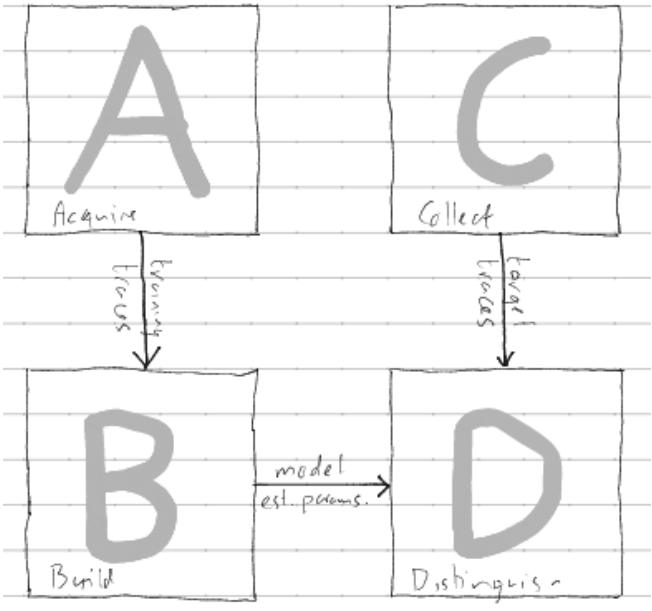
Pipeline for Our Example

Using SCALE Atmel dataset



A *Acquire training data*

- 1000 traces, random known plaintexts
- Fixed known key is less ideal
- Traces are already aligned

B *Build a profile*

1. We already identified potential PoIs
2. Model and profiling tbd

C *Collect target traces*

- 1000 traces, random known plaintexts

D Distinguish

1. Template Attack
2. Stochastic Attack

Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...

1. From Probability to Likelihood

For each key candidate k determine its a posteriori probability *given* the observed leakage L

Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...

1. From Probability to Likelihood

$$\Pr[k \mid L] = \frac{\Pr[L \mid k] \cdot \Pr[k]}{\Pr[L]}$$

$\Pr[L \mid k]$ is the likelihood
$\Pr[k]$ and $\Pr[L]$ can be ignored

Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...

2. From Likelihood to Sum of Log Likelihoods

Assume each trace leaks independently, then

$$\Pr[L \mid k] = \prod_i \Pr[L_i \mid k]$$

Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...

2. From Likelihood to Sum of Log Likelihoods

Assume each trace leaks independently, then *after taking logs*

$$\log_2 \Pr[L \mid k] = \sum_i \log_2 \Pr[L_i \mid k]$$

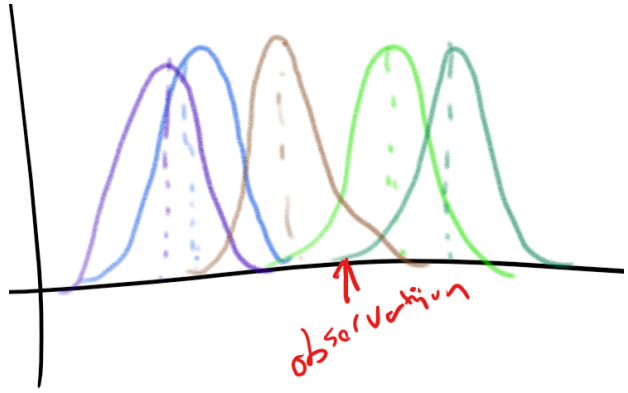
Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35…
1	123	4.65…
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79…

$\Rightarrow$

k_0	score
0	0.134…
1	0.116…
⋮	⋮
255	0.098…



From Log Likelihood to QDA

Assume

$$L(data) \sim \hat{M}(x_i \oplus k^*) + \mathcal{N}(0, \sigma)$$

then

$$\log_2 \Pr[L_i \mid k] = \log_2 \mathcal{N}(L_i - \hat{M}(x_i \oplus k); 0, \sigma)$$

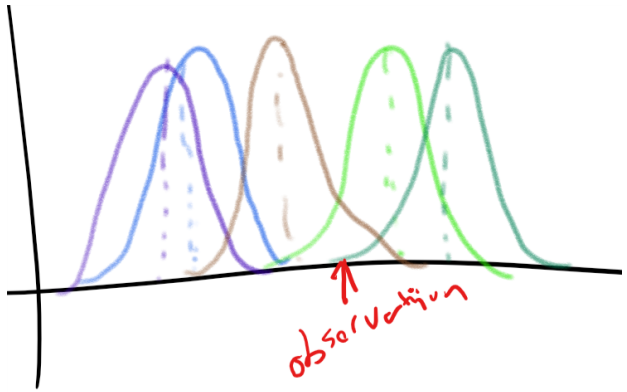
Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...



From Log Likelihood to QDA

Assume

$$L(data) \sim \hat{M}(x_i \oplus k^*) + \mathcal{N}(0, \sigma)$$

then

$$\log_2 \Pr[L_i \mid k] = -\log_2 e \left(L_i - \hat{M}(x_i \oplus k)\right)^2 / 2\sigma^2 - \frac{1}{2}(1 + \log_2 \pi) - \sigma$$

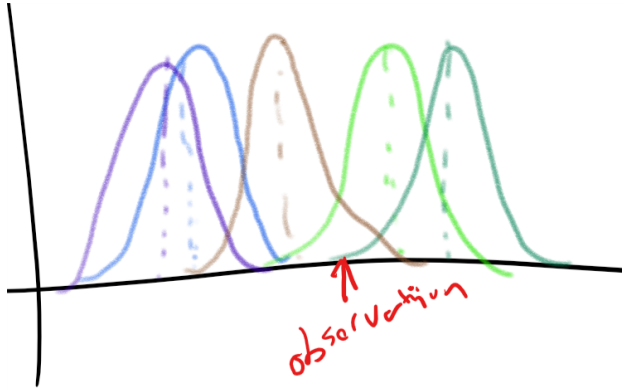
Template Attacks

Naive Bayes

i	x_i	power
0	12	1.35...
1	123	4.65...
⋮	⋮	⋮
⋮	⋮	⋮
999	59	2.79...

