Characterisation and Estimation of the Key Rank Distribution in the Context of Side Channel Evaluations

`https://github.com/bristol-sca/labynkyr`

December 7, 2016

Daniel P. Martin, Luke Mather, Elisabeth Oswald, Martijn Stam

Cryptography Group, University of Bristol

OUTLINE

- **Claim**: we're not evaluating the resistance of a device to non-invasive side-channel attacks as well as we could be.
- **This work**: outline the reasons why, and describe an improved evaluation methodology.

WHY IS THIS IMPORTANT?

Being able to accurately evaluate the resistance of a device to SCA is important:

- Resistance to SCA encoded in several evaluation processes (Common Criteria; FIPS 140-3);
- Billions of devices implementing cryptography;
- SCA is almost always probabilistic in nature;
- Have to make a value judgement on the strength of an attacker that captures this probabilistic nature.

PLAN

