Formally Proved Security of Assembly Code Against Leakage

Pablo RAUZY
Sylvain GUILLEY

Institut MINES-TELECOM,
TELECOM-ParisTech,
CNRS LTCI (UMR 5141).
Paris, France

Context: countermeasures

	Hardware	Software
Masking	***	***
Hiding (dual-rail)	***	*few works!*

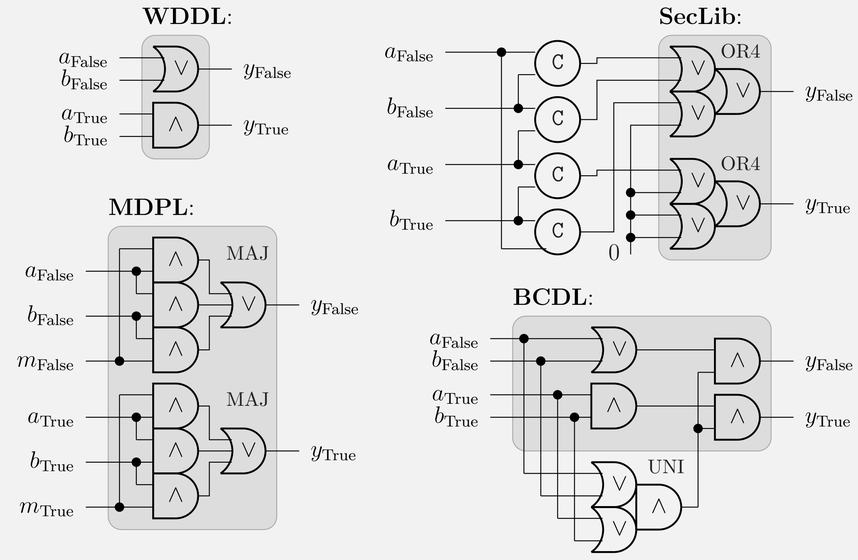
Problems of masking in software

- Lots of entropy (*not available on resource-constrained devices*)
- Structural vulnerability: existence *high-order* attacks

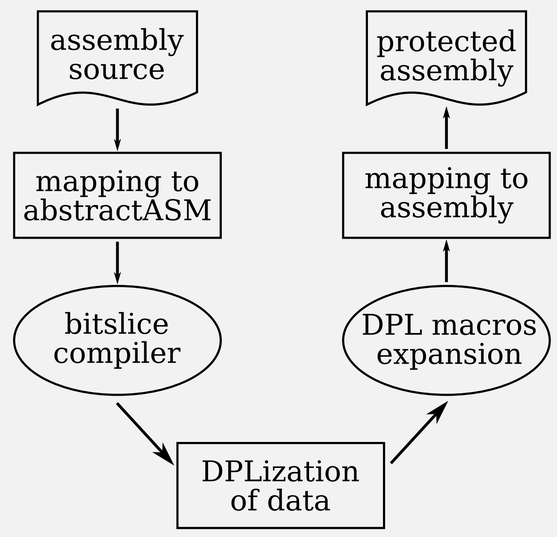
Dual-rail in software: opportunities

- No need for entropy
- Provable correction of leakage-free (with a finite number of *physical* hypotheses, to do by pre-characterization)

State-of-the-art about dual-rail in hardware [DGBN09]



Pure software dual-rail: design flow



DPL: **D**ual-**R**ail with **P**recharge

Macro for Boolean operation *op*

```
r1 ← r0          mov r1 r0
r1 ← a           mov r1 a
r1 ← r1 ∧ 3      and r1 r1 #3
r1 ← r1 ≪ 1      shl r1 r1 #1
r1 ← r1 ≪ 1      shl r1 r1 #1
r2 ← r0          mov r2 r0
r2 ← b           mov r2 b
r2 ← r2 ∧ 3      and r2 r2 #3
r1 ← r1 ∨ r2     orr r1 r1 r2
r3 ← r0          mov r3 r0
r3 ← op[r1]      mov r3 !r1,op
 d ← r0          mov d r0
 d ← r3          mov d r3
```

Cost on PRESENT [BKL+07] case-study

	cycle count	code size*	RAM words*
state-of-the-art	11342	1000	18
bitsliced	6473	1194	144
DPL protected	182572	2674	192

* The state-of-the-art code size and RAM words are given for encryption + decryption, while ours are for encryption only. Code size and RAM words are given in bytes.