$\Rightarrow$

k_0	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...



QDA Summary

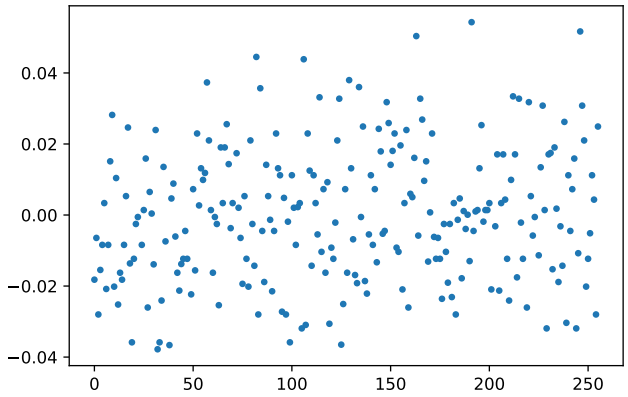
$$score(k|L) = \sum_i (L_i - \hat{M}(x_i \oplus k))^2$$

To profile: $\hat{M}(z)$ for all 256 possible z

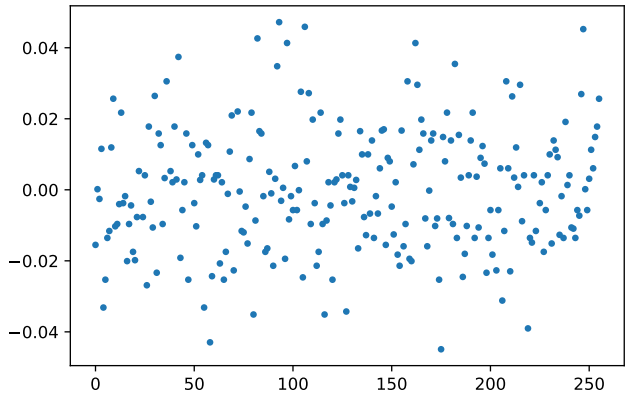
Warning: Scores can no longer be interpreted as posteriors

Template and Stochastic Attacks

SCALE Atmel Profiling



11005



11223

Template Attack

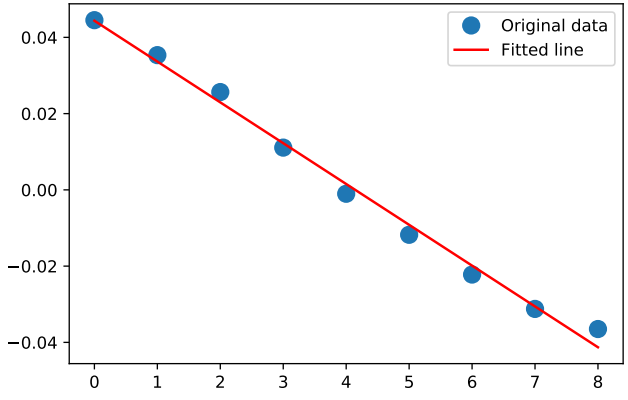
For all 256 possible S-box input values

- determine the sample mean
- (optional) determine the sample variance

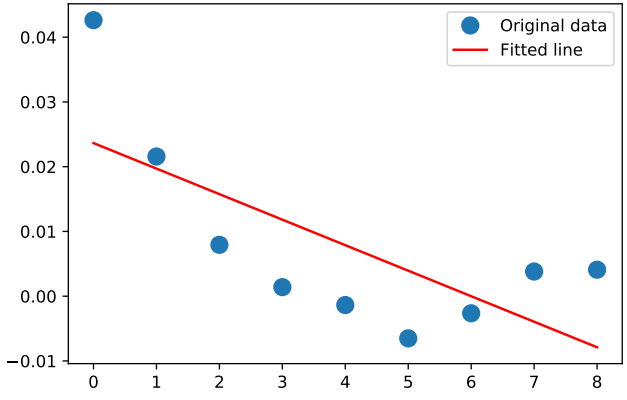
Problem: 1000 traces is not enough to estimate 256 parameters

Template and Stochastic Attacks

SCALE Atmel Profiling



11005



11223

Stochastic Attack

Assume the leakage model

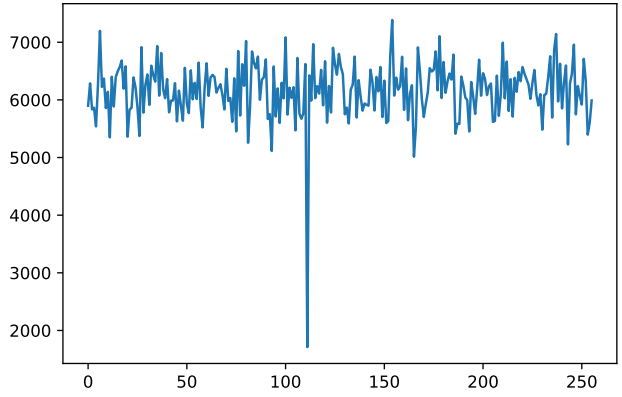
$M_{a,b}(k,x) = a \cdot HammingWeight(Sbox(x \oplus k)) + b$

- estimate a and b

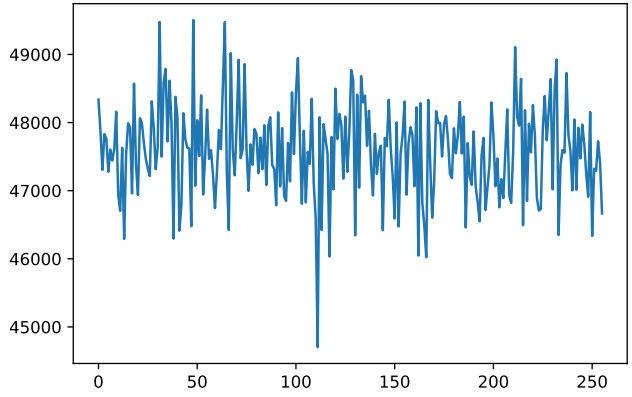
(*Warning:* The right estimation is naively unweighted)

