WHIP THE WHISPERER: SIMULATING SIDE CHANNEL LEAKAGE

JASPER VAN WOUDENBERG
@JZVW
UPDATED SLIDES AT

WWW.RISCURE.COM/PUBLICATIONS/WHIP-THE-WHISPERER-BLACKHAT-2022
ABOUT JASPER

CTO Riscure North America
Author “Hardware Hacking Handbook”

• Software security since y2k
• Riscure: SCA/Fl/Hardware security since 2005
• Author since 2021
• Grey hair since 2015 becoming a father

https://www.hardwarehacking.io/
https://linkedin.com/jaspervw
https://twitter.com/@jzvw
WHAT DAT?
Flutter Flop Power Consumption

\[
\text{cell} \quad \begin{array}{c}
D \\
Q \\
CLK
\end{array}
\]

\[
Q \Rightarrow D \quad Q = Q
\]

Power during clock cycle

- \(Q \Rightarrow !Q\)
- \(Q = Q\)

Time
GUESS WHO?

Measurement A

- Q==0
- Q’==1

Measurement B

- Q==0
- Q’==0
GUESS THE AES KEY*

* This slide is incorrect (to avoid a 30 min power analysis lecture)
OK, ONE CELL LEAKS

YOU CAN’T MEASURE THAT
(CORRELATION) POWER ANALYSIS
POST-SILICON IS TOO LATE!

Post-silicon, we can fix the next chip.
Pre-silicon, we can fix the current chip.
SIMULATE IT!
SIMULATION INGREDIENTS

- RTL Netlist (post-syn)
- Netlist (post-pnr)
  - (‘Verilog / VHDL / DEF’ files)

All cell power models
- (‘liberty’ file)

All cell outputs over time
- (‘vcd’ file)

Power simulation tool
- (your fav EDA vendor)
# POWER TRACE GENERATION RUNTIMES

All experiments generated 256 power traces

<table>
<thead>
<tr>
<th>AES Core</th>
<th>RTL Gate Count</th>
<th>Synthesis Gate Count</th>
<th>P&amp;R Gate Count</th>
<th>RTL Trace Gen. Time</th>
<th>Synthesis Trace Gen. Time</th>
<th>P&amp;R Trace Gen. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>aes_256_serial_wddl</td>
<td>20991</td>
<td>19038</td>
<td>20991</td>
<td>1:20:21</td>
<td>3:54:20</td>
<td>4:03:43</td>
</tr>
<tr>
<td>aes_256_serial</td>
<td>6808</td>
<td>5849</td>
<td>7136</td>
<td>0:38:23</td>
<td>1:52:03</td>
<td>2:22:33</td>
</tr>
<tr>
<td>picoaes</td>
<td>10663</td>
<td>8765</td>
<td>9833</td>
<td>0:29:38</td>
<td>1:48:00</td>
<td>2:44:35</td>
</tr>
<tr>
<td>aes_fast_mix_sub</td>
<td>31321</td>
<td>25455</td>
<td>27627</td>
<td>1:18:15</td>
<td>2:32:13</td>
<td>3:02:21</td>
</tr>
<tr>
<td>femtoaes</td>
<td>4274</td>
<td>3442</td>
<td>4177</td>
<td>1:01:53</td>
<td>1:50:10</td>
<td>2:05:47</td>
</tr>
<tr>
<td>aes_128_serial</td>
<td>5303</td>
<td>4335</td>
<td>5101</td>
<td>1:02:58</td>
<td>1:39:51</td>
<td>1:55:03</td>
</tr>
</tbody>
</table>
(CORRELATION) POWER ANALYSIS ON SIMULATIONS

Modeled leakage

<table>
<thead>
<tr>
<th>K</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Time

d=2

d=1

d=3
PRE-SILICON SIDE CHANNEL LEAKAGE!
“WHEN” IT LEAKS DOESN’T HELP FIX IT

ENTER... SCATE

Red team can now go back to hacking
WE KNOW WHEN... BUT WHERE??
IDEA: CORRELATE NET SWITCHES TO LEAKAGE MODEL

Ranked list of ‘leaky’ gates:
- sbox_out1
- sbox_out2

Correlate
ARCHITECTURE CORRELATION ANALYSIS

For each net $N_i$

unrelated0
sbox_out1
sbox_out2
unrelated1
clk

"Toggle Trace" if

\[
\begin{align*}
\text{or} & = +1 \\
\text{or} & = -1
\end{align*}
\]
ARCHITECTURE CORRELATION ANALYSIS

unrelated0
sbox_out1
sbox_out2
unrelated1
clk

public

LTI

unrelated0
sbox_out1
sbox_out2
unrelated1
clk

Test vector $V_0$

Test vector $V_1$

“Toggle Trace”

or

= +1

= -1

For each net $N_i$

Event Toggle Trace $K_{ij}$

Architecture Correlation Cell $C_i$

$C_i = \text{correlate}(L(V_j), K_{ij})$
SUCH LEAKAGE
function [127 : 0] addroundkey(input [127 : 0] data, input [127 : 0] rkey);
begin
    addroundkey = data ^ rkey;
end
endfunction // addroundkey

public
COUNTERMEASURE TESTING
DOES MY COUNTERMEASURE WORK?

https://bandori.party/ajax/activity/5951/

https://upload.wikimedia.org/wikipedia/commons/f/f2/Quacky.jpg
COUNTERMEASURE: MASKING

CASE STUDY: MASKED AES DESIGN

How well does a masked AES design mitigate leakage?
CASE STUDY: MASKED AES DESIGN

