          a2,-0x4(a4)
00000b68 e3 92 77 ff    bne         row,s7,LAB_00000b4c
00000b6c 83 27 81 00    lw          row,0x8(sp)
00000b70 93 86 16 00    addi        statePrt,statePrt,0x1
00000b74 e3 98 d7 fc    bne         row,statePrt,LAB_00000b44
00000b78 03 47 54 00    lbu         a4,0x5(s0)
00000b7c 83 47 14 00    lbu         row,0x1(s0)
00000b80 a3 00 e4 00    sb          a4,0x1(s0)
00000b84 03 47 94 00    lbu         a4,0x9(s0)
00000b88 a3 02 e4 00    sb          a4,0x5(s0)
00000b8c 03 47 d4 00    lbu         a4,13(s0)
00000b90 a3 06 f4 00    sb          row,0xd(s0)
00000b94 83 47 24 00    lbu         row,0x2(s0)
00000b98 a3 04 e4 00    sb          a4,0x9(s0)
00000b9c 03 47 a4 00    lbu         a4,0xa(s0)
00000ba0 23 05 f4 00    sb          row,0xa(s0)
00000ba4 83 47 64 00    lbu         row,0x6(s0)
00000ba8 23 01 e4 00    sb          a4,0x2(s0)
00000bac 03 47 e4 00    lbu         a4,0xe(s0)
00000bb0 23 07 f4 00    sb          row,0xe(s0)
00000bb4 83 47 34 00    lbu         row,0x3(s0)
00000bb8 23 03 e4 00    sb          a4,0x6(s0)
00000bbc 03 47 f4 00    lbu         a4,0xf(s0)
00000bc0 a3 01 e4 00    sb          a4,0x3(s0)
00000bc4 03 47 b4 00    lbu         a4,0xb(s0)
00000bc8 a3 07 e4 00    sb          a4,0xf(s0)
00000bcc 03 47 74 00    lbu         a4,0x7(s0)
00000bd0 a3 03 f4 00    sb          row,0x7(s0)
00000bd4 a3 05 e4 00    sb          a4,0xb(s0)
00000bd8 63 8c 8d 09    beq         round,s8,LAB_00000c70

                        LAB_00000bdc
00000bdc 03 cb 04 00    lbu         s6,0x0(s1)