Template Attacks

SCALE Atmel Scores



11005



11223

Final distinguishing scores

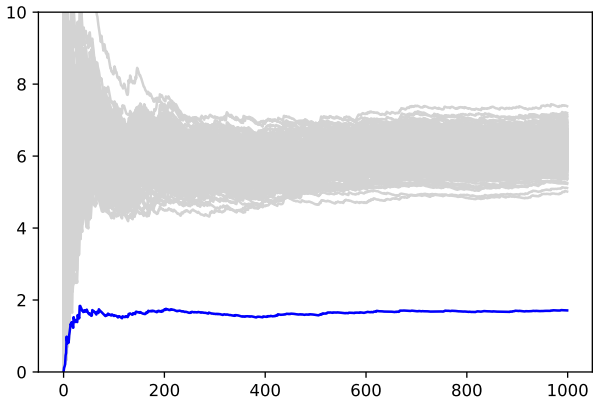
After incorporating 1000 target traces

`left` One candidate key *very* clearly sticks out

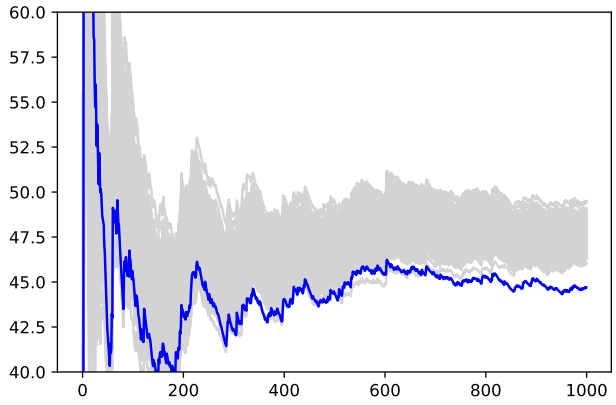
`right` One candidate key sticks out, but not as much

Template Attacks

SCALE Atmel Scores



11005



11223

Evolution of distinguishing scores

Look at scores as a function of number of traces incorporated

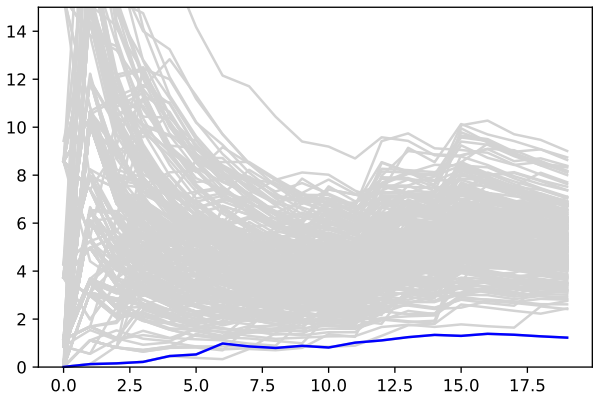
`left` the true key quickly separates from the rest

`right` it takes much longer for the true key to stand out

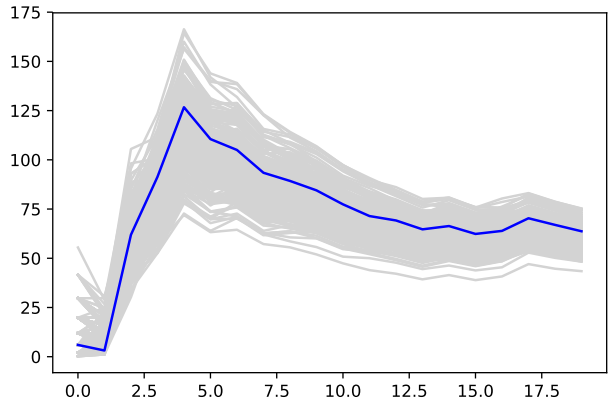
In blue the actual keybyte

Template Attacks

SCALE Atmel Scores



11005



11223

Evolution of distinguishing scores

Look at scores as a function of number of traces incorporated

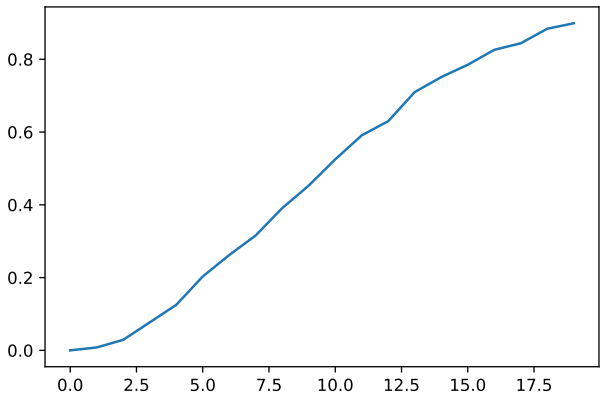
`left` the true key quickly separates from the rest