Optimizations (still with formal proof of correction)

- The existence of non-sensitive signals (*e.g.*, the selection of key size); or loop counters;
- The limited data range of some variables, that makes some parts of the code use constant variables;
- The possibility to go from one macro to the other through register, thereby saving time from the memory transfers;
- The possibility to merge instructions given certain patterns;
- The use of architecture-specific intructions not included in our abstractASM.

State-of-the-art: mixed HW/SW, *e.g.*, dual-rail instruction set [CSS13]

Original C program

```
unsigned char in, key;
unsigned char out;
unsigned temp;
unsigned char sbox[256] = {0x63, ...};
      ...
temp = in ^ key;
out = sbox[temp];
      ...
```

(a)

Compile

Original Assembly

```
; r6 = in
; r7 = key
; r3 = sbox = 0xXX00
; r4 = &out
      ...
xor   r8, r7, r6   ; add key
lbzx  r3, r5, r8   ; sbox lookup
stb   r3, 0(r4)    ; store out
      ...
```

(b)

Transform

Program

```
; r6 = in, r7 = key,
; r0 = 0,  r11 = 0xff,
; r12 = 0xXX00,
; (r12) = 0          don't care
; xor conversion
1.  xor  r9,r0,r0
2.  xor  r9,r7,r11
3.  xor  r10,r0,r0
4.  xor  r10,r6,r11
5.  and  r11,r0,r0
6.  and  r11,r9,r6
7.  and  r9,r0,r0
8.  and  r9,r10,r7
9.  or   r8,r0,r0
10. or   r8,r9,r11
; lbzx conversion
11. lwzx r3,r12,r0
12. lbzx r3,r5,r8
13. lwzx r9,r12,r0
; stb conversion
14. stwx r9,r4,r0
15. stb  r3,0(r4)
16. stwx r9,r12,r0
17. lwzx r9,r12,r0
```

Program

```
; r6 = in, r7 = key,
; r0 = 0,  r11 = 0xff,
; r12 = 0xXX00,
; (r12) = 0.
; xor conversion
1.  xor  r9,r0,r0
2.  xor  r9,r7,r11
3.  xor  r10,r0,r0
4.  xor  r10,r6,r11
5.  or   r11,r0,r0
6.  or   r11,r9,r6
7.  or   r9,r0,r0
8.  or   r9,r10,r7
9.  and  r8,r0,r0
10. and  r8,r9,r11
; lbzx conversion
11. lwzx r3,r12,r0
12. lbzx r3,r5,r8
13. lwzx r9,r12,r0
; stb conversion
14. stwx r9,r4,r0
15. stb  r3,0(r4)
16. stwx r9,r12,r0
17. lwzx r9,r12,r0
```

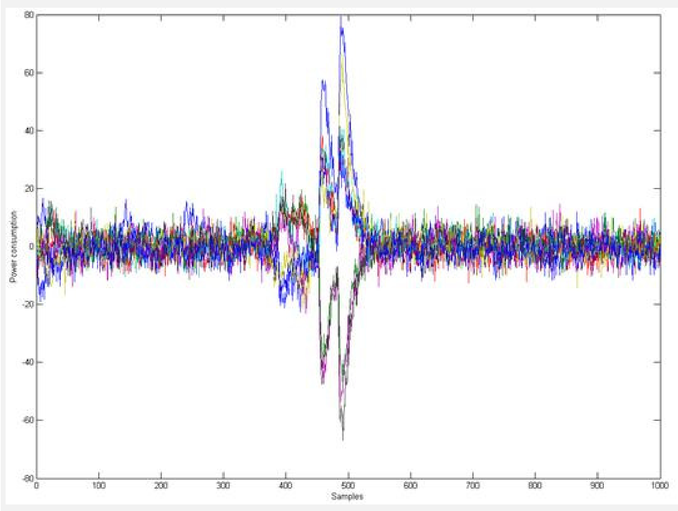
(c)