00000be0 03 cd 14 00    lbu         s10,0x1(s1)
00000be4 03 c9 24 00    lbu         s2,0x2(s1)
00000be8 03 ca 34 00    lbu         s4,0x3(s1)
00000bec 33 45 ab 01    xor         key,s6,s10
```



Optimization level: -Os

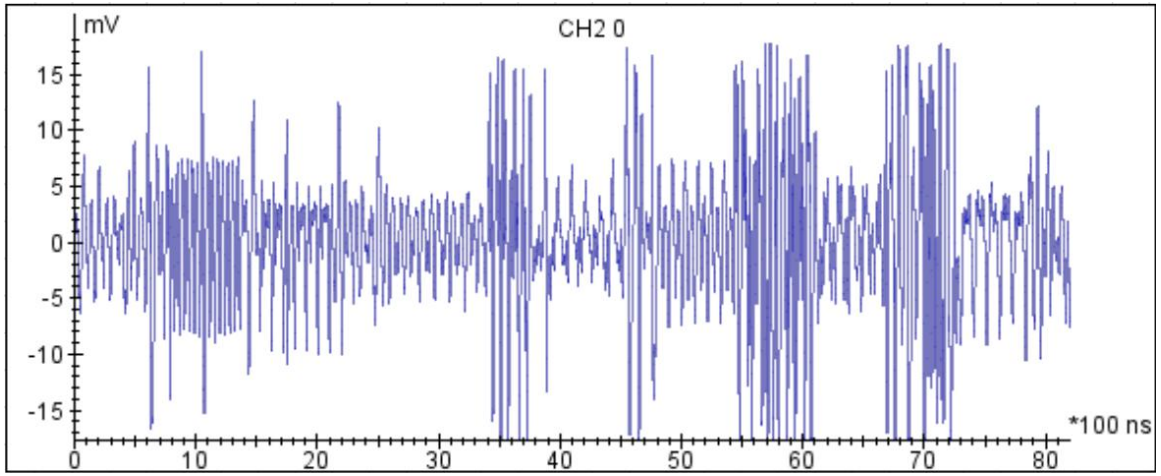
Effects on compiled binary

Optimization Level	Strength Reduction	Instruction Rescheduling	Loop Unrolling	Loop Fusion	Function Inlining	Constant Folding
-O0	○	○	○	○	○	○
-Os	●	●	○	○	◐	○
-O1	●	●	○	○	◐	○
-O2	●	●	○	○	●	●
-O3	●	●	●	●	●	●

Optimization impact on compiled binary for unprotected AES

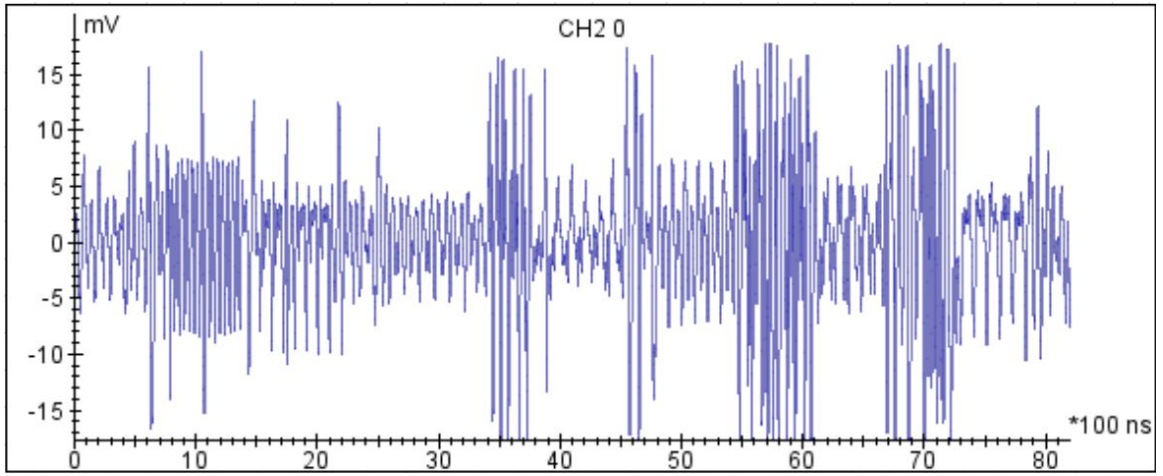
CMOS Power Consumption Model

$$P_{total} = P_{op} + P_{data} + P_{noise} + P_{const}$$



CMOS Power Consumption Model

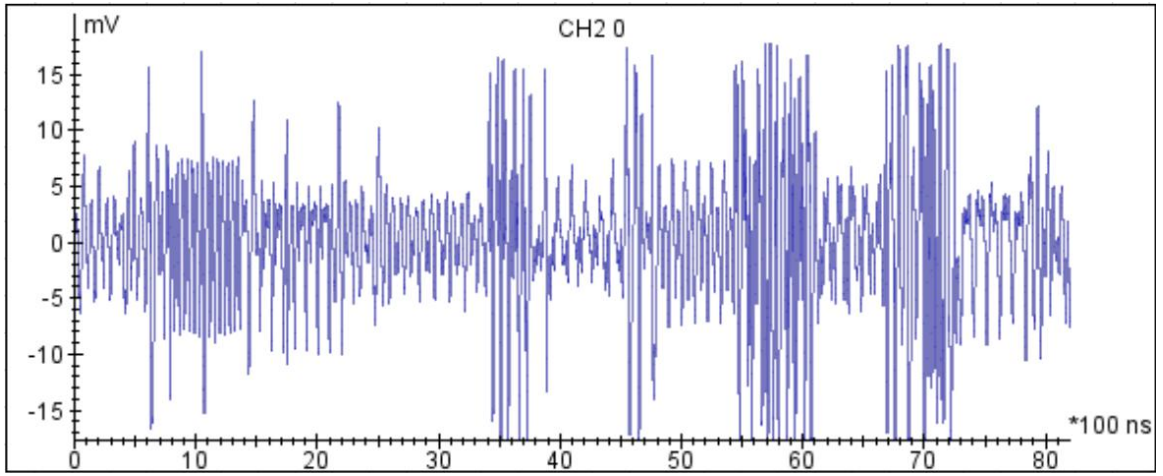
$$P_{total} = \underbrace{P^{A}_{op} + P^{M}_{op}}_{P_{op}} + P_{data} + P_{noise} + P_{const}$$



CMOS Power Consumption Model

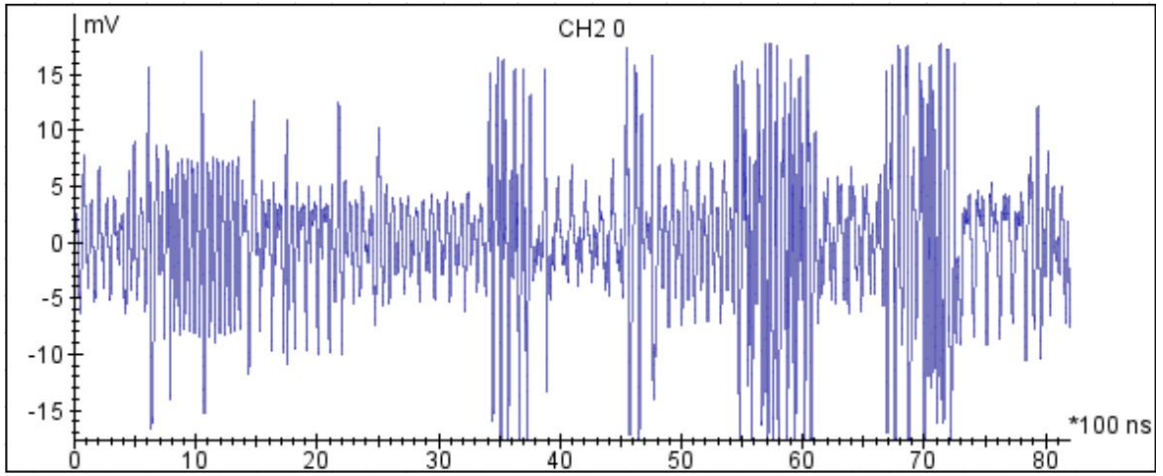
$$P_{total} = P^{M}_{op} + \underbrace{P^{A}_{op} + P_{data}} + P_{noise} + P_{const}$$

(Architectural component)



CMOS Power Consumption Model

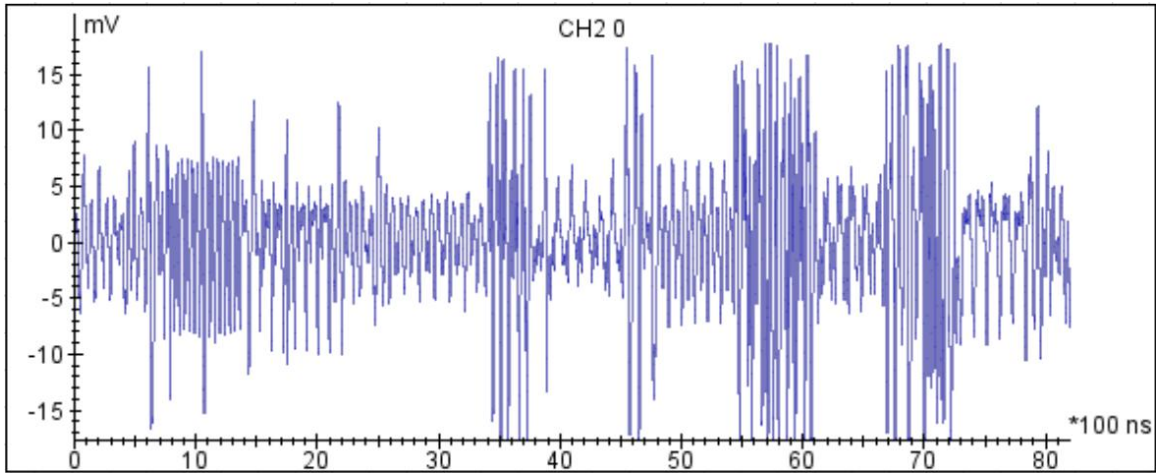
$$P_{total} = P^{M}_{op} + \underbrace{P^{A}_{op} + P_{data}}_{\text{(Architectural component)}} + P_{noise} + P_{const}$$



Can we isolate the impact of these optimizations?

CMOS Power Consumption Model

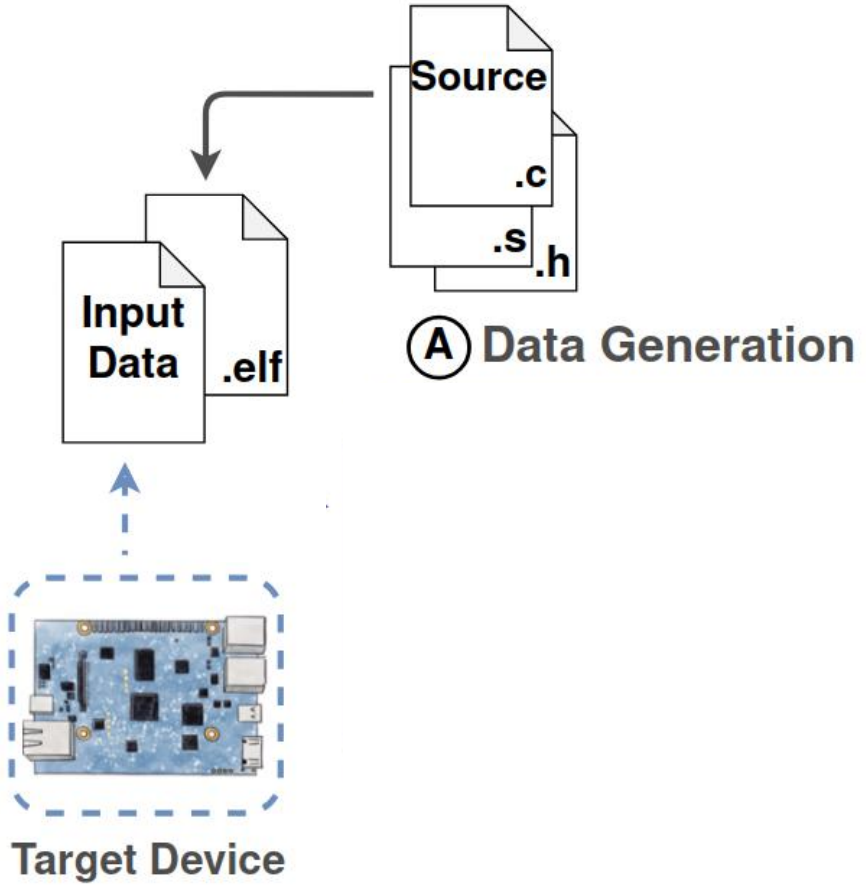
$$P_{total} = P^{M}_{op} + \underbrace{P^{A}_{op} + P_{data}}_{\text{(Architectural component)}} + P_{noise} + P_{const}$$



Can we isolate the impact of these optimizations?