No masking

With masking

\[
x \oplus m_1 \rightarrow x \oplus m_2
\]

\[
((x \oplus m_1) \oplus m_1) \oplus m_2 \rightarrow x \oplus m_2
\]
CASE STUDY: MASKED AES DESIGN

Masking – unconstrained synthesis

\[
\begin{align*}
x \oplus m_1 \rightarrow x \oplus m_2 \\
((x \oplus m_1) \oplus m_1) \oplus m_2 \rightarrow x \oplus m_2
\end{align*}
\]

Masking – constrained synthesis

\[
\begin{align*}
x \oplus m_1 \rightarrow x \oplus m_2 \\
((x \oplus m_1) \oplus m_2) \oplus m_1 \rightarrow x \oplus m_2
\end{align*}
\]
SOC TESTING
This is fine
SOC ANALYSIS

29,575 cells sky130
50MHz System Clock

li    a4,8
lw    a3,0(a4)    ; load plaintext[0]
sw    a3,4(a5)    ; STALL
lw    a3,4(a4)    ; load plaintext[1]
sw    a3,8(a5)    ; STALL
lw    a3,8(a4)    ; load plaintext[2]
sw    a3,12(a5)   ; STALL
lw    a4,12(a4)   ; load plaintext[3]
sw    a4,16(a5)   ; STALL
li    a4,24
lw    a3,0(a4)    ; load key[0]
sw    a3,20(a5)   ; STALL
lw    a3,4(a4)    ; load key[1]
sw    a3,24(a5)   ; STALL
lw    a3,8(a4)    ; load key[2]
sw    a3,28(a5)   ; STALL
lw    a4,12(a4)   ; load key[3]
sw    a4,32(a5)   ; STALL
li    a4,6
sw    a4,0(a5)    ; control
li    a4,4
sw    a4,0(a5)    ; start
li    a3,1
.L121:    ; poll until done
lw    a4,68(a5)
bne    a4,a3,.L121

modeled in ACA

RISC-V  UART  DMA

Memory Interface

External Memory

GPIO

AES

Timer
Software reads ciphertext

Software writes plaintext, key

Key sched Round 1

Software polling loop

plaintext

```assembly
li a4, 8
lw a3, 0(a4) ; load plaintext[0]
sw a3, 4(a5) ; STALL
lw a3, 4(a4) ; load plaintext[1]
sw a3, 8(a5) ; STALL
lw a3, 8(a4) ; load plaintext[2]
sw a3, 12(a5) ; STALL
lw a4, 12(a4) ; load plaintext[3]
sw a4, 16(a5) ; STALL
li a4, 24
lw a3, 0(a4) ; load key[0]
sw a3, 20(a5) ; STALL
lw a3, 4(a4) ; load key[1]
sw a3, 24(a5) ; STALL
lw a3, 8(a4) ; load key[2]
sw a3, 28(a5) ; STALL
lw a4, 12(a4) ; load key[3]
sw a4, 32(a5) ; STALL
li a4, 6
sw a4, 0(a5) ; control
li a4, 4
sw a4, 0(a5) ; start
li a3, 1
.L121: ; poll until done
lw a4, 68(a5)
bne a4,a3,.L121
```
ACA ON A SOC

L(V) = HW(pt xor key) at $p_{\text{threshold}} = 0.2$

Leakage outside AES engine

HW dev: “AES engine fine”
SW dev: “Microarch / SoC leakage not my domain”
Leakage is found in unexpected places. If you don’t look, you won’t find it.
FUTURE WORK
AUTOMATED COUNTERMEASURES

Future work
DESIGN SPACE EXPLORATION

For top N leaky gates, trade off security / user defined function / countermeasure.
## SECURITY SIGNOFF

<table>
<thead>
<tr>
<th>Metric</th>
<th>Target</th>
<th>Actual</th>
<th>Passed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leaky points</td>
<td>&lt;5</td>
<td>3</td>
<td>✓</td>
</tr>
<tr>
<td>Max T value</td>
<td>&lt;5</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>CPA rho</td>
<td>&lt;0.01</td>
<td>0.001</td>
<td>✓</td>
</tr>
</tbody>
</table>
CONCLUSIONS
CAVEATS

• Speed limited by CPU cores / design size
• Simulations are digital only
• Rely on accuracy of EDA simulators
CONCLUSIONS

• Post-silicon, we can fix the next chip. Pre-silicon, we can fix the current chip.
• Only when we know where leakage is, can we fix it.
• Leakage is found in unexpected places. If you don’t look, you won’t find it.
• Little knowledge of SCA is needed to run the analysis.
• Future work: automated countermeasures, signoff metrics, silicon validation.
Some slides stolen from:

- “Tools and Methods for Pre-silicon Analysis of Secure Hardware” – slides, Patrick Schaumont, WPI, 2022

SCATE acknowledgements:

- Peter Grossman et al, Intrinsix (a CEVA company)
- Patrick Shaumont et al, Worchester Polytechnic Institute
- Shreyas Sen et al, Purdue University
- Cees Breunesse, Nicole Fern, Raj Velegalati, Riscure

Other references

- Zerotoasic course – Matt Venn

https://www.hardwarehacking.io/
TRACK RECORD

Secure SoC Fl vulnerabilities:
• Analyzed 5-10 security critical IP blocks
• Found vulnerabilities in almost all
• Initial setup few weeks per IP block
• Re-testing fixes few days

Various AES cores for SCA:
• With and without countermeasures
We are:

- Enabling **non-security-expert** designers to **root cause** security issues
- Using **tools** to find identify issues when it’s **still quick** to fix them
- Reducing the need for **time consuming / expensive** post-silicon testing
- Reducing the **risk** of post-silicon **certification failure** and **insecure** products