`right` it takes much longer for the true key to stand out

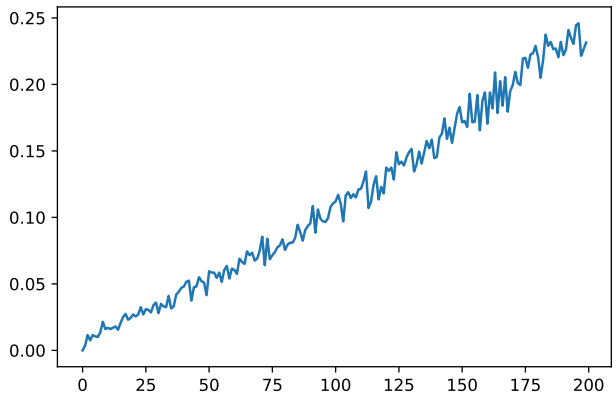
In blue the actual keybyte

Template Attacks

SCALE Atmel Success Rate



11005



11223

Success Rate: Probability that best guess wins

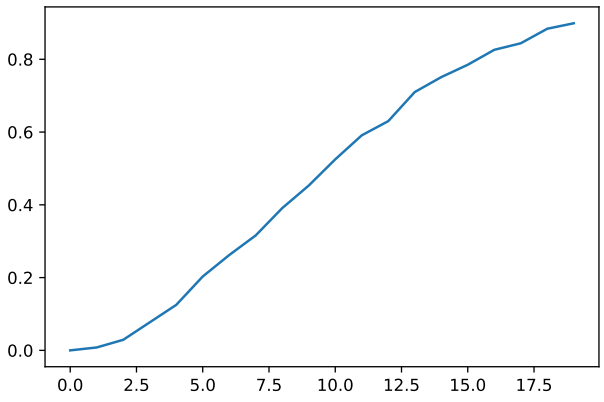
For each *i* (x-axis), ran 2000 experiments:

1. Selected *i* out of 1000 traces
2. Check if best guess is actual keybyte

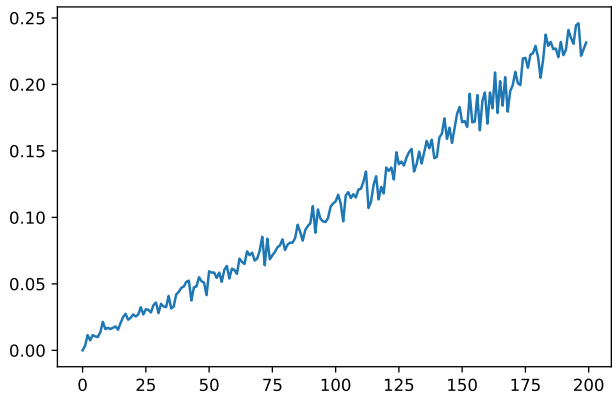
Warning: resampling methodology used due to available data

Template Attacks

SCALE Atmel Success Rate



11005



11223

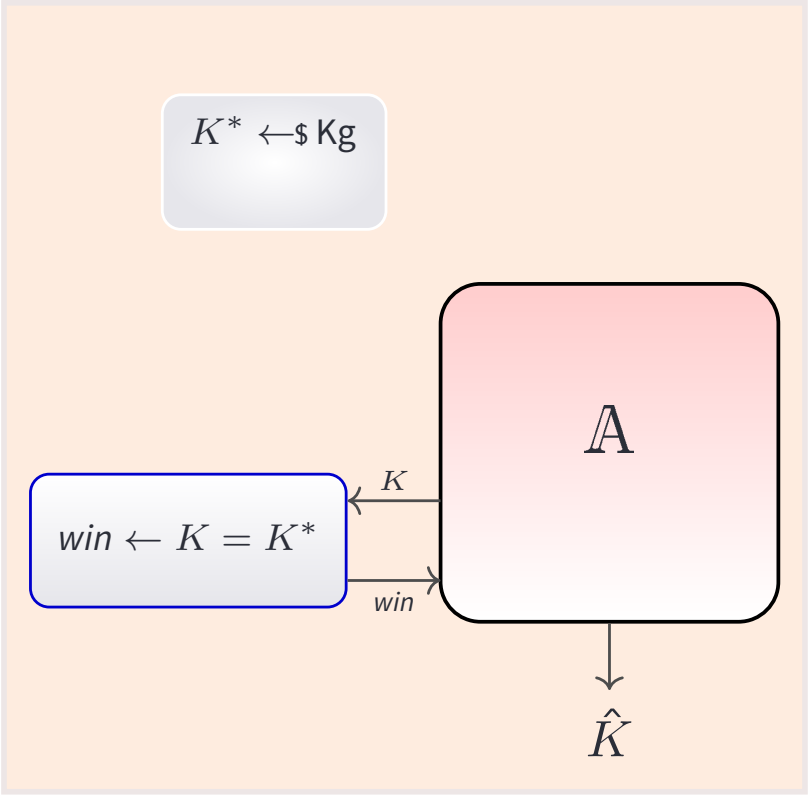
Success rate conclusion

1. Left performs better than right
2. Success rate 2^{-2} for a *single* keybyte, only gives 2^{-32} for the full 16-byte key.