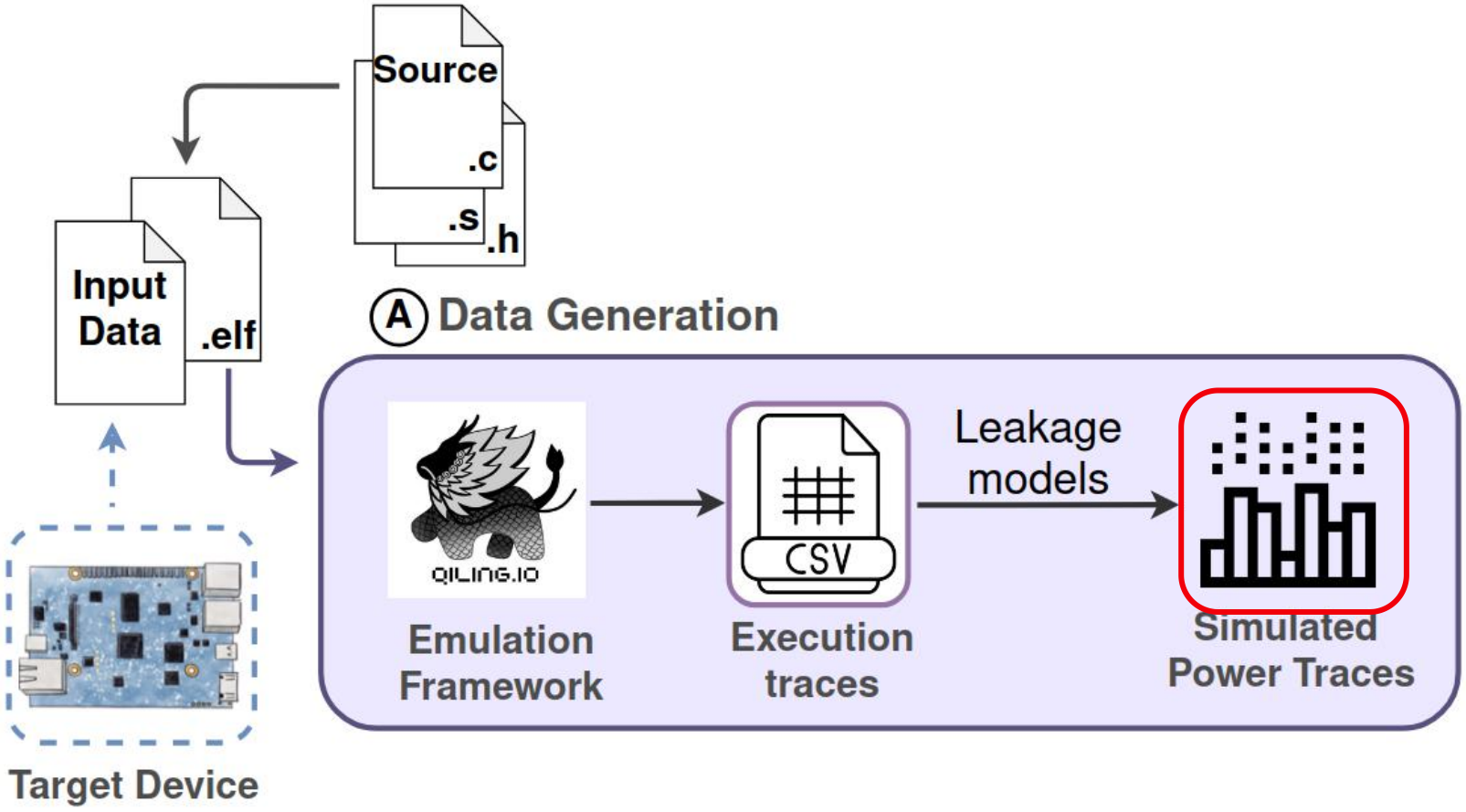
Yes! Architecture-level simulation

ARCHER: Architecture-Level Simulator



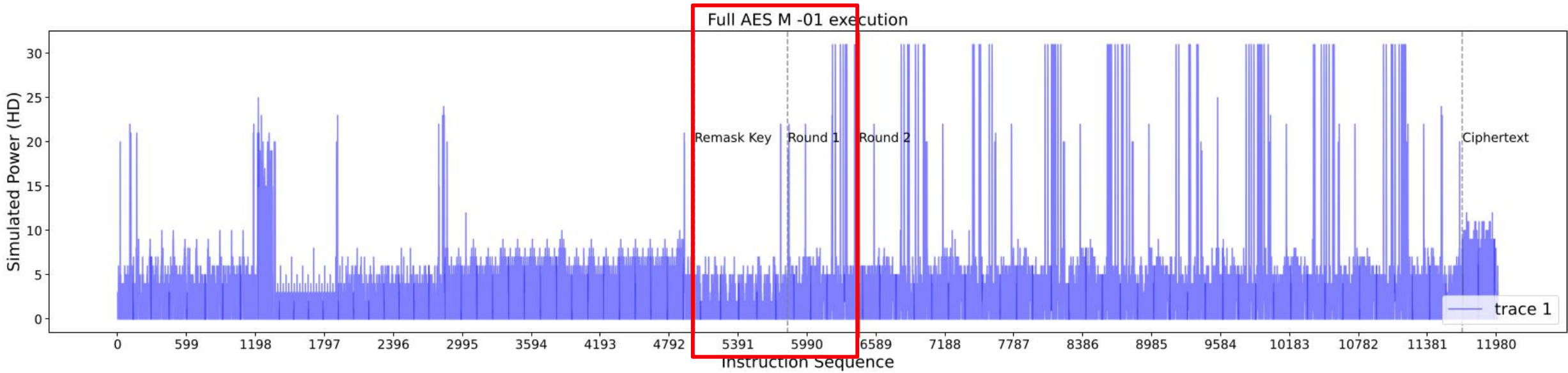
"ARCHER: Architecture-Level Simulator for Side-Channel Analysis in RISC-V Processors." Adhikary, Asmita, Abraham J. Basurto Becerra, Lejla Batina, Ileana Buhan, Durba Chatterjee, Senna Van Hoek, and Eloi Sanfelix Gonzalez. Cryptology ePrint Archive (2024).

ARCHER: Architecture-Level Simulator

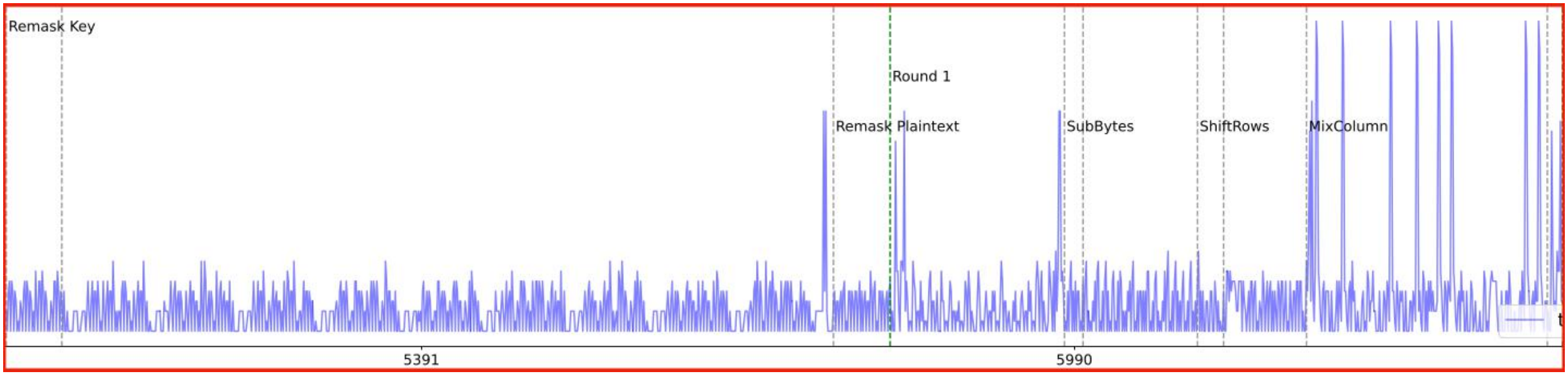


"ARCHER: Architecture-Level Simulator for Side-Channel Analysis in RISC-V Processors." Adhikary, Asmita, Abraham J. Basurto Becerra, Lejla Batina, Ileana Buhan, Durba Chatterjee, Senna Van Hoek, and Eloi Sanfelix Gonzalez. Cryptology ePrint Archive (2024).

