Analyzing the Fault Sensitivity of Secure Embedded Software

Patrick Schaumont

Professor
Bradley Department of ECE
Virginia Tech

Acknowledgments
National Science Foundation and SRC

Bilgiday Yuce, Nahid Farhady Ghalaty, Conor Patrick, Chinmay Despande, Marjan Ghodrati, Leyla Nazhandali

(C) 2016 P. Schaumont (VT)
1. Faults are a security liability
2. Faults as a side-channel - DFIA
3. Biased Fault Attacks on Software
4. Breaking Software Fault Countermeasures
5. Outlook
The Fault Attack Principle

input → (Crypto) SW or HW → output

Fault

correct behavior

Analysis

output'
faulty behavior

Time Resolution

ns
µs
ms
DC

clk, V glitch
EM pulses
laser pulses
Trojan

Stronger fault analysis
Expensive injection

weaker fault analysis
low-cost injection

Spatial Resolution

global cm mm µm

LO-V, hi-T

(C) 2016 P. Schaumont (VT)
Why are Faults a Security Issue?

- May enable external control of execution
  - Denial of service
  - Control of critical decisions

```c
if (! access_allowed)
    abort();
```

(C) 2016 P. Schaumont (VT)
Why are Faults a Security Issue?

• May enable external control of execution
  • Denial of service
  • Control of critical decisions

```
if (! access_allowed)
    abort();
```

• May cause information leakage of secrets

```
if (key_bit)
    r1 = r1 + 1;
else
    r0 = r0 + 1;

out = f(r1);
```

(key_bit) leaks indirectly via out
Current DFA methods are quite good
IF
the fault model can be realized

[TM 2010] Single random byte fault at 8th round of AES-128: Key $2^{128} \rightarrow 2^{12}$
[LGS+ 2010] Two seq. byte fault at 9th, 10th round of AES-192: Key $2^{128} \rightarrow 1$
Implementations and Actual Faults

Cryptographic Algorithm → Fault Model

Fault Model

\[
\begin{align*}
\text{Random Byte} \\
\text{Random Bit} \\
\text{Chosen Bit}
\end{align*}
\]

DFA

C, C', C'', .. → K

Implementation

Fault Injection

Cryptographic Architecture → Fault
Biased Fault Attacks

Cryptographic Algorithm \rightarrow Fault Model

Random Byte
Random Bit
Chosen Bit

DFA
C, C', C'', .. \rightarrow K

Implementation

Fault Injection

Variable Fault Intensity

Cryptographic Architecture

Fault

Fault Bias
1-bit, 2-bit, ..

FSA [2010]
NUEVA [2012]
NUFVA [2013]
DFIA [2014]
DERA [2015]
...

(C) 2016 P. Schaumont (VT)
1. Faults are a security liability

2. Faults as a side-channel - DFIA

3. Biased Fault Attacks on Software

4. Breaking Software Fault Countermeasures

5. Outlook
Do Biased Faults Exist?
Do Biased Faults Exist?

(C) 2016 P. Schaumont (VT)
Yes, Biased Faults Exist

Clock Glitching

(C) 2016 P. Schaumont (VT)
Yes, Biased Faults Exist

Voltage Starving

T0  T1  T2  T3  Tclk
Q3
Q2
Q1
Q0
CLK

(C) 2016 P. Schaumont (VT)
Fault Intensity, Bias and Sensitivity

Fault Intensity

\[ \text{FI} = \frac{#\text{violated\_paths}}{#\text{total\_paths}} \]

Fault Bias

\[ \text{FB}(i) = \frac{\text{#violated\_paths}}{\text{#total\_paths}} \]

Fault Sensitivity

\[ \text{FS} = i \text{ for which } \text{FB}(i) < \varepsilon \]
Fault Bias as function of Fault Intensity

32-bit ripple carry adder
Spartan 3E

![Graph showing fault bias as a function of fault intensity.](image)

- Fault Bias = $\frac{29}{32} = 0.91$
- Fault Bias = $\frac{7}{32} = 0.22$

(C) 2016 P. Schaumont (VT)
Biased Faults as a Side Channel

**Classic Differential Power Analysis**

- Power Measurement
- Power Estimate

Diagram:
- **Input:** S, RK
- **Process:** SBOX(S ⊕ RK)
- **Output:** C

Plots:
- Power Measurement: Two curves with different patterns over time (t).
- Power Estimate: A single flat line over time (t).

(C) 2016 P. Schaumont (VT)
Biased Faults as a Side Channel

- Biased Fault Injection

S → SBOX → C

RK → C'

Correct S vs. Faulty S' (8-dimensional space)
Biased Faults as a Side Channel

Under Correct Key Hypothesis

\[ SBOX^{-1}(C' \oplus RK_{\text{hyp}}) \]

Under Wrong Key Hypothesis

\[ SBOX^{-1}(C' \oplus RK_{\text{hyp}}) \]
Differential Fault Intensity Analysis

1. Inject Faults at different Fault Intensities
\[ \text{HW}(S \oplus S') < \varepsilon \]

2. Collect Fault Ciphertext \( C' \)

3. For all Key hypothesis \( R_{K_{hyp}} \) compute
\[ S_{i,RK} = \text{SBOX}^{-1}(C' \oplus R_{K_{hyp}}) \]

4. Select RK for which
\[ RK = \text{ArgMin}(\sum_i \sum_j \text{HD}(S_{i,RK}, S_{j,RK})) \]
DFIA versus DFA

**DFA**
- makes a precise assumption on the injected fault
- needs a system of equations to resolve key guess

**DFIA**
- makes an approximate model of the injected fault
- uses max likelihood testing to resolve key guess

DFIA relaxes the fault model requirements and is more suitable when fault injection is hard to control

*Relevant publications [DATE14] [FDTC14] [COSADE15] [IEEE ESL16]*
1. Faults are a security liability
2. Faults as a side-channel - DFIA
3. Biased Fault Attacks on Software
4. Breaking Software Fault Countermeasures
5. Outlook
Fault Attacks on Software

The black-box model

input

(Crypto) SW

output

output'

correct behavior

faulty behavior

Analysis
Fault Attacks on Software

The grey-box model

SW (fault analysis target) ≠ HW (fault injection target)
In-order RISC Pipeline Example

```
ld    [%o3 + 0xb], %o4
ldub  [%o0 + 0xb], %o5
ldub  [%o4 + 0xb], %g1
xor   %g1, %o5, %g1
stb   %g1, [%o3 + 0xb]
```
### In-order RISC Pipeline Example

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld</td>
<td>[%o3 + 0xb], %o4</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o0 + 0xb], %o5</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o4 + 0xb], %g1</td>
</tr>
<tr>
<td>xor</td>
<td>%g1, %o5, %g1</td>
</tr>
<tr>
<td>stb</td>
<td>%g1, [%o3 + 0xb]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stage</th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
In-order RISC Pipeline Example

```
ld [%o3 + 0xb], %o4
ldub [%o0 + 0xb], %o5
ldub [%o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [%o3 + 0xb]
```

<table>
<thead>
<tr>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
### In-order RISC Pipeline Example

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld</td>
<td>[%o3 + 0xb], %o4</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o0 + 0xb], %o5</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o4 + 0xb], %g1</td>
</tr>
<tr>
<td>xor</td>
<td>%g1, %o5, %g1</td>
</tr>
<tr>
<td>stb</td>
<td>%g1, [%o3 + 0xb]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
In-order RISC Pipeline Example

ld [0o3 + 0xb], %o4
ldub [0o0 + 0xb], %o5
ldub [0o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [0o3 + 0xb]

<table>
<thead>
<tr>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### In-order RISC Pipeline Example

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld</td>
<td>[%o3 + 0xb], %o4</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o0 + 0xb], %o5</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o4 + 0xb], %g1</td>
</tr>
<tr>
<td>xor</td>
<td>%g1, %o5, %g1</td>
</tr>
<tr>
<td>stb</td>
<td>%g1, [%o3 + 0xb]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In-order RISC Pipeline Example

ld [%o3 + 0xb], %o4
ldub [%o0 + 0xb], %o5
ldub [%o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [%o3 + 0xb]

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
In-order RISC Pipeline Example

\[
\begin{align*}
\text{ld} & \quad [%o3 + 0xb], %o4 \\
\text{ldub} & \quad [%o0 + 0xb], %o5 \\
\text{ldub} & \quad [%o4 + 0xb], %g1 \\
\text{xor} & \quad %g1, %o5, %g1 \\
\text{stb} & \quad %g1, [%o3 + 0xb] \\
\end{align*}
\]
In-order RISC Pipeline Example

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld</td>
<td>[%o3 + 0xb], %o4</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o0 + 0xb], %o5</td>
</tr>
<tr>
<td>ldub</td>
<td>[%o4 + 0xb], %g1</td>
</tr>
<tr>
<td>xor</td>
<td>%g1, %o5, %g1</td>
</tr>
<tr>
<td>stb</td>
<td>%g1, [%o3 + 0xb]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stage</th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
In-order RISC Pipeline Example

```
ld [%o3 + 0xb], %o4
ldub [%o0 + 0xb], %o5
ldub [%o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [%o3 + 0xb]
```

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td></td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
In-order RISC Pipeline Example

ld [%o3 + 0xb], %o4
ldub [%o0 + 0xb], %o5
ldub [%o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [%o3 + 0xb]

<table>
<thead>
<tr>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LL3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LL3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td></td>
<td>LL3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(C) 2016 P. Schaumont (VT)
1. **FI affects multiple instructions**

2. **Pipeline hazards affect sensitivity**

3. **FI effect depends on pipeline stage**

4. **Fault sensitivity depends on Instruction Pipeline Stage**

---

### Fault Injection (FI) Observations

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>D</td>
<td>A</td>
<td>E</td>
<td>M</td>
<td>X</td>
</tr>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td>LD1</td>
</tr>
</tbody>
</table>

---

(C) 2016 P. Schaumont (VT)
DFIA on SBOX access

ld [%o3 + 0xb0], %o4
ldub [%o0 + 0xb], %o5
ldub [%o4 + 0xb], %g1
xor %g1, %o5, %g1
stb %g1, [%o3 + 0xb]
DFIA on SBOX access

\[
\begin{align*}
ld & \quad [\%03 + 0xb0], \%04 \\
ld \text{dub} & \quad [\%00 + 0xb], \%05 \\
ld \text{dub} & \quad [\%04 + 0xb], \%01 \\
xor & \quad \%01, \%05, \%01 \\
\text{stb} & \quad \%01, [\%03 + 0xb]
\end{align*}
\]

<table>
<thead>
<tr>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>XOR</td>
<td>LD3</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FI Parameters**

- **ST (F)**: 6.7
- **XOR (D)**: 3.7
- **LD3 (A)**: 5.4
- **LD2 (E)**: 5.2
- **LD1 (M)**: 7.6

(C) 2016 P. Schaumont (VT)
Cryptographic Software

Timing = f(cycle, instruction)

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>STB</td>
<td>6.7</td>
<td>3.2</td>
<td>9.1</td>
<td>7</td>
<td>5.4</td>
<td>4.9</td>
<td>0</td>
</tr>
<tr>
<td>LD</td>
<td>5.7</td>
<td>3.4</td>
<td>5.4</td>
<td>5.2</td>
<td>7.6</td>
<td>2.8</td>
<td>5.6</td>
</tr>
<tr>
<td>LDUB</td>
<td>7</td>
<td>3.6</td>
<td>6.5</td>
<td>5.1</td>
<td>7.2</td>
<td>2.6</td>
<td>4.5</td>
</tr>
<tr>
<td>XOR</td>
<td>5.5</td>
<td>3.7</td>
<td>6.5</td>
<td>5.1</td>
<td>0</td>
<td>3.3</td>
<td>4.9</td>
</tr>
</tbody>
</table>
Measurement & Verification Setup

- Spartan-6 XC6SLX75 Debug Support Unit (DSU)
- Instruction Trace Buffer (ITB)
- Pipeline Trace Register (PTR)
- USB/Serial i/f (FTDI)
- Clock Glitch Controller
- Clock Glitch Injector
- Glitch Injector
- Glitch Injects
- USB GRMON Debug Monitor
- Python Scripts
- Control PC
- Glitch-injector
- JTAG
- JTAG Trigger
- Glitch-free clock
- Pulse Generator (Agilent 81110A)
- Circular Instruction Queue
- Pipeline Snapshot @glitch
- Relevant publication [FDTC16]
Using the μArch Fault Sensitivity Model

DFIA attack with/without FS Model
  • with/without fault sensitivity model
  • on AES-SBOX and TBOX software

→ 10x reduction of fault injection space

<table>
<thead>
<tr>
<th>Design</th>
<th>Glitch Span Fault Intensity</th>
<th># Cycles</th>
<th># Fault Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black-Box (wo model)</td>
<td>3 – 15.8</td>
<td>13</td>
<td>1040</td>
</tr>
<tr>
<td>AES-SBOX</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES-TBOX</td>
<td>3 – 15.8</td>
<td>16</td>
<td>1280</td>
</tr>
<tr>
<td>with model</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES-SBOX</td>
<td>4.7 – 6.9</td>
<td>6</td>
<td>90</td>
</tr>
<tr>
<td>AES-TBOX</td>
<td>5.4 – 6.6</td>
<td>9</td>
<td>81</td>
</tr>
</tbody>
</table>

@160ps resolution

Relevant publication [FDTC15]
1. Faults are a security liability
2. Faults as a side-channel - DFIA
3. Biased Fault Attacks on Software
4. Breaking Software Fault Countermeasures
5. Outlook
Software Fault-Attack Countermeasures

Software Countermeasure Fault Injection Detection based

Information Redundancy
Algorithmic Redundancy

at instruction-level:
  instruction duplication
  instruction triplication
  parity-invariant

Infection based

Currently, mostly broken ..
Instruction Duplication Countermeasure

- Duplicated execution of instructions, compare
- Breaking countermeasure requires back-to-back fault injection - considered difficult
- Micro-architecture Fault Sensitivity Model can pin down the weak spot of this countermeasure
Analyzing Instruction Duplication

```
ld    [%fp - 12], %g2
ld    [%fp - 12], %g3
cmp   %g2, %g3
bne   .error
```

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>X</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmp</td>
<td>LD2</td>
<td>LD1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bne</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Data hazard**
  - LD1
  - LD2
  - LD1
  - BNE
  - CMP
  - LD2
  - LD1

- **Branch interlock hazard**
  - LD1
  - LD2
  - LD1
  - BNE
  - CMP
  - LD2
  - LD1
  - BNE
  - CMP
  - LD2
  - BNE
  - CMP

(C) 2016 P. Schaumont (VT)
**Scenario 1 (single-glitch):**

1. Instruction Fault in CMP  
   CMP → NOP
2. Computation Fault in LD1  
   (Biased fault)
**Attack Scenarios**

```
ld [%fp - 12], %g2
ld [%fp - 12], %g3
cmp %g2, %g3
bne .error
```

**Scenario 2 (single glitch):**
1. Instruction Fault in BNE
   BNE → NOP
2. Computation Fault in LD1
   (Biased fault)
Scenario 3 (multi-glitch):
1. Computation Fault in LD1 (Biased fault)
2. Instruction Fault in BNE → NOP
### Verification on Prototype

<table>
<thead>
<tr>
<th>Glitch FI (ns)</th>
<th>Impacted Instruction</th>
<th>Fault Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scenario 1</strong></td>
<td>25 - 33</td>
<td>LD1 (A) CMP (D)</td>
</tr>
<tr>
<td><strong>Scenario 2</strong></td>
<td>32 - 38</td>
<td>LD1 (M) BNE (F)</td>
</tr>
<tr>
<td><strong>Scenario 3</strong></td>
<td>34 - 36</td>
<td>LD1 (E) BNE (D)</td>
</tr>
</tbody>
</table>

→ All scenarios break duplication countermeasure

*Relevant publication [FDTC16]*
1. Faults are a security liability
2. Faults as a side-channel - DFIA
3. Biased Fault Attacks on Software
4. Breaking Software Fault Countermeasures
5. Outlook
Summary

• Fault Models not just cryptographer’s imagination
  • Fault effects have physical causes and can be understood by a computer engineer
  • Insight into the fault effect leads to better fault attack

• Existing processors: Can software countermeasures be improved?
  • Yes: improve redundancy using bitslicing

*Relevant publication [SAC16]*
• **New Processors:**
  Can we integrate countermeasures into the $\mu$Arch?
  - **FAME** – Fault-attack Aware Microprocessor Extension
  - Fault Detection with Hardware Sensors;
    Micro-architectural support for state recovery;
    Software Trap handler to implement fault response
  - Design Details
    SRC e-Workshop 2/26
    HASP 2016 paper
  - Planned Tape-out 9/16
  - Design Report
    SRC Review 9/27/16
<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Event</th>
<th>Pages/Publication Details</th>
</tr>
</thead>
</table>
Thank you for your attention!
I’ll be happy to answer your questions.

Patrick Schaumont
schaum@vt.edu