Courtesy of Zhimin Chen and Patrick Schaumont ECE Department, Virginia Tech Blacksburg VA 24061, USA

An example of Virtual Secure Circuit (VSC):

(a) `KeyAddition` and `SubBytes` operations in C code;
(b) Compiled assembly code;
(c) Converted VSC assembly code.

Leakage analysis (physical part)

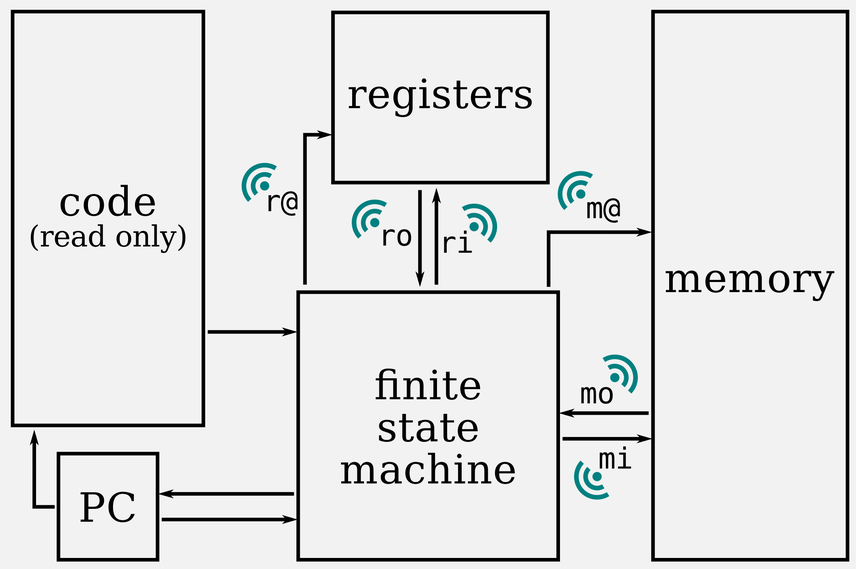


Verifications:
- indistinguishable resources
- for data and addresses

Tools:
- profiling
- linear regression

Stochastic characterization [SLP05] of every bit in a general purpose CPU.

Leakage analysis (formal part)



Verification:

- Leakage: **Hamming distance** of values, **should be constant**
- **Symbolic execution** to check this constantness property

References

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *CHES*, volume 4727 of *LNCS*, pages 450–466. Springer, September 10-13 2007. Vienna, Austria.

[CSS13] Zhimin Chen, Ambuj Sinha, and Patrick Schaumont. Using Virtual Secure Circuit to Protect Embedded Software from Side-Channel Attacks. *IEEE Trans. Computers*, 62(1):124–136, 2013.

[DGBN09] Jean-Luc Danger, Sylvain Guilley, Shivam Bhasin, and Maxime Nassar. Overview of Dual Rail with Precharge Logic Styles to Thwart Implementation-Level Attacks on Hardware Cryptoprocessors, — *New Attacks and Improved Counter-Measures* —. In *SCS*, IEEE, pages 1–8, November 6–8 2009. Jerba, Tunisia. DOI: 10.1109/ICSCS.2009.5412599.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In LNCS, editor, *CHES*, volume 3659 of *LNCS*, pages 30–46. Springer, Sept 2005. Edinburgh, Scotland, UK.