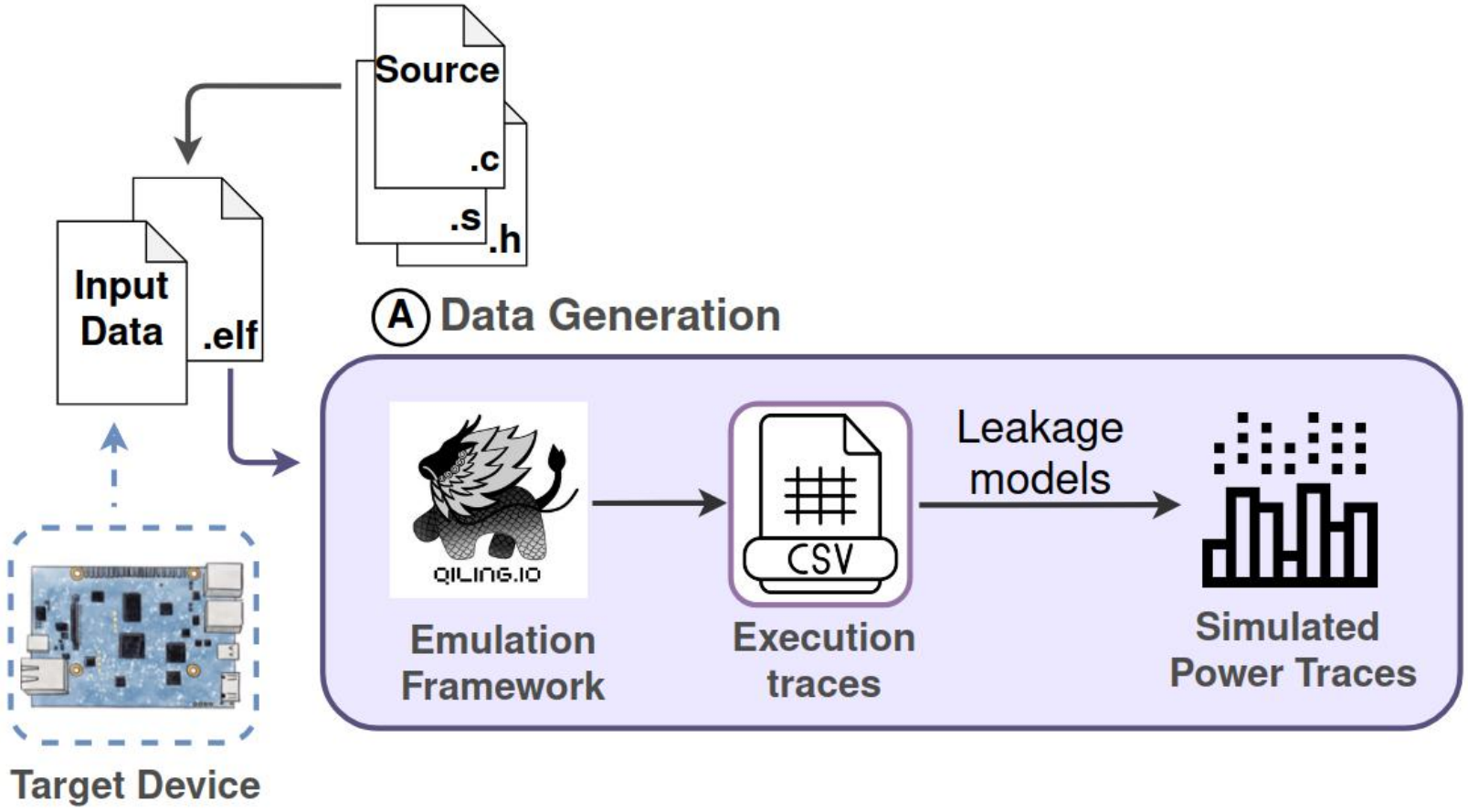
Simulated Power Trace of Masked AES



Simulated Power Trace of Masked AES

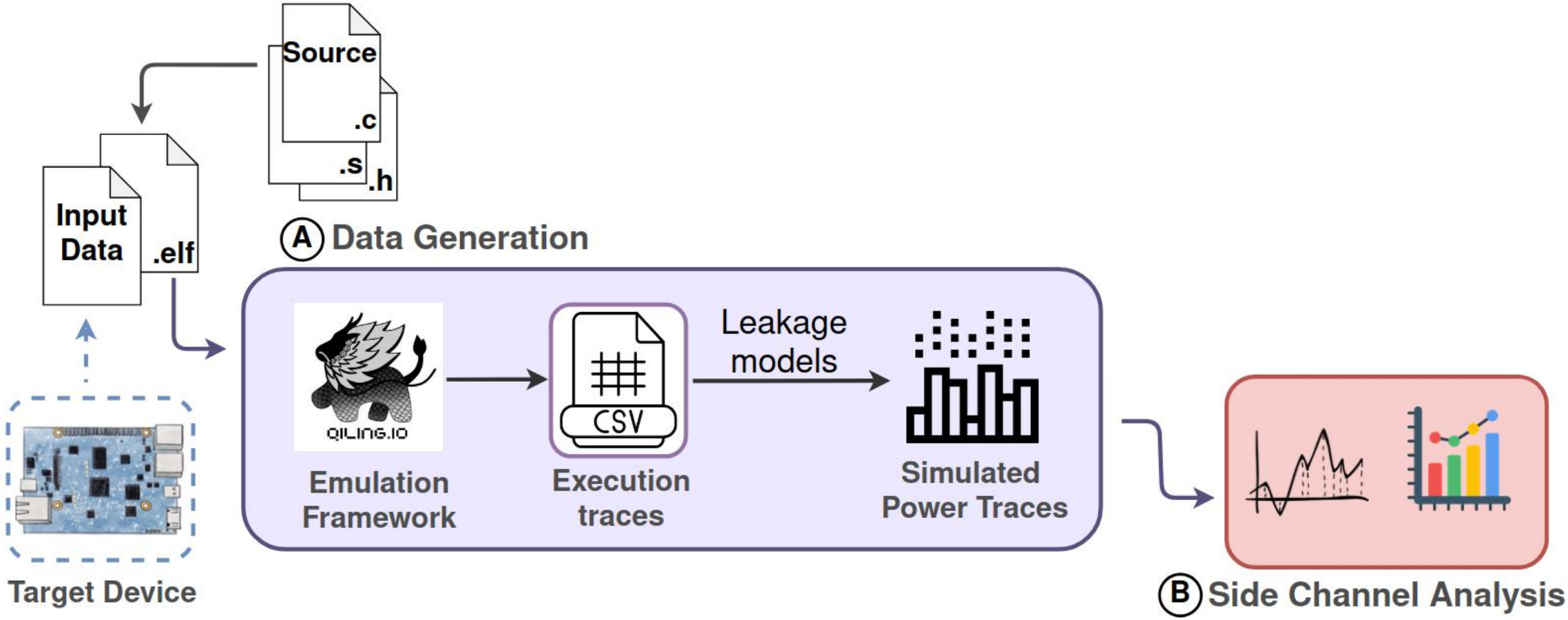


ARCHER: Architecture-Level Simulator



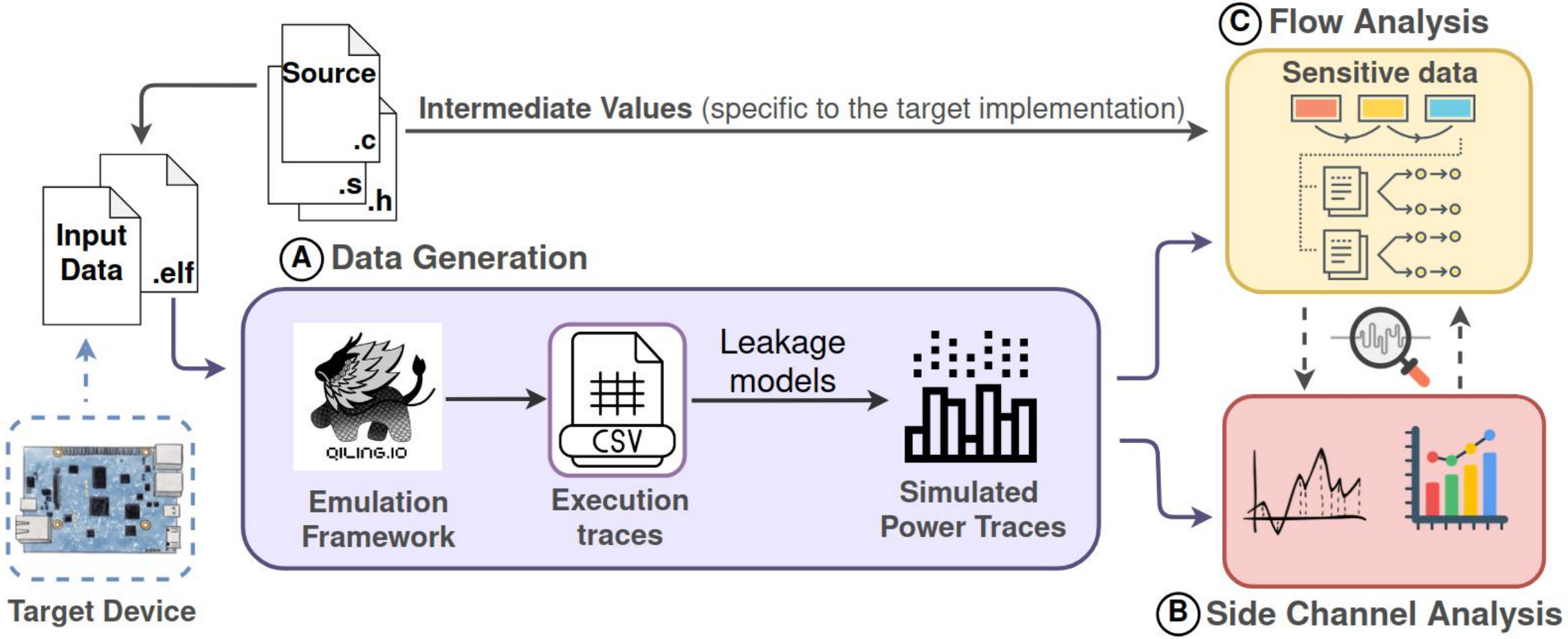
"ARCHER: Architecture-Level Simulator for Side-Channel Analysis in RISC-V Processors." Adhikary, Asmita, Abraham J. Basurto Becerra, Lejla Batina, Ileana Buhan, Durba Chatterjee, Senna Van Hoek, and Eloi Sanfelix Gonzalez. Cryptology ePrint Archive (2024).