Note: jaggedness likely due to low number of experiments

Different Adversarial Scenarios

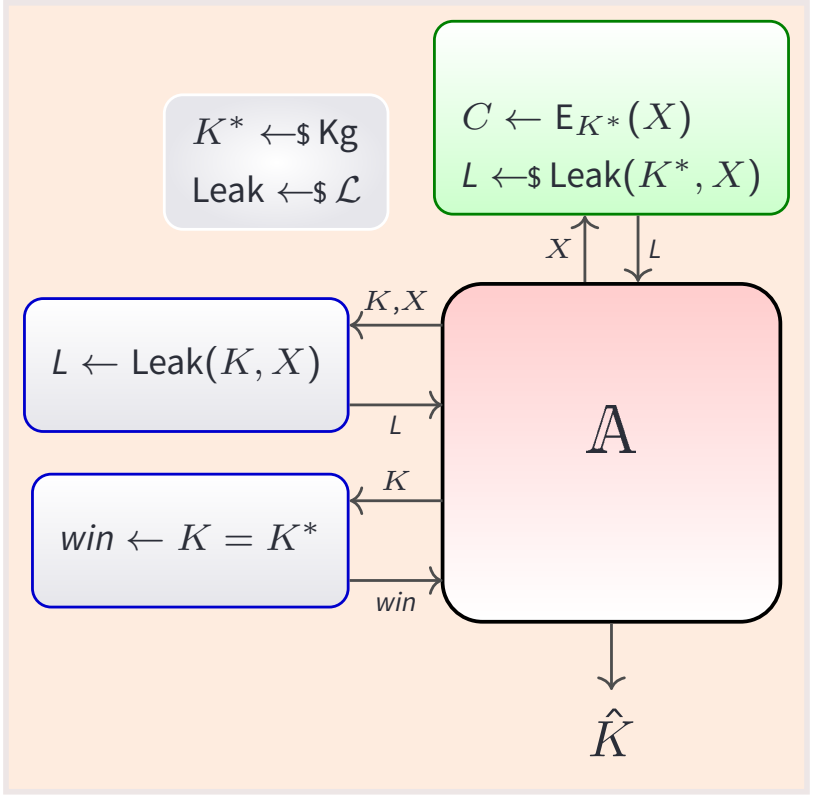
Not-Quite-Kerckhoffs Principle



The adversary can exhaustively search the key

Different Adversarial Scenarios

Not-Quite-Kerckhoffs Principle



The adversary can enumerate the key

Enumeration

Enhancing Divide-and-Conquer Attacks

k_0	score
0	0.123…
1	0.127…
⋮	⋮
255	0.238…

k_1	score
0	0.134…
1	0.116…
⋮	⋮
255	0.098…

…

k_{15}	score
0	0.184…
1	0.167…
⋮	⋮
255	0.152…

Best guess Simply output the most likely 128-bit key overall

Key enumeration Test keys from most likely to least likely until success

Enumeration

Enhancing Divide-and-Conquer Attacks

k_0	score
0	0.123…
1	0.127…
⋮	⋮
255	0.238…

k_1	score
0	0.134…
1	0.116…
⋮	⋮
255	0.098…

…

k_{15}	score
0	0.184…
1	0.167…
⋮	⋮
255	0.152…

Best guess obviously $k_0 = 0, k_1 = 255, \ldots, k_{15} = 255$

But what about the next best guess?

Question posed by Veyrat-Charvillon *et al.* (SAC'12)

Enumeration

Enhancing Divide-and-Conquer Attacks

k_0	score
0	0.123...
1	0.127...
⋮	⋮
255	0.238...

k_1	score
0	0.134...
1	0.116...
⋮	⋮
255	0.098...

...

k_{15}	score
0	0.184...
1	0.167...
⋮	⋮
255	0.152...

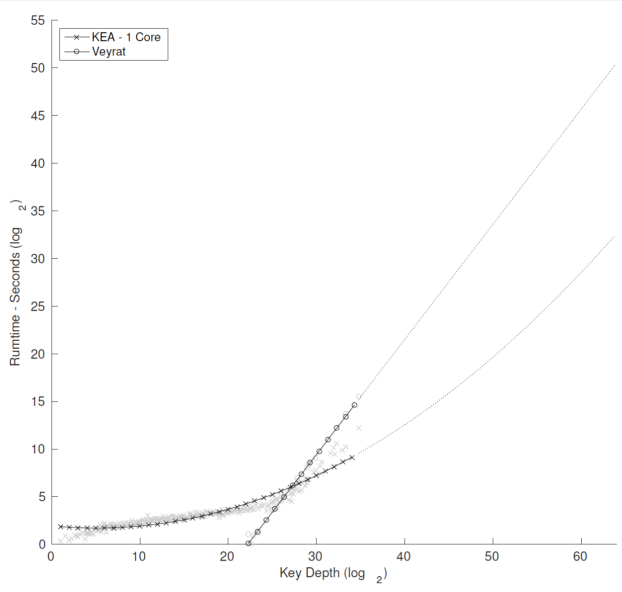
DPA with Enumeration

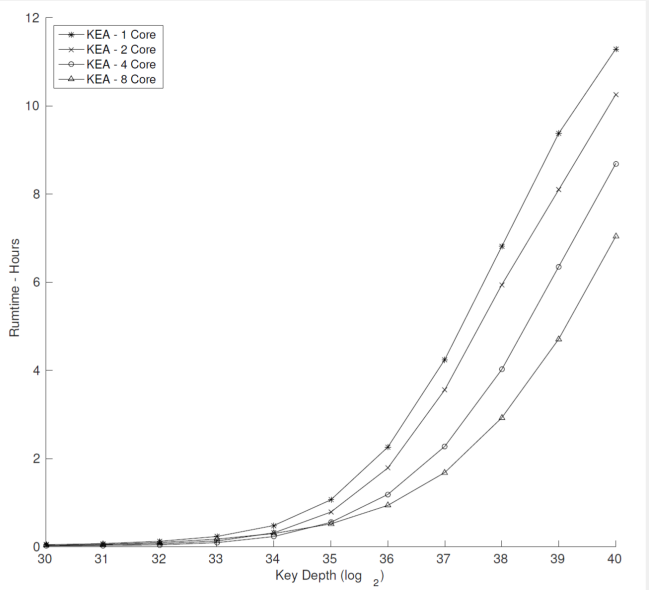
A number of cost metrics

1. The number of traces (profile vs.target)
2. The running time of the distinguisher
3. The number of keys to test
4. *The overhead (in time) to enumerate*

