Error-Tolerance in Trace-Driven Cache Collision Attacks

Jean-François Gallais Ilya Kizhvatov

UNIVERSITÉ DU LUXEMBOURG

Darmstadt, 25 February 2011

Motivation

- Improvement of the previous attack and adaptation to real life scenarii
- Observation of the cache profile in EM measurements
- Adjustment of the theoretical model

Outline

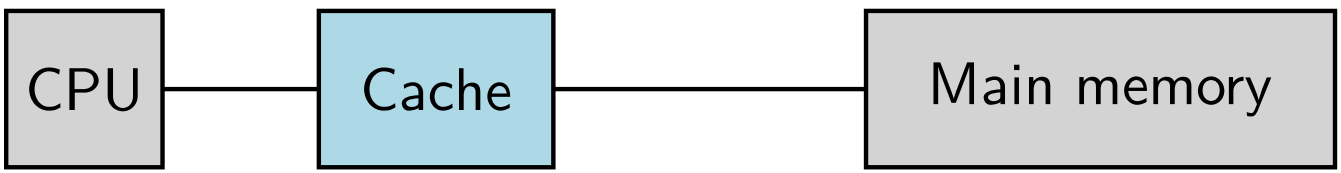
Outline

In this part

- the target is AES-128
- random inputs
- describe the leakage induced by the cache mechanism
- exploit it optimally

Cache in embedded devices

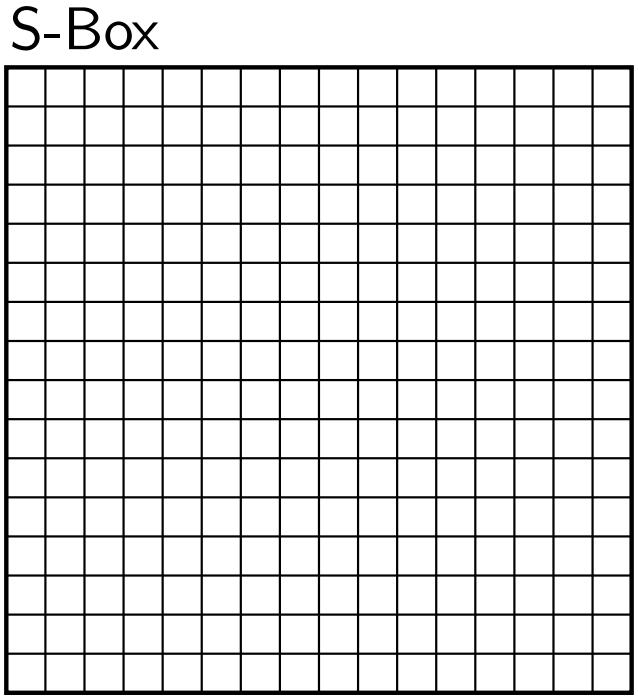
Many modern embedded μC's come with cache onboard: several ARM7, ARM9+



Cache speeds up access to frequently used memory areas, but creates a side-channel.

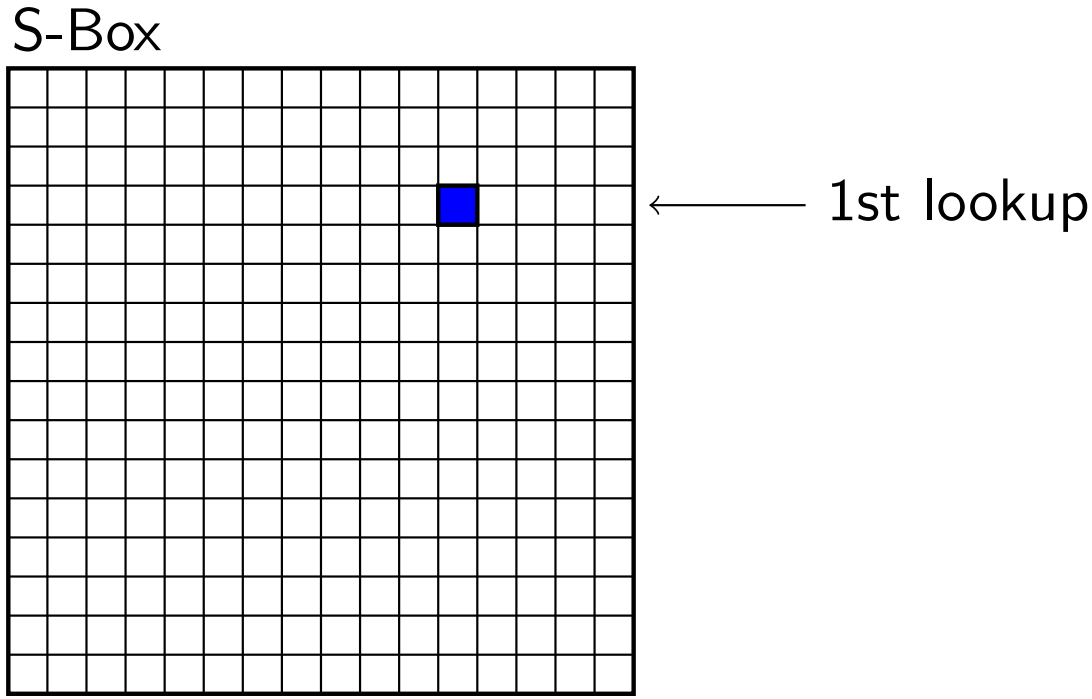
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



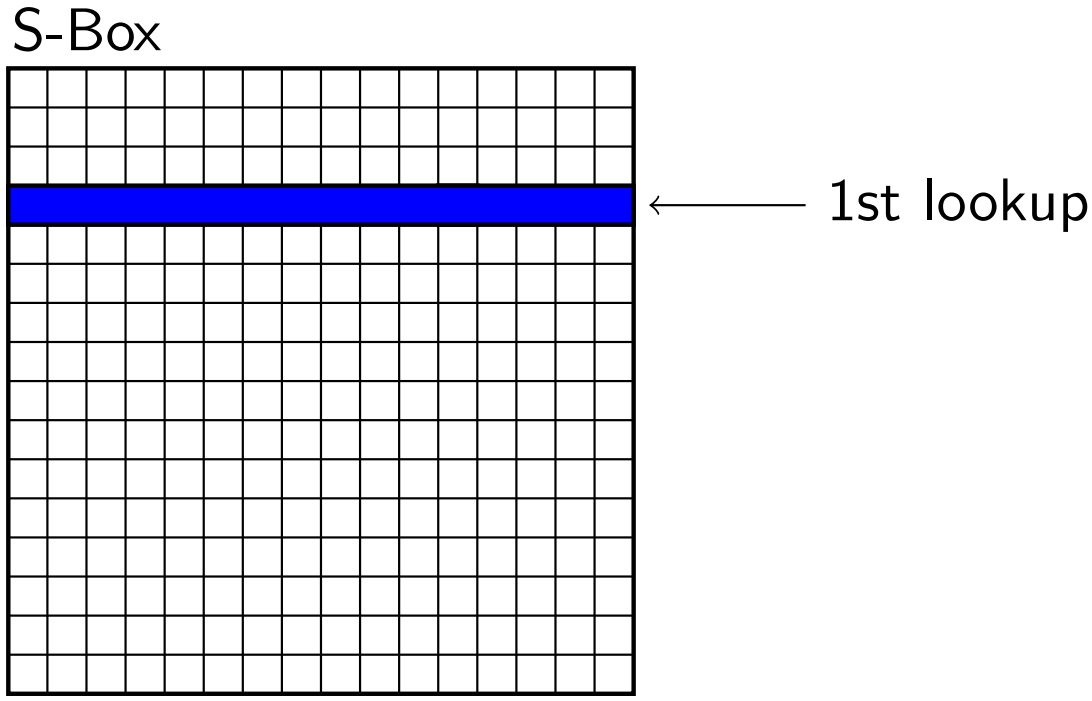
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



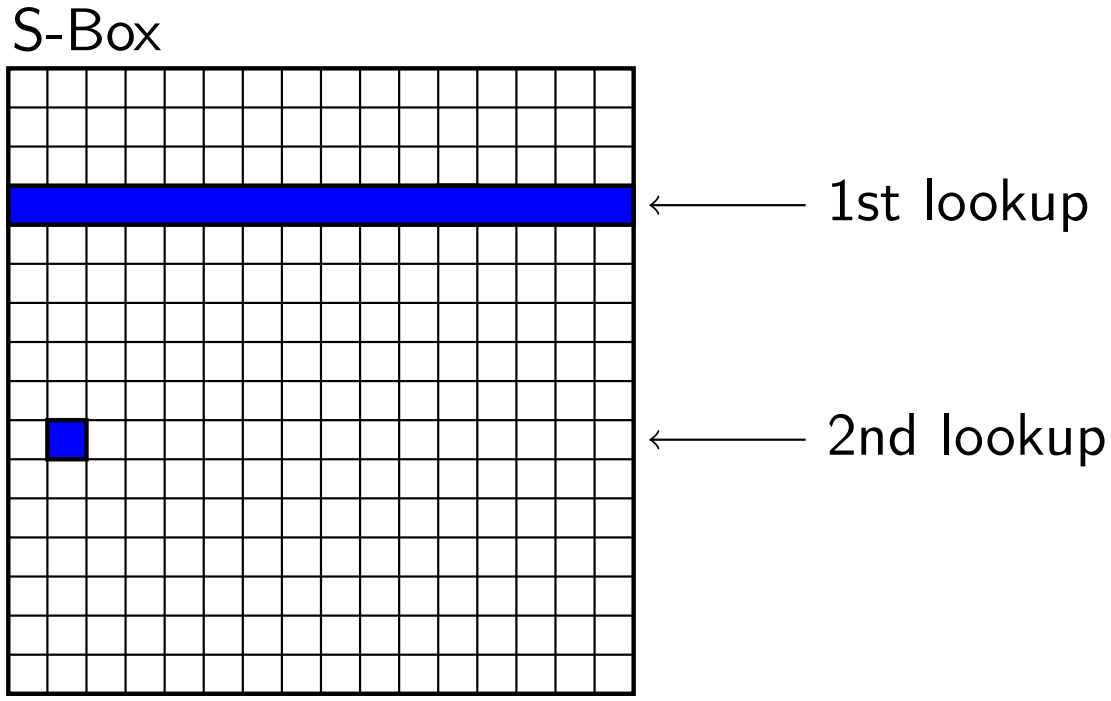
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



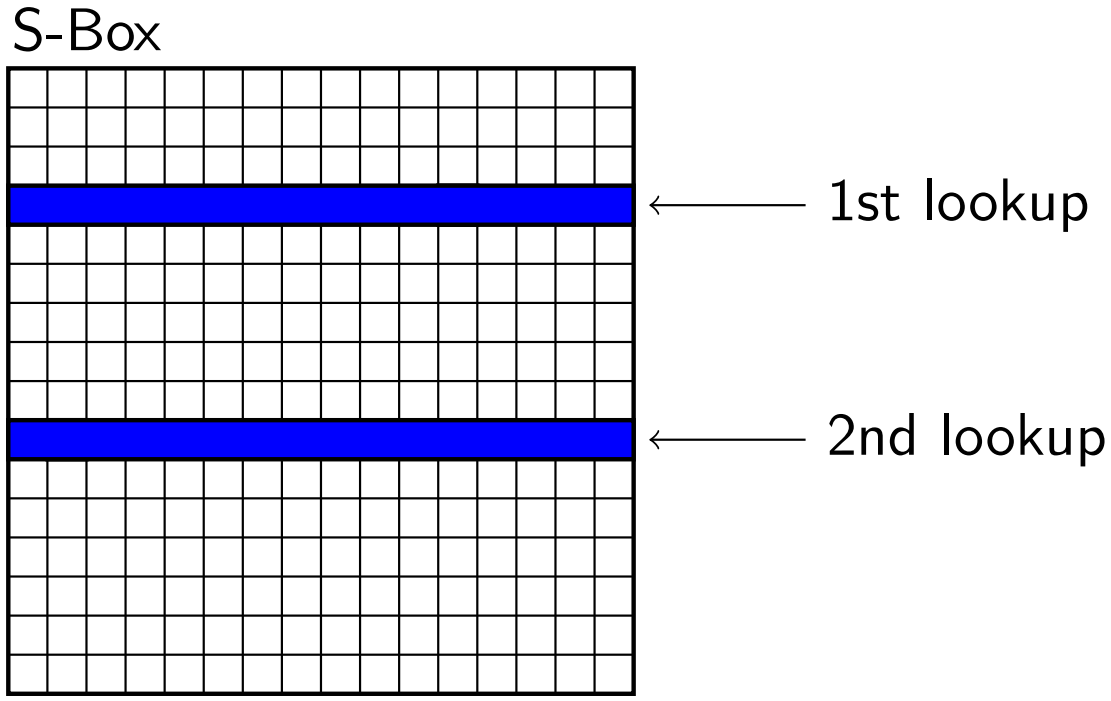
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



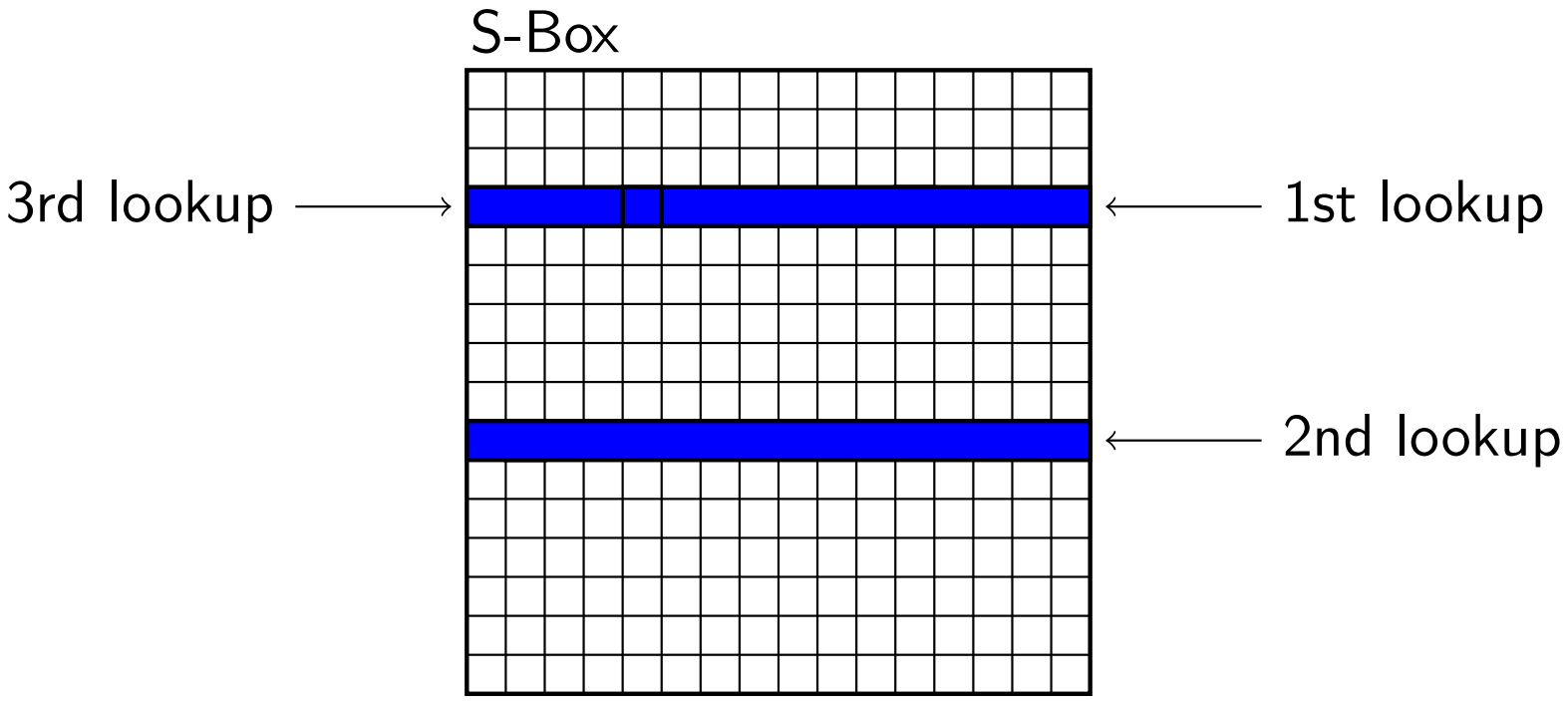
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



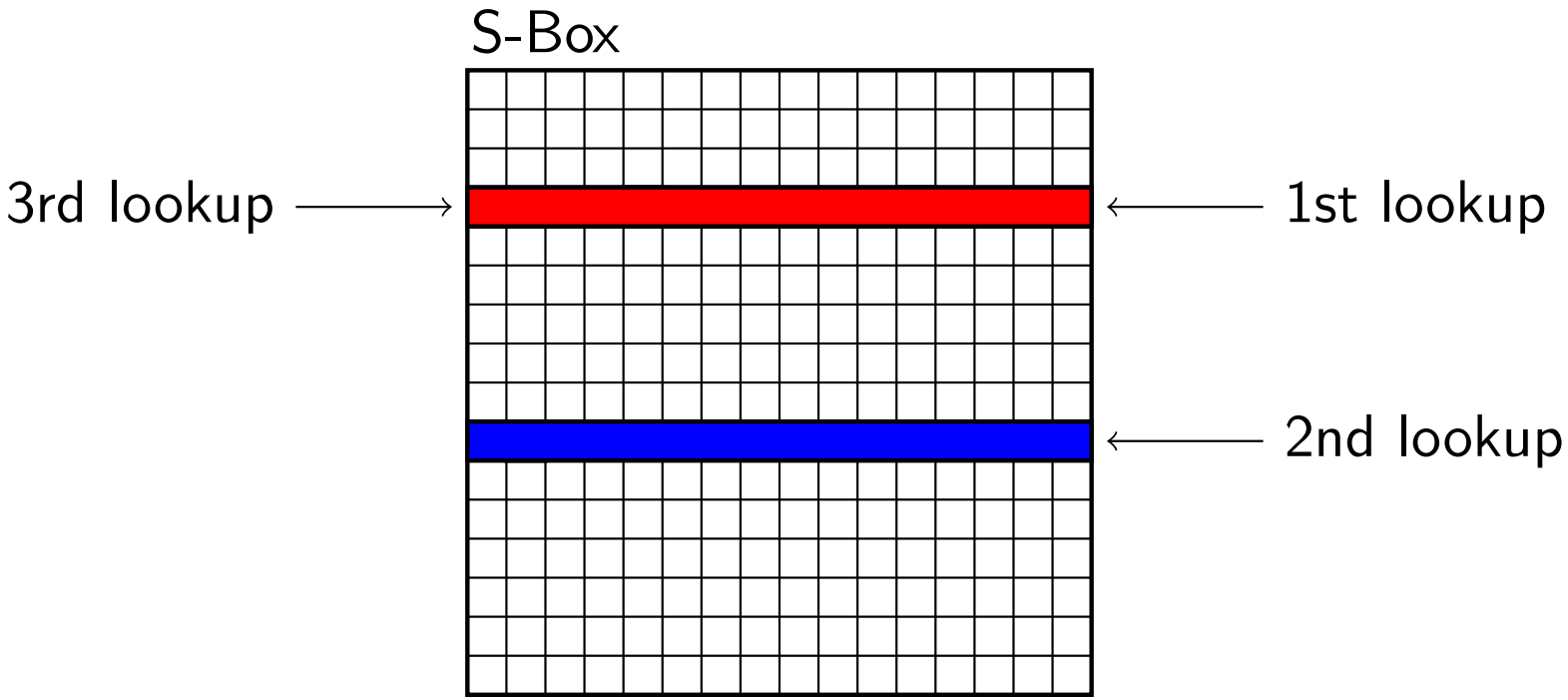
Cache collisions

For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.



Cache collisions

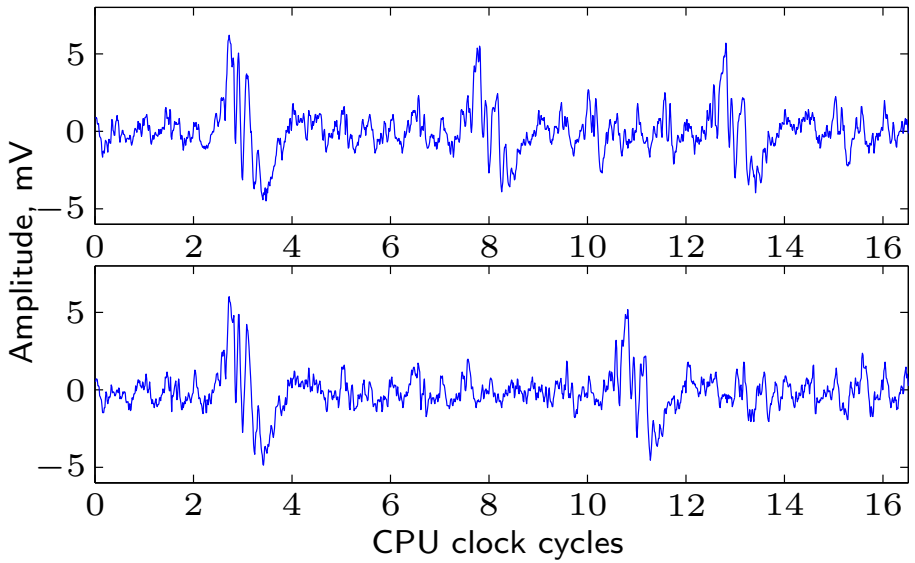
For the AES S-Box, cache events (hits and misses) reveal collisions in the higher order bits of S-Box inputs.

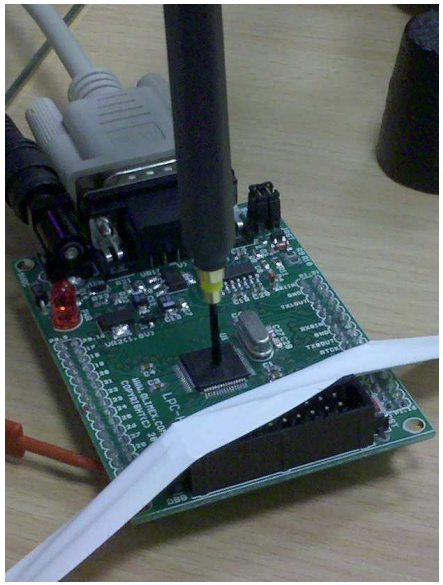


Cache collisions

- For cache line size of δ bytes, $8 - \log_2 \delta$ higher order bits are colliding.
- We will work with $\delta = 16$, so collisions will be in the upper nibbles; another common cache line size for embedded μC's is $\delta = 32$.
- Cache events can be visible in a side-channel trace.

Cache events in side channel traces





- NXP LPC2124 – an ARM7 with a primitive cache
- H-field passive microprobe, no shielding, no averaging
- execution of 3 AES S-Box lookups
- top: miss-miss-miss; bottom: miss-hit-miss

From cache profile To equations & inequations

First round cache trace

lookup index	0	1	2	3	4	5	6	...	15
cache event	M	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$	...	$\star$

At position i,

$$M \Rightarrow \forall j \in \Gamma,\ \widehat{k_i \oplus p_i} \neq \widehat{k_j \oplus p_j}$$

$$H \Rightarrow \exists! j \in \Gamma,\ \widehat{k_i \oplus p_i} = \widehat{k_j \oplus p_j}$$

From cache profile To equations & inequations

First round cache trace

lookup index	0	1	2	3	4	5	6	...	15
cache event	M	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$	...	$\star$

At position i,

$$M \Rightarrow \forall j \in \Gamma,\ \widehat{k_i \oplus p_i} \neq \widehat{k_j \oplus p_j}$$
$$H \Rightarrow \exists! j \in \Gamma,\ \widehat{k_i \oplus p_i} = \widehat{k_j \oplus p_j}$$

The relation $\widehat{k_i \oplus k_j} = \widehat{k_i \oplus k_0} \oplus \widehat{k_0 \oplus k_j}$ leads to:

$$M \Rightarrow \forall j \in \Gamma,\ \widehat{k_i \oplus k_0} \neq \widehat{p_i \oplus p_j} \oplus \widehat{k_j \oplus k_0}$$
$$H \Rightarrow \exists! j \in \Gamma,\ \widehat{k_i \oplus k_0} = \widehat{p_i \oplus p_j} \oplus \widehat{k_j \oplus k_0}$$

This suggests to process the analysis lookup by lookup.

Key recovery: first round

First round cache trace

lookup index	0	1	2	3	4	...	15
cache event	M	H	M	M	H	...	M

Resulting (in)equations

1. $\widehat{p_1 \oplus k_1} = \widehat{p_0 \oplus k_0}$
2. $\widehat{p_2 \oplus k_2} \neq \widehat{p_0 \oplus k_0}$
3. $\begin{cases} \widehat{p_3 \oplus k_3} \neq \widehat{p_0 \oplus k_0} \\ \widehat{p_3 \oplus k_3} \neq \widehat{p_2 \oplus k_2} \end{cases}$
4. $\begin{cases} \widehat{p_4 \oplus k_4} = \widehat{p_0 \oplus k_0} \\ \widehat{p_4 \oplus k_4} = \widehat{p_2 \oplus k_2} \\ \widehat{p_4 \oplus k_4} = \widehat{p_3 \oplus k_3} \end{cases}$

Key recovery: first round

By analyzing cache lookups sequentially with several traces, we obtain the full chain of 15 linear equations:

$$\widehat{k_0 \oplus k_1} = d_1$$

$$\widehat{k_0 \oplus k_2} = d_2$$

$$\dots$$

$$\widehat{k_0 \oplus k_{15}} = d_{15}$$

This chain reduces the key entropy by $15 \times 4 = 60$ bits.
The remaining 68 bits can be recovered from the second round.

Key recovery: second round

Input to the first S-Box lookup in the 2nd round

$$y_0 = 2 \bullet s(x_0) \oplus 3 \bullet s(x_5) \oplus s(x_{10}) \oplus s(x_{15}) \oplus s(k_7) \oplus k_0 \oplus 1$$

Resulting (in)equations

$$\begin{cases} \widehat{y}_0 & \neq & \widehat{x}_{j_1} \\ \dots & & \\ \widehat{y}_0 & \neq & \widehat{x}_{j_L} \end{cases}, \text{ if } \textbf{miss} \qquad \begin{cases} \widehat{y}_0 & = & \widehat{x}_{j_1} \\ \dots & & \\ \widehat{y}_0 & = & \widehat{x}_{j_L} \end{cases}, \text{ if } \textbf{hit}$$

where $j_1, \dots, j_L$ are indices of previous cache misses.

Sieve the 24-bit candidates $(\widehat{k}_0, \check{k}_0, \check{k}_5, \check{k}_7, \check{k}_{10}, \check{k}_{15})$ by checking against this system of (in)equations and remain with up to 4 candidates.

Key recovery: second round

By analyzing 4 lookups of the second round with several cache traces, we are left with up to 10 candidates for the full key.

$$y_0 = 2 \bullet s(x_0) \oplus 3 \bullet s(x_5) \oplus s(x_{10}) \oplus s(x_{15}) \oplus s(k_7) \oplus k_0 \oplus 1$$
$$y_1 = 2 \bullet s(x_1) \oplus 3 \bullet s(x_6) \oplus s(x_{11}) \oplus s(x_{12}) \oplus s(k_7) \oplus k_0 \oplus k_1 \oplus 1$$
$$y_2 = 2 \bullet s(x_2) \oplus 3 \bullet s(x_7) \oplus s(x_8) \oplus s(x_{13}) \oplus s(k_7) \oplus k_0 \oplus k_1 \oplus k_2 \oplus 1$$
$$y_3 = 2 \bullet s(x_3) \oplus 3 \bullet s(x_4) \oplus s(x_9) \oplus s(x_{14}) \oplus s(k_7) \oplus k_0 \oplus k_1 \oplus k_2 \oplus k_3 \oplus 1$$

Cache attack against boolean masking

A common countermeasure against DPA : boolean masking using a single mask.

Indices of S-box lookups share the same mask:
$\Rightarrow$ the equations and inequations remain valid.

Outline

Complications in real life

Two problems

- Cache not clean of S-Box lines prior to the execution.
- Errors in cache event detection: taking hits for misses and vice versa.

Both will ruin the basic attack due to incorrect equations.

Complications in real life

- The solution for the pre-loaded cache was already suggested [Bonneau06]: use only cache misses in the analysis.
- We provide the solution for the second problem: **error-tolerant** attack.

Error-tolerant attack

Modifications to the basic attack

- **hit**: treat previous uncertain events as misses
  i.e. include them into the system of equations
- **miss**: treat previous uncertain events as hits
  i.e. do not include them into the system of inequations
- **uncertain**: skip without analysis

This prevents us from excluding correct candidates with incorrect (in)equations.

Error-tolerant attack: an example

First round cache trace (no errors)

lookup index	0	1	2	3	4	...	15
cache event	M	H	M	M	H	...	M

Resulting (in)equations

1. $\widehat{p_1 \oplus k_1} = \widehat{p_0 \oplus k_0}$
2. $\widehat{p_2 \oplus k_2} \neq \widehat{p_0 \oplus k_0}$
3. $$\begin{cases} \widehat{p_3 \oplus k_3} \neq \widehat{p_0 \oplus k_0} \\ \widehat{p_3 \oplus k_3} \neq \widehat{p_2 \oplus k_2} \end{cases}$$
4. $$\begin{cases} \widehat{p_4 \oplus k_4} = \widehat{p_0 \oplus k_0} \\ \widehat{p_4 \oplus k_4} = \widehat{p_2 \oplus k_2} \\ \widehat{p_4 \oplus k_4} = \widehat{p_3 \oplus k_3} \end{cases}$$

Error-tolerant attack: an example

First round cache trace (with errors)

lookup index	0	1	2	3	4	...	15
cache event	M	U	U	M	H	...	M

Resulting (in)equations

1. skip
2. skip
3. $\widehat{p_3 \oplus k_3} \neq \widehat{p_0 \oplus k_0}$
4. $\widehat{p_4 \oplus k_4} = \widehat{p_0 \oplus k_0}$

   $\left\langle \begin{array}{l} \widehat{p_4 \oplus k_4} = \widehat{p_2 \oplus k_2} \\ \widehat{p_4 \oplus k_4} = \widehat{p_3 \oplus k_3} \\ \widehat{p_4 \oplus k_4} = \widehat{p_1 \oplus k_1} \end{array} \right.$

Error-tolerant attack

- Works both for 1st and 2nd round stages.
- Can be combined with the misses-only analysis in case of a preloaded cache.

Outline

In this part

- description of an accurate univariate model
- statistical dependency between the cache events

Aim: Estimating the measurement complexity (number of traces required)

- improvement of the univariate model
- inclusion of uncertain events

Univariate Model

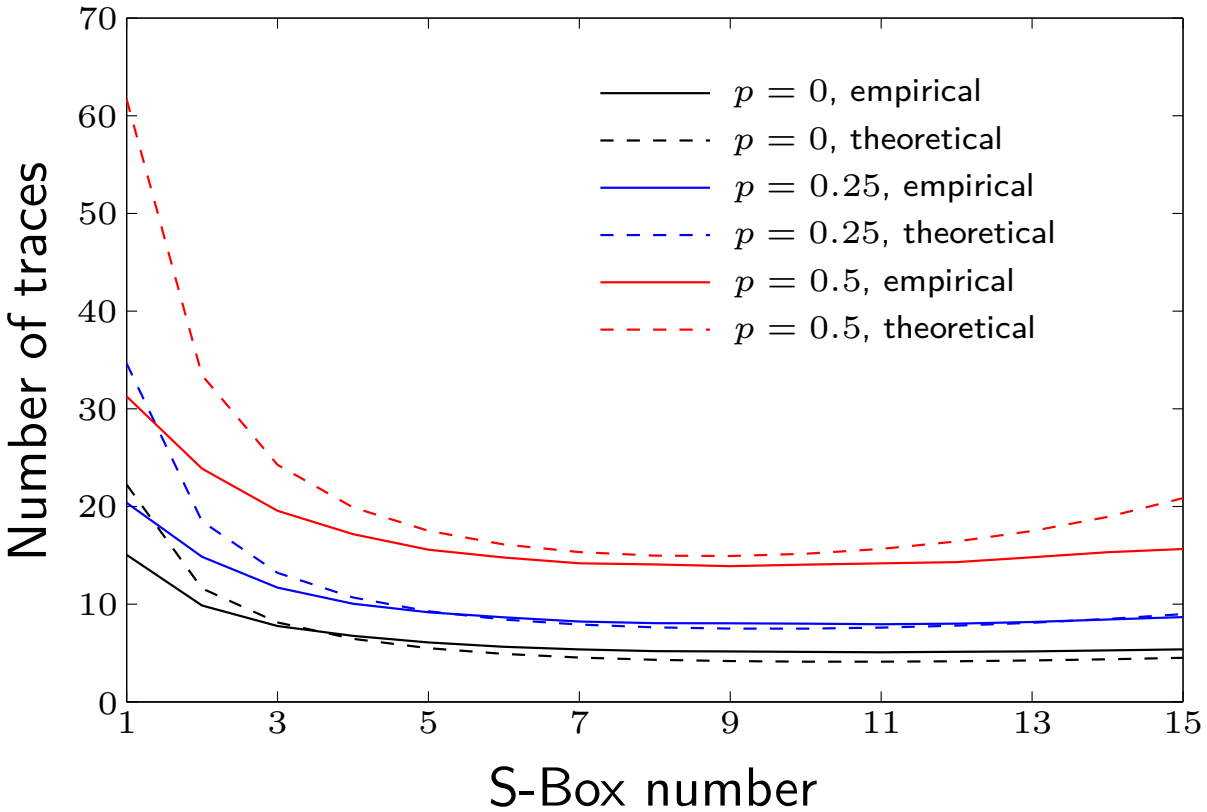
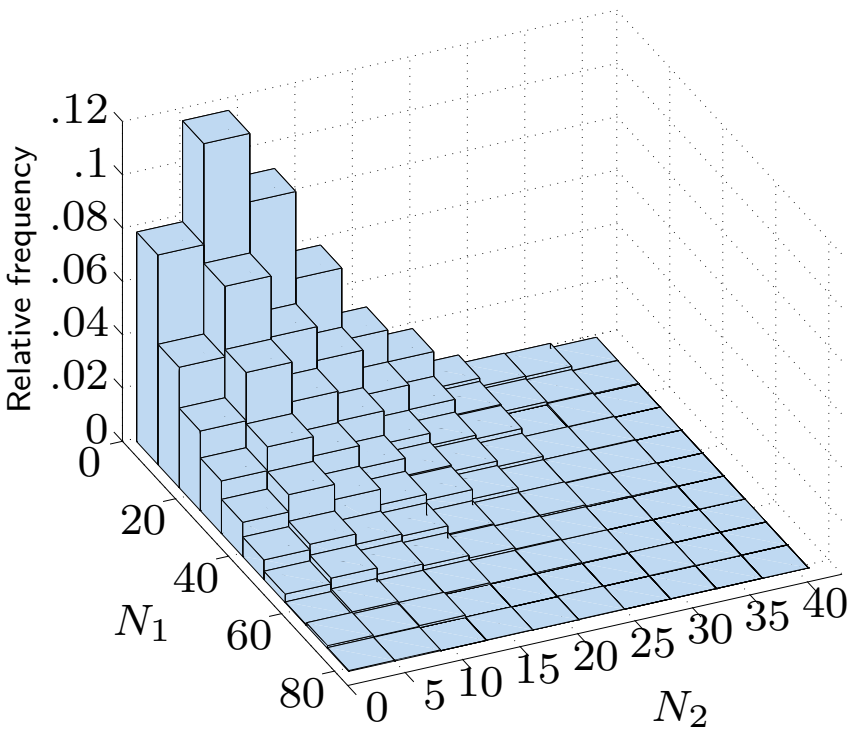


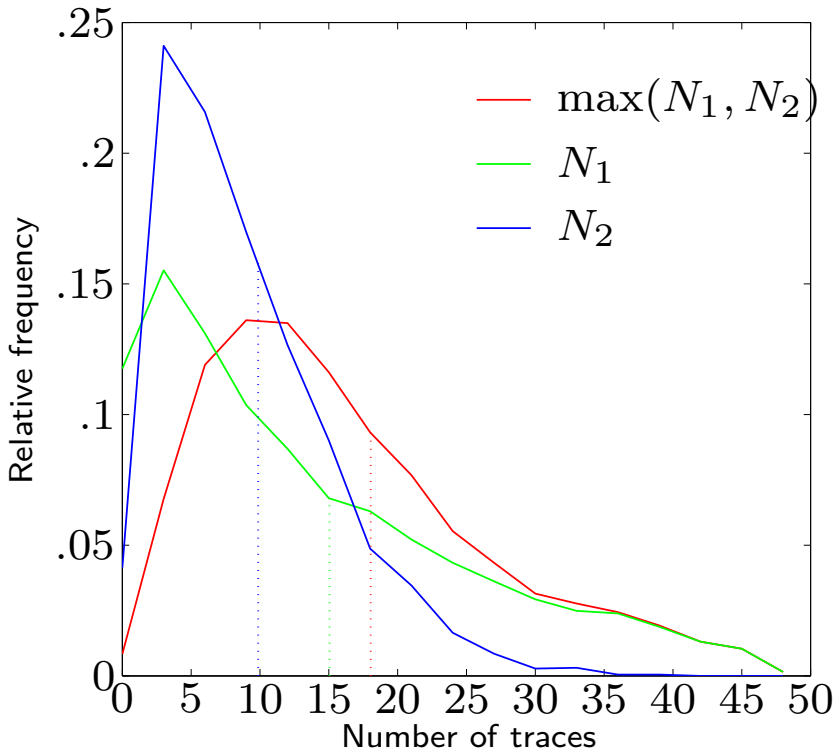
Figure: Expected number of traces for each lookup in the first round

Dependency between the events

The cache events are dependent, therefore:

$$\mathbf{E}(\max_i N_i) \neq \max_i(\mathbf{E}N_i)$$





Outline

Results

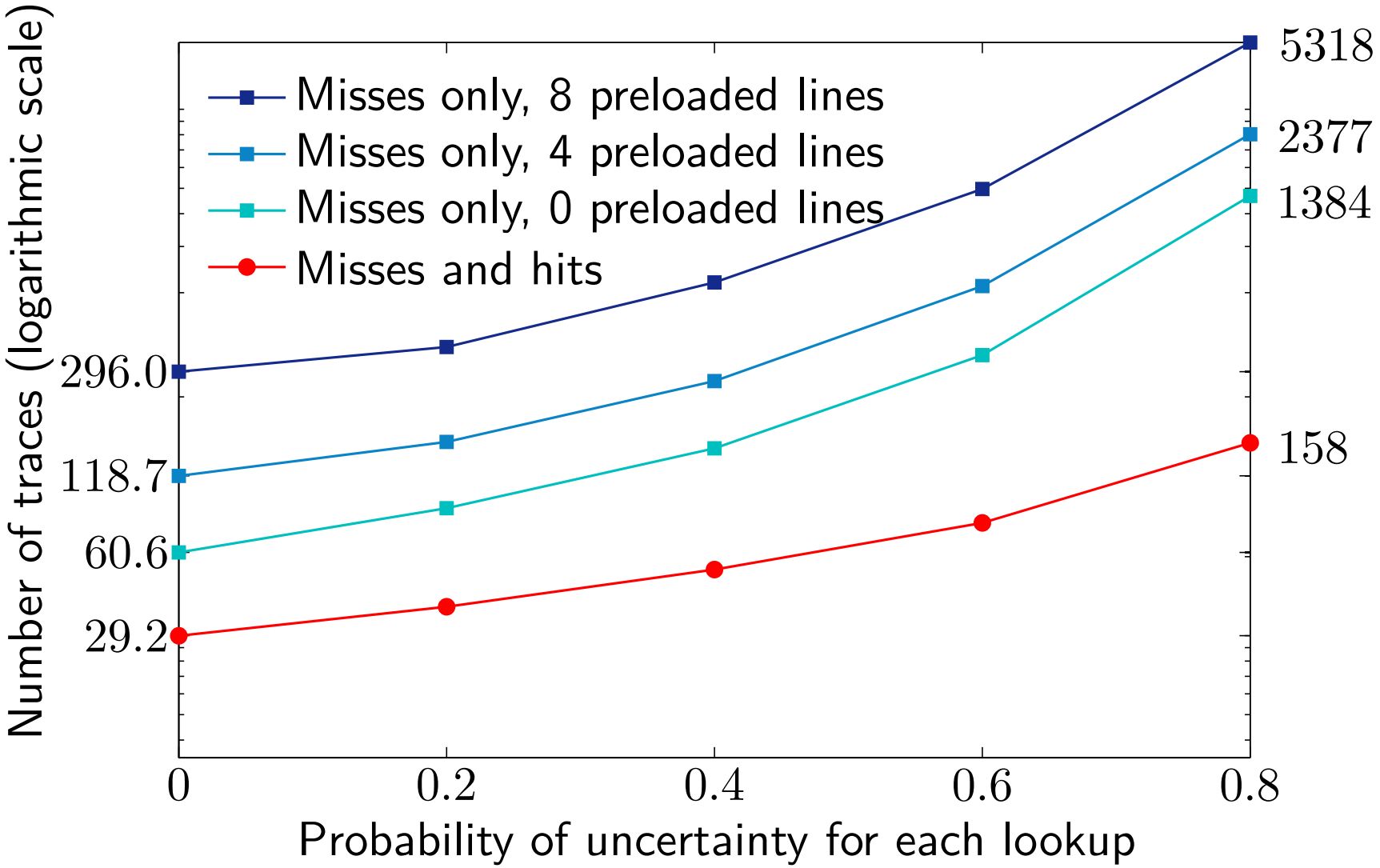


Figure: Average number of traces required for the full key recovery.

Summary

Conclusions

- cache profile observable in the EM leakage
- attack of decreased complexity
- fully adapted to error-tolerance and pre-loaded cache
- univariate model scrutinized
- only a multivariate model can describe the dependency between the events