ARCHER: Architecture-Level Simulator



"ARCHER: Architecture-Level Simulator for Side-Channel Analysis in RISC-V Processors." Adhikary, Asmita, Abraham J. Basurto Becerra, Lejla Batina, Ileana Buhan, Durba Chatterjee, Senna Van Hoek, and Eloi Sanfelix Gonzalez. Cryptology ePrint Archive (2024).

ARCHER: Architecture-Level Simulator



"ARCHER: Architecture-Level Simulator for Side-Channel Analysis in RISC-V Processors." Adhikary, Asmita, Abraham J. Basurto Becerra, Lejla Batina, Ileana Buhan, Durba Chatterjee, Senna Van Hoek, and Eloi Sanfelix Gonzalez. Cryptology ePrint Archive (2024).

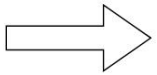
ShiftRows Operation of Masked AES

s_0^m	s_4^m	s_8^m	s_{12}^m
s_1^m	s_5^m	s_9^m	s_{13}^m
s_2^m	s_6^m	s_{10}^m	s_{14}^m
s_3^m	s_7^m	s_{11}^m	s_{15}^m

State Matrix

ShiftRows Operation of Masked AES

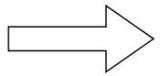
s_0^m	s_4^m	s_8^m	s_{12}^m
s_1^m	s_5^m	s_9^m	s_{13}^m
s_2^m	s_6^m	s_{10}^m	s_{14}^m
s_3^m	s_7^m	s_{11}^m	s_{15}^m



s_0^m	s_4^m	s_8^m	s_{12}^m
s_5^m	s_9^m	s_{13}^m	s_1^m
s_{10}^m	s_{14}^m	s_2^m	s_6^m
s_{15}^m	s_3^m	s_7^m	s_{11}^m

ShiftRows Operation of Masked AES

s_0^m	s_4^m	s_8^m	s_{12}^m
s_1^m	s_5^m	s_9^m	s_{13}^m
s_2^m	s_6^m	s_{10}^m	s_{14}^m
s_3^m	s_7^m	s_{11}^m	s_{15}^m



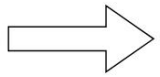
s_0^m	s_4^m	s_8^m	s_{12}^m
s_5^m	s_9^m	s_{13}^m	s_1^m
s_{10}^m	s_{14}^m	s_2^m	s_6^m
s_{15}^m	s_3^m	s_7^m	s_{11}^m

PC	d90	d94	d98	d9c	da0	da4	da8
a4	s_5^m			s_9^m		s_{13}^m	
a5		s_1^m					

Assembly implementation for level -Os

ShiftRows Operation of Masked AES

s_0^m	s_4^m	s_8^m	s_{12}^m
s_1^m	s_5^m	s_9^m	s_{13}^m
s_2^m	s_6^m	s_{10}^m	s_{14}^m
s_3^m	s_7^m	s_{11}^m	s_{15}^m



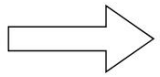
s_0^m	s_4^m	s_8^m	s_{12}^m
s_5^m	s_9^m	s_{13}^m	s_1^m
s_{10}^m	s_{14}^m	s_2^m	s_6^m
s_{15}^m	s_3^m	s_7^m	s_{11}^m

PC	d90	d94	d98	d9c	da0	da4	da8
a4	s_5^m			s_9^m		s_{13}^m	
a5		s_1^m					

Assembly implementation for level -Os

ShiftRows Operation of Masked AES

s_0^m	s_4^m	s_8^m	s_{12}^m
s_1^m	s_5^m	s_9^m	s_{13}^m
s_2^m	s_6^m	s_{10}^m	s_{14}^m
s_3^m	s_7^m	s_{11}^m	s_{15}^m



s_0^m	s_4^m	s_8^m	s_{12}^m
s_5^m	s_9^m	s_{13}^m	s_1^m
s_{10}^m	s_{14}^m	s_2^m	s_6^m
s_{15}^m	s_3^m	s_7^m	s_{11}^m

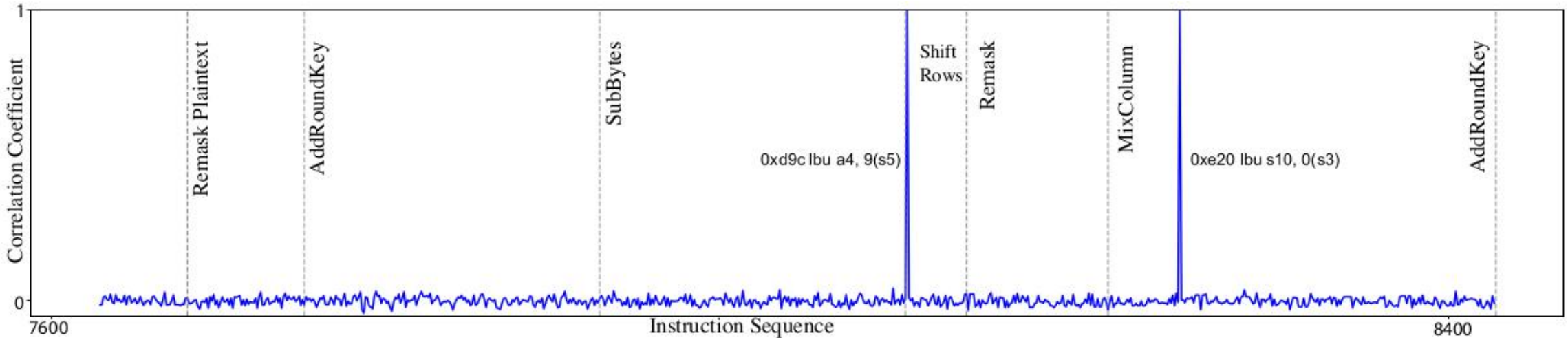
$$s_5^m = \mathrm{SBox}(p_5 \oplus k_5) \oplus m$$
$$s_9^m = \mathrm{SBox}(p_9 \oplus k_9) \oplus m$$

$$s_5^m \oplus s_9^m = \mathrm{SBox}(p_5 \oplus k_5) \oplus \mathrm{SBox}(p_9 \oplus k_9)$$

PC	d90	d94	d98	d9c	da0	da4	da8
a4	s_5^m			s_9^m		s_{13}^m	
a5		s_1^m					

Assembly implementation for level -Os

Correlation Analysis on simulated traces

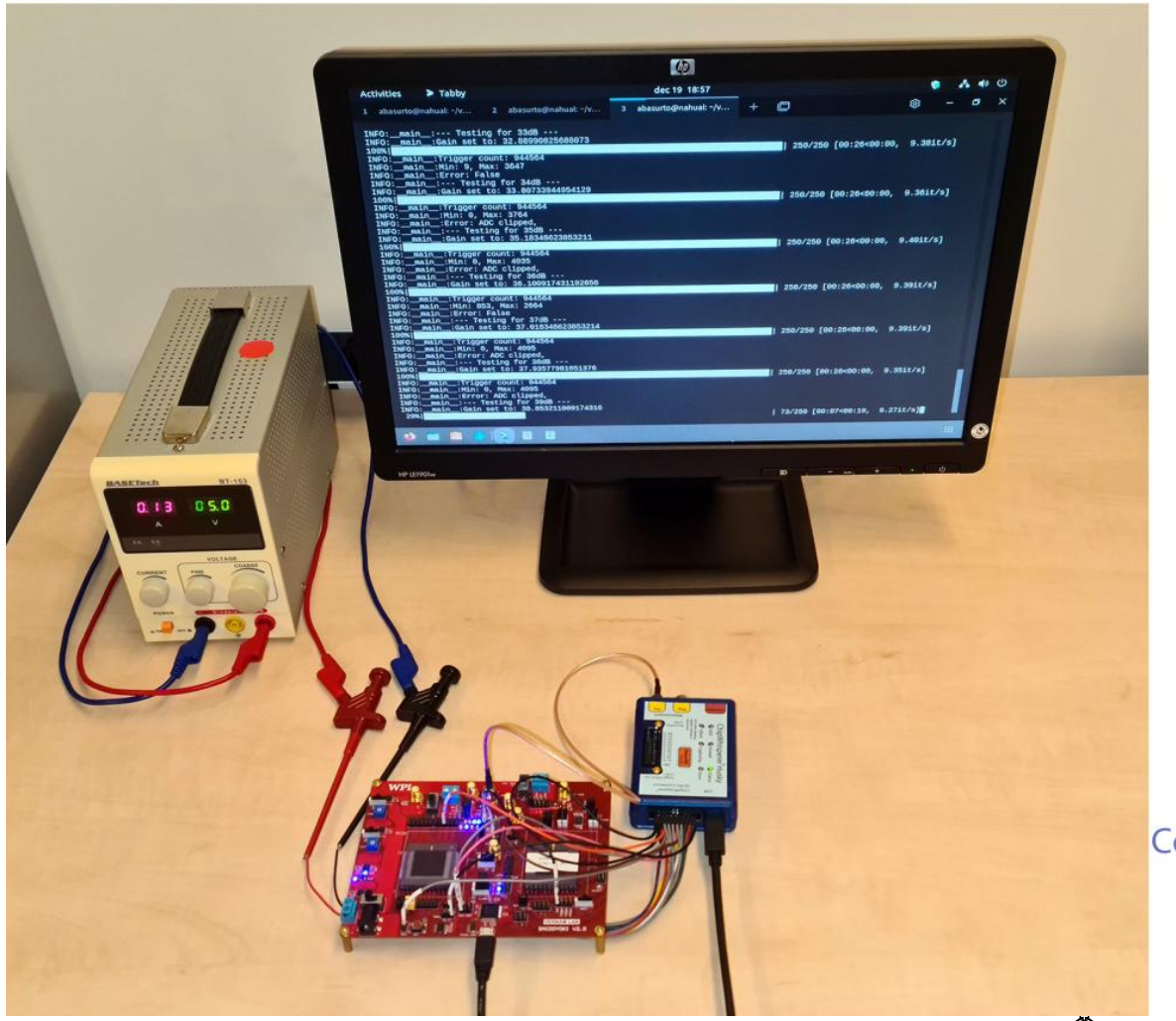


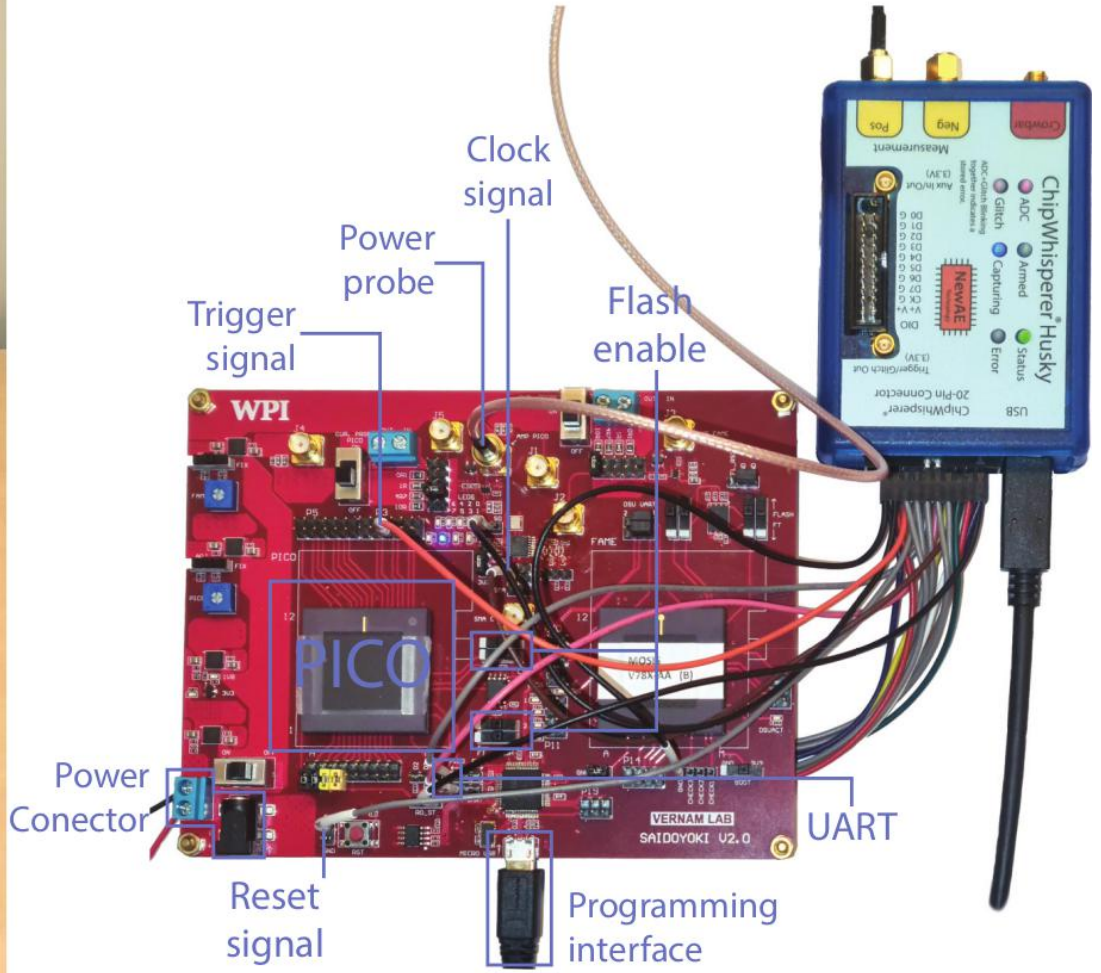
Level: -Os

Correlated with $\mathrm{SBox}(p_5 \oplus k_5) \oplus \mathrm{SBox}(p_9 \oplus k_9)$
over 5000 simulated traces where plaintext byte 9 is randomly selected.

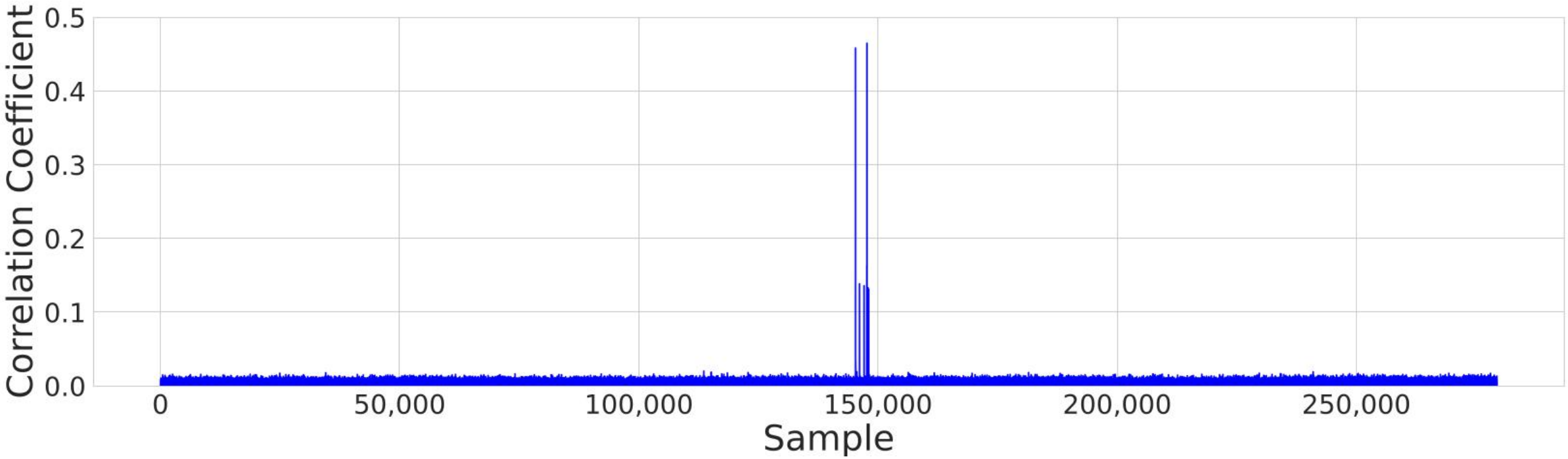
Overwriting masked shares at architecture level reveals unmaksed variables

Experimental Setup





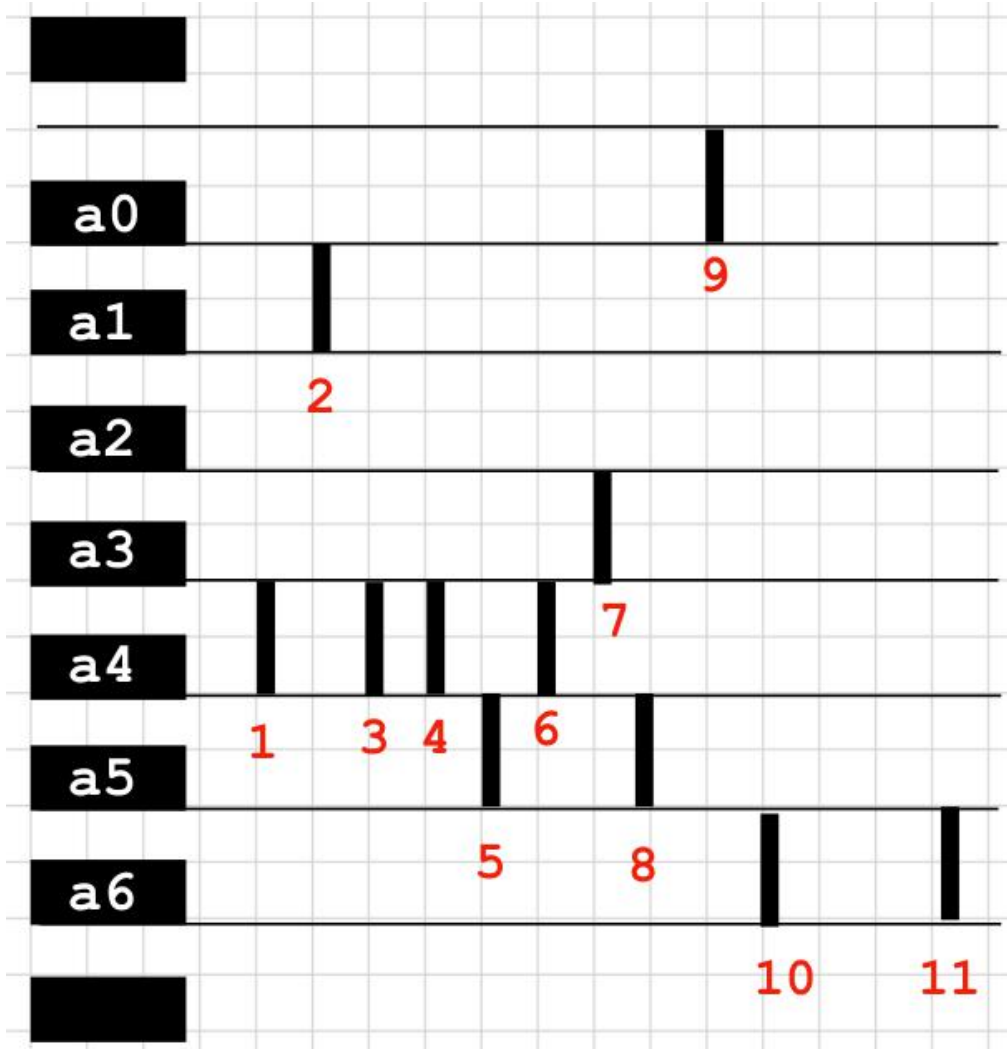
Does leakage propagate to real power traces?



Level: -Os

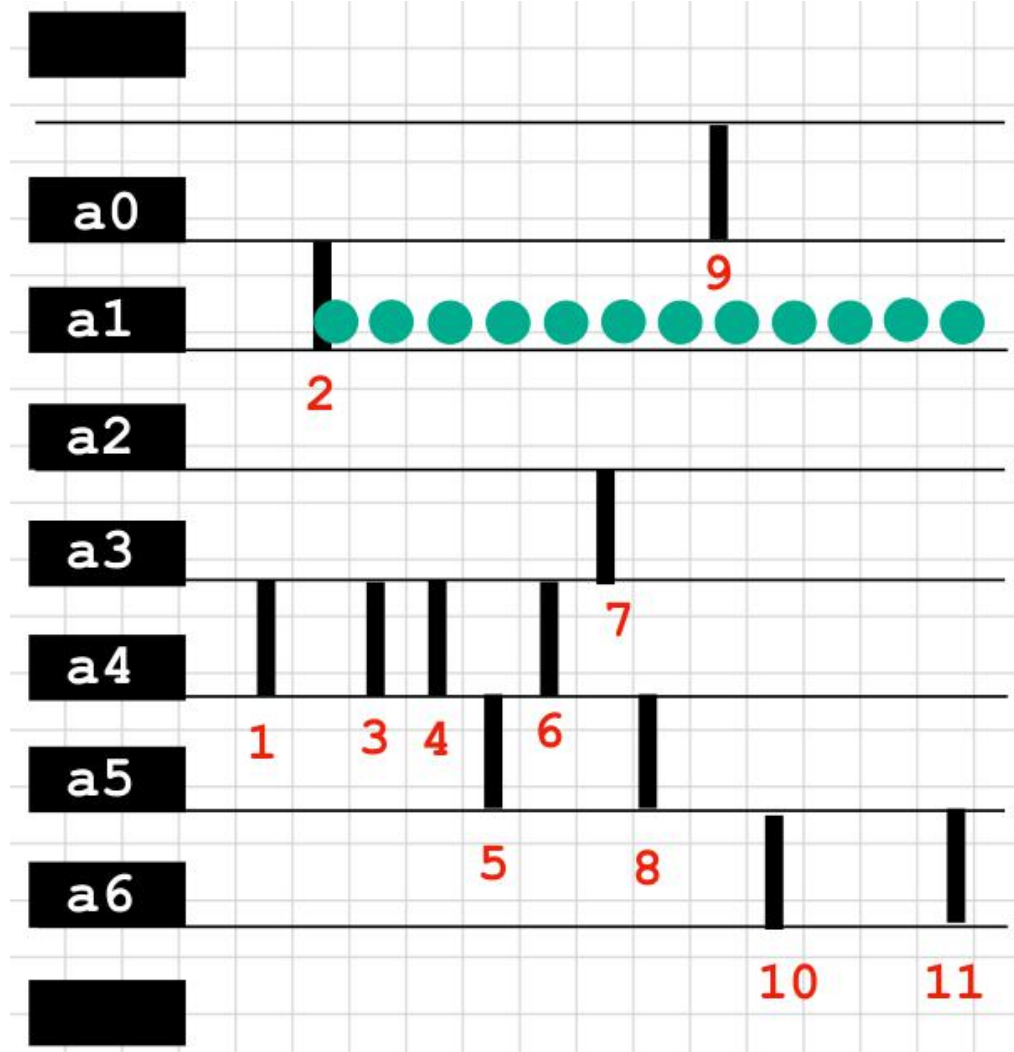
Yes! Architectutal leaks visible in real traces.

Predicting Leakage



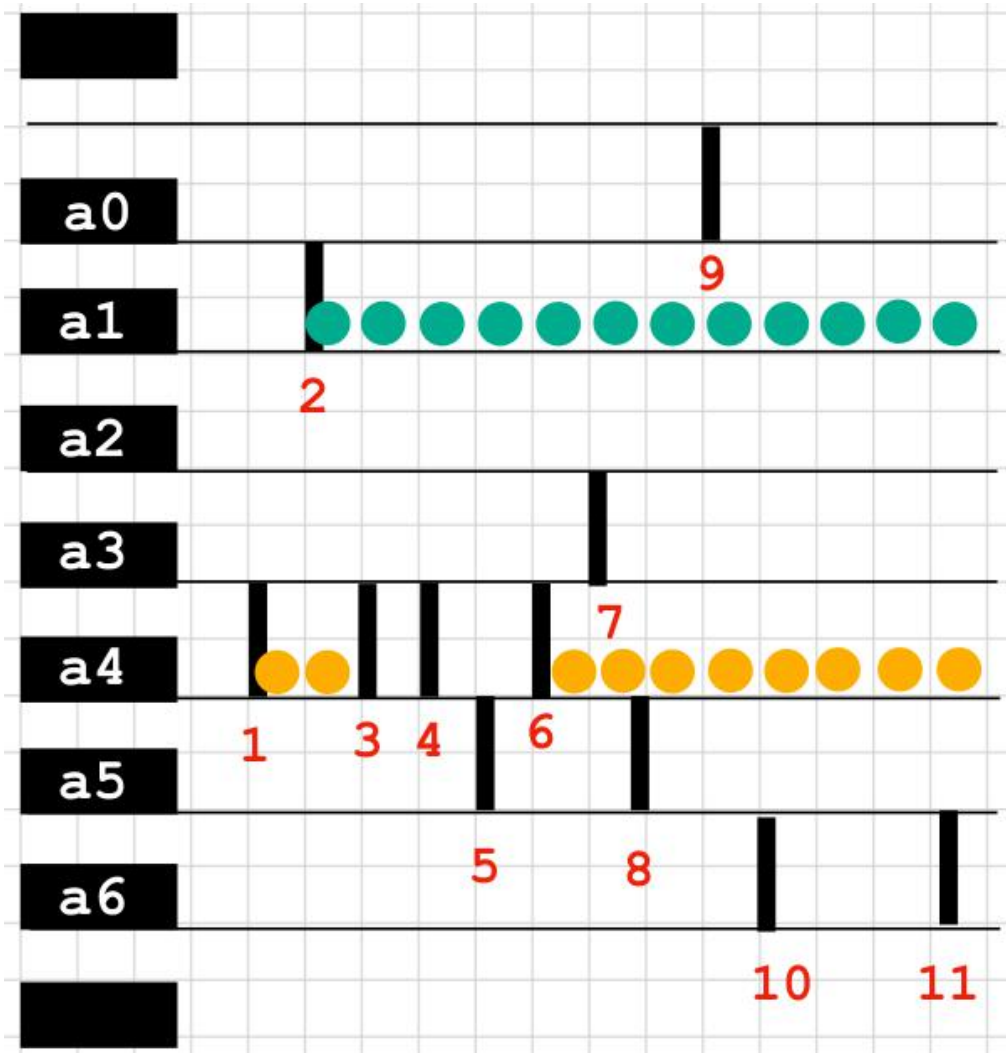

```
1   lbu   a4,0x0(a5)
2   lbu   a1,0x0(a3)
3   xor   a4,a4,a1
4   sb    a4,0x0(a5)
5   addi  a5,a5,0x1
6   lbu   a4,0x0(a5)
7   addi  a3,a3,0x1
8   mov   a5,a6
9   addi  a0,a0,0x4
10  addi  a6,a6,0x4
11  lbu   a6,0x0(a5)
```

Remanence



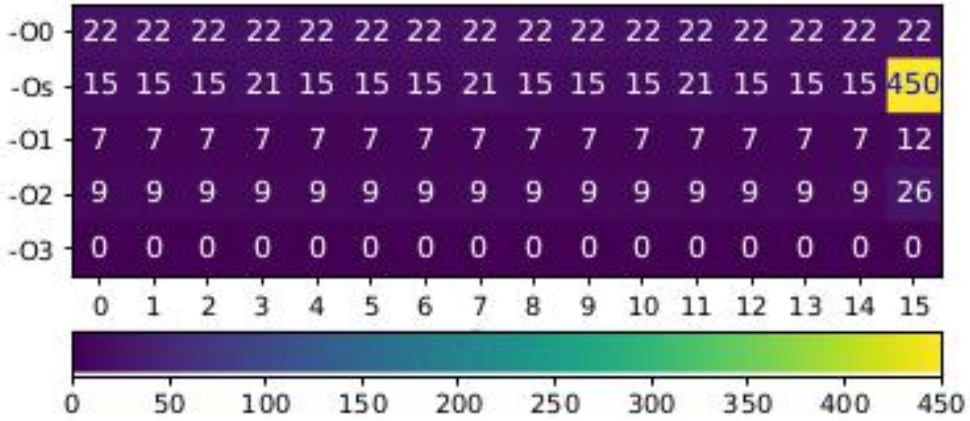

```
1   lbu   a4,0x0(a5)
2   lbu   a1,0x0(a3)
3   xor   a4,a4,a1
4   sb    a4,0x0(a5)
5   addi  a5,a5,0x1
6   lbu   a4,0x0(a5)
7   addi  a3,a3,0x1
8   mov   a5,a6
9   addi  a0,a0,0x4
10  addi  a6,a6,0x4
11  lbu   a6,0x0(a5)
```

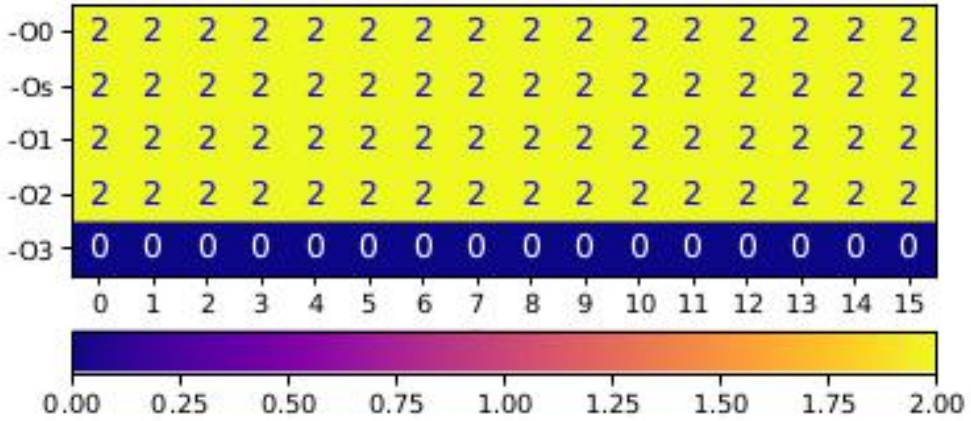
Remanence and Revive




```
1   lbu   a4,0x0(a5)
2   lbu   a1,0x0(a3)
3   xor   a4,a4,a1
4   sb    a4,0x10(a5)
5   addi  a5,a5,0x1
6   lbu   a4,0x0(a5)
7   addi  a3,a3,0x1
8   mov   a5,a6
9   addi  a0,a0,0x4
10  addi  a6,a6,0x4
11  lbu   a6,0x0(a5)
```

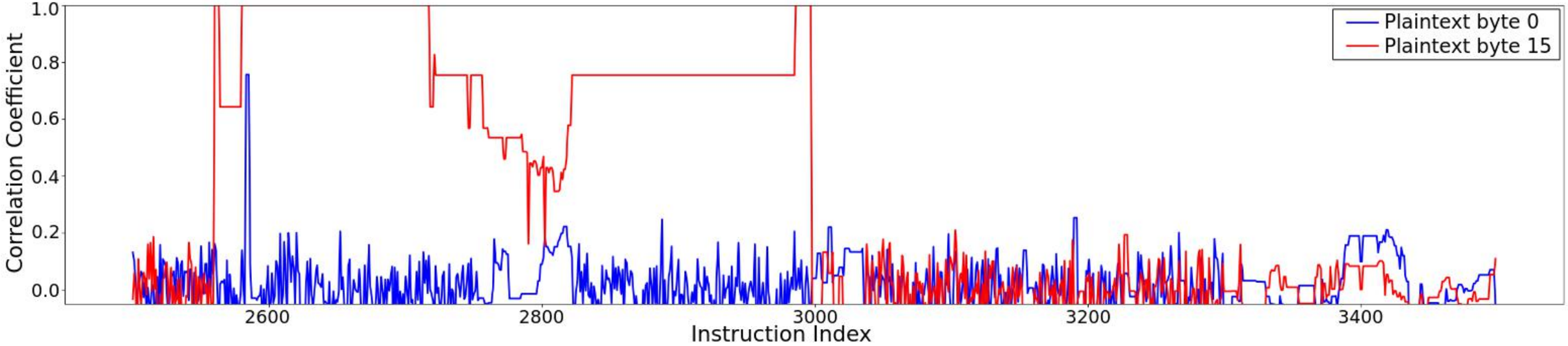
Predicting Leakage





Plaintext Remanence

Plaintext Revive



THANK YOU
QUESTIONS?