Enumeration

Enhancing Divide-and-Conquer Attacks





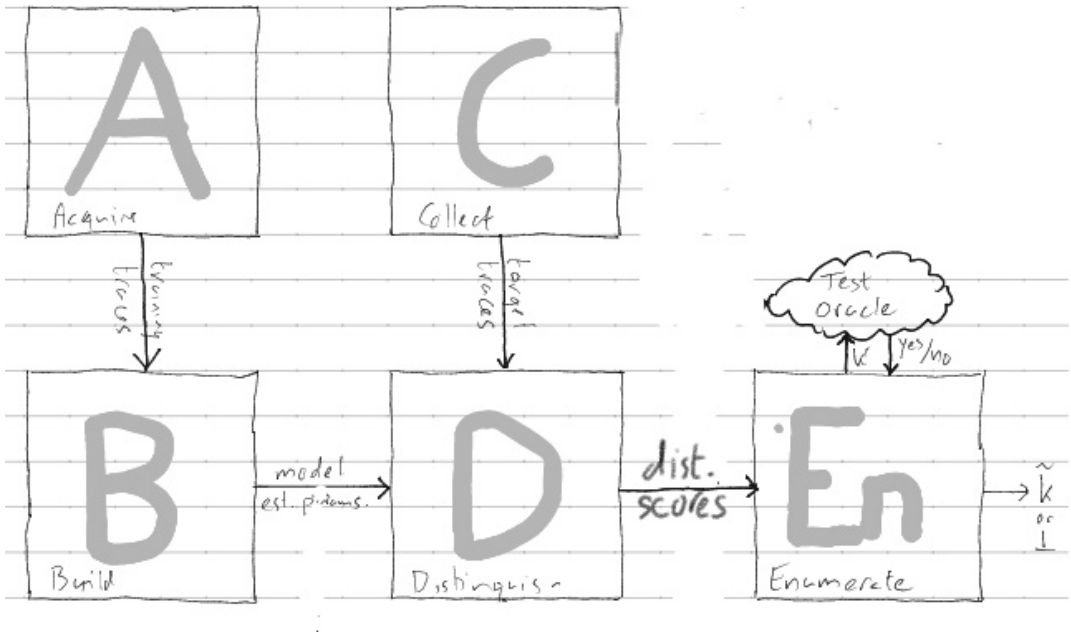
Some approaches

Naive Create ordered list of all 2^{128} keys

2012 Tree-like recursion algorithm
[Veyrat-Charvillon, Gérard, Renauld, Standaert / SAC]

2015 Dynamic programming enabling parallellization
[Martin, O'Connell, Oswald, Stam / Asiacrypt]

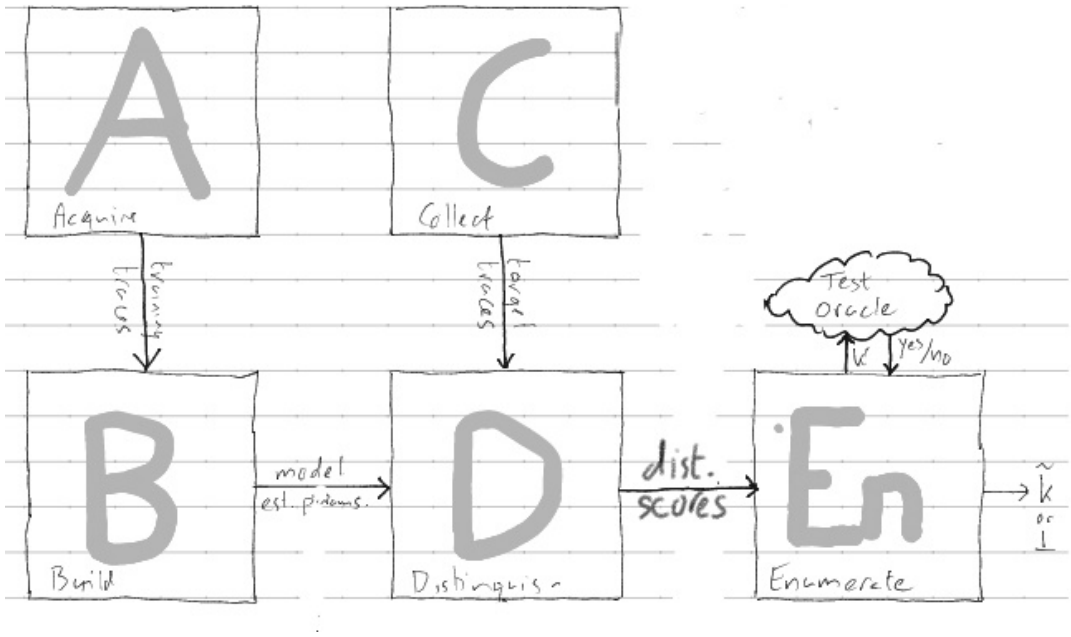
A Typical Side-Channel Attack Pipeline



Adding Enumeration

After the Distinguish phase, the scores are fed to an Enumeration phase

A Typical Side-Channel Attack Pipeline



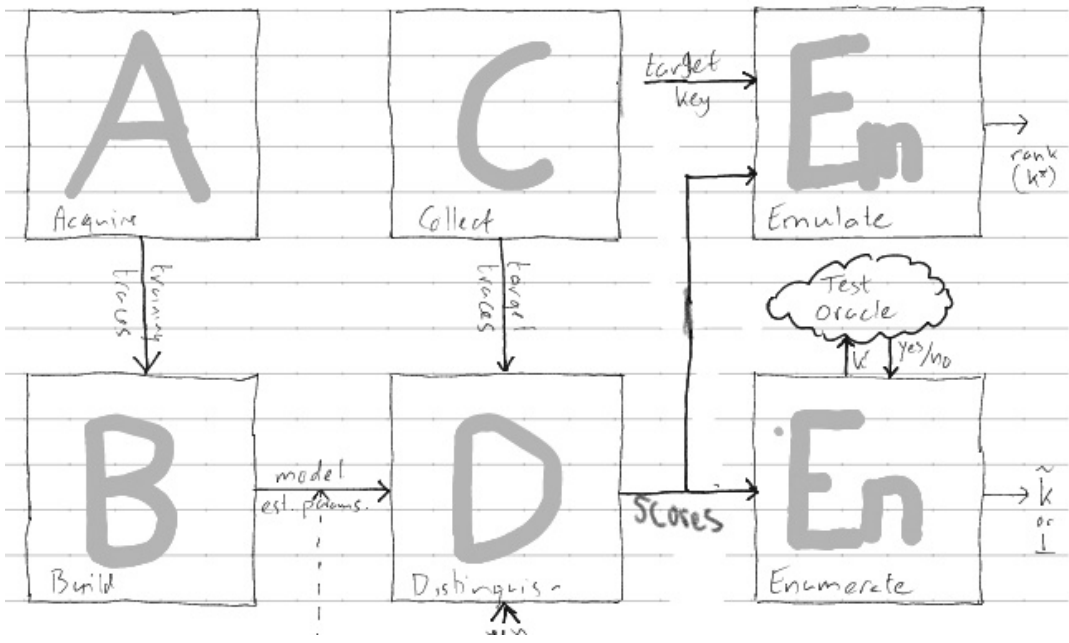
Adding Enumeration

After the Distinguish phase, the scores are fed to an Enumeration phase

But how long will it take, roughly?

Question posed by Veyrat-Charvillon *et al.* (Eurocrypt'13)

A Typical Side-Channel Attack Pipeline



Emulating Enumeration

After the Distinguishing phase,

- use *knowledge of the target key* to determine its rank.

Rather than running enumeration, Emulate it to predict its runtime

Key Ranking

Emulating the cost of key enumeration

Relevance: Evaluation

Many SCA are run by evalution labs:

- The care not about actually recovering the key
- *Only* how difficult it is to do so

The target key will be known already!

Ranking algorithms

A number of relevant metrics

1. The time to compute
2. Potential for parallellization
3. Quality of the returned rank when *approximating*

Key Ranking

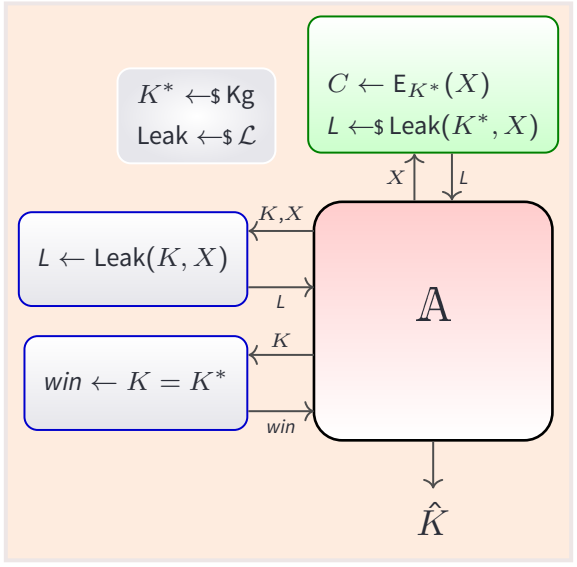
Emulating the cost of key enumeration

Some algorithmic approaches

- lore Adding “guessing entropies”
- 2013 Tree-like recursion algorithm
  [Veyrat-Charvillon, Gérard, Renauld, Standaert / Eurocrypt]
- 2015 Dynamic programming enabling parallellization
  [Martin, O’Connell, Oswald, Stam / Asiacrypt]
- 2015 Convolution of histograms
  [Glowacz, Grosso, Poussier, Schüth, Standaert / FSE]
  [Bernstein, Lange, van Vredendaal / eprint]

Key Rank Distributions

Martin, Mather, Oswald, Stam / Asiacrypt'16



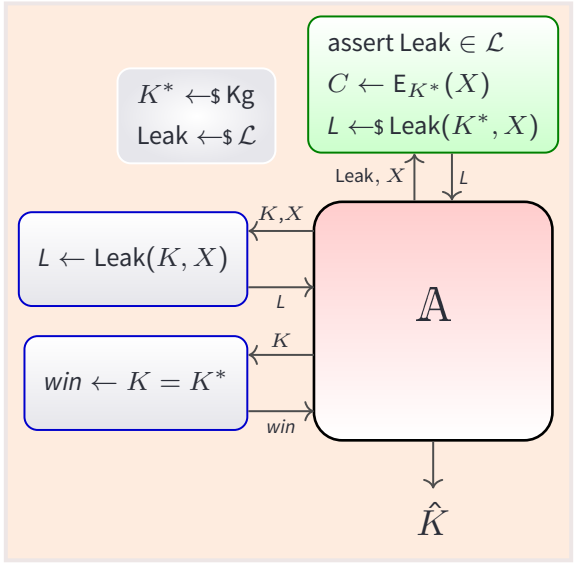
The Rank Distribution

Evaluator's task for some keyed device:

How long will it roughly take to recover the key as a function of the number of traces?

Key Rank Distributions

Martin, Mather, Oswald, Stam / Asiacrypt'16

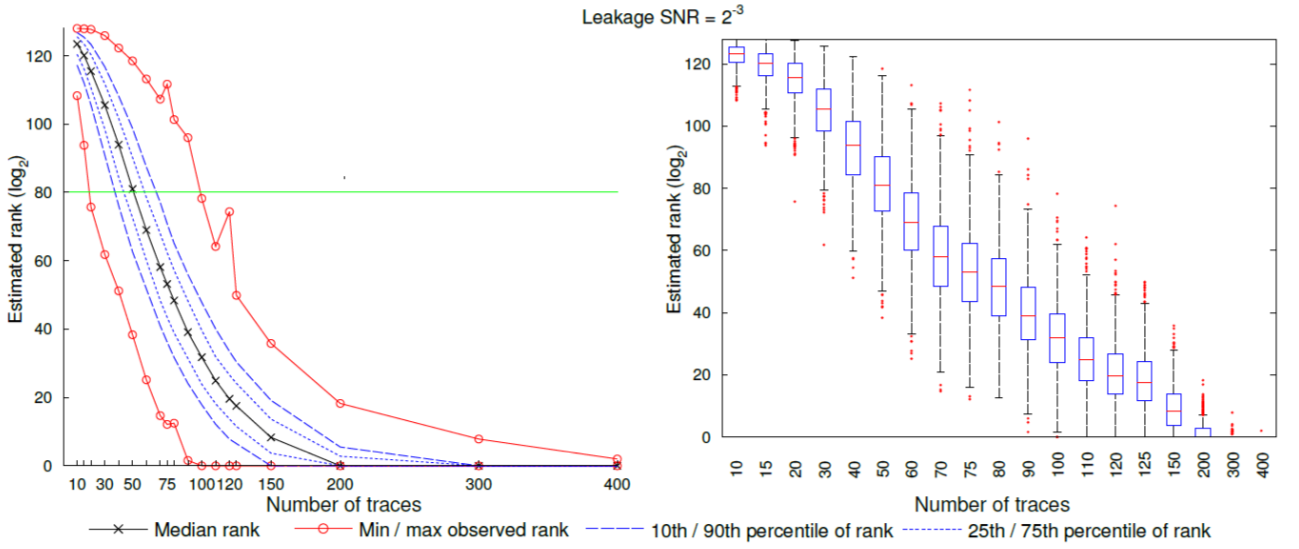


MMOS Setup

- AES-128 with simulated leakage
- Sbox output Hamming weight with Gaussian noise
- For SNRs 2^x with $x \in \{-7, -5, -3\}$
- Ran an unprofiled Correlation Power Attack (CPA)

Key Rank Distributions

Martin, Mather, Oswald, Stam / Asiacrypt'16

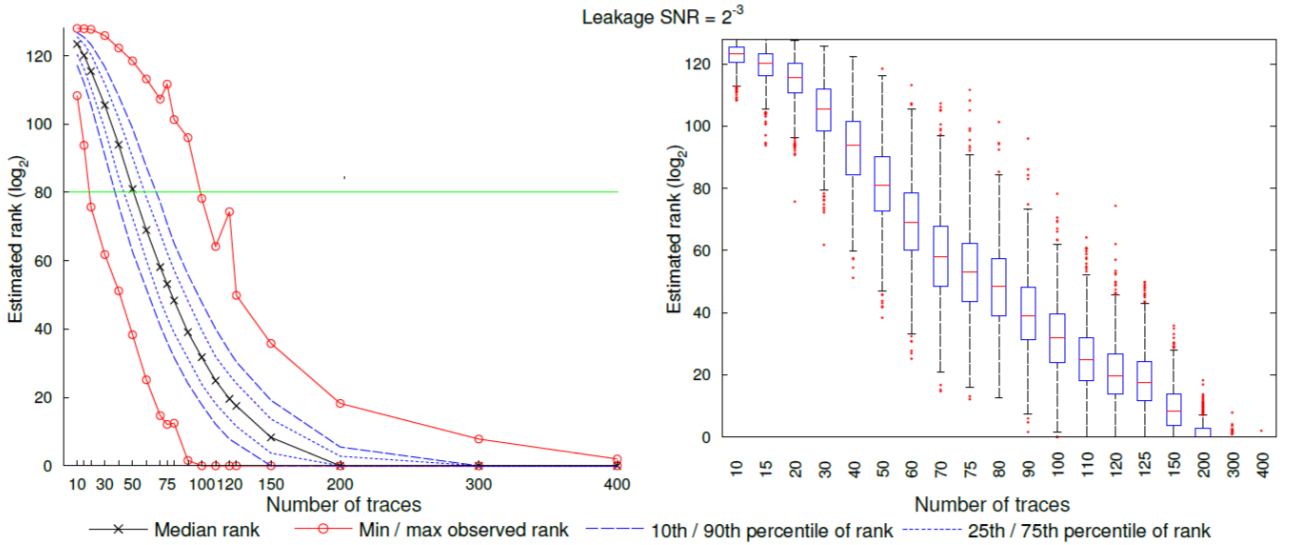


MMOS Lessons

1. Average log rank is more useful than log of average rank
   geometric mean versus arithmetic mean
2. The variance in the rank is considerable, esp. in the middle
3. SNR does not affect the shape of the distribution beyond scaling *x*-axis

Key Rank Distributions

Martin, Mather, Oswald, Stam / Asiacrypt'16



Challenges

1. Improved sensor fusion to combine subkey scores
2. Optimize distinguishers w.r.t. resulting key ranks
   - Model and feature selection
   - Score computation
3. Rank distribution against various countermeasures

SCALE: A Resource by Dan Page

https://github.com/danpage/scale

Side-Channel Attack Lab. Exercises

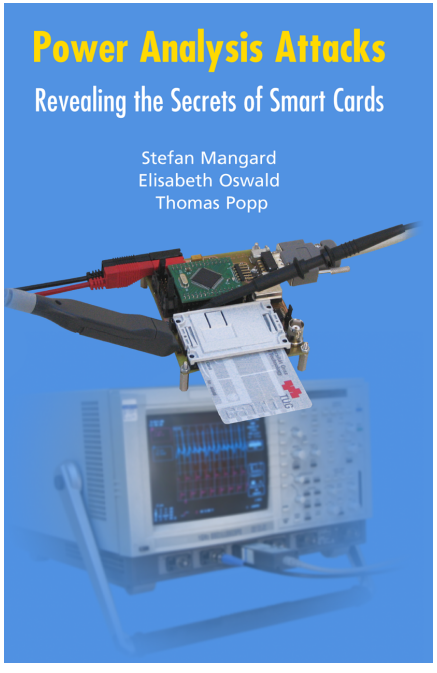
Provides a suite of material related to side-channel (and fault) attacks that is low-cost, accessible, relevant, coherent, and effective.

SCALE Data Sets

1. Four platforms: an *Atmel atmega328p* (an AVR) plus three NXP ARM Cortex-M processors
2. Implementation uses an 8-bit datapath and look-up tables for the S-box and xtime operations (but code not known)
3. 2×1000 traces of AES-128 each (known vs. unknown key)
4. Traces acquired using a Picoscope 2206B, using triggers for alignment

Power Analysis Attacks

Stefan Mangard, Elisabeth Oswald, and Thomas Popp's Classic



Revealing the Secrets of Smart Cards

- "first comprehensive treatment of power analysis attacks and countermeasures"
- Aimed at the practitioner
- From 2007 $\Rightarrow$ no modern ideas and theory

CHES

An IACR Conference

Cryptographic Hardware and Embedded Systems

Established in 1999

- Efficient implementations
- How to mount implementation attacks
- How to protect against them
- New designs that allow efficient yet secure implementations

https://ches.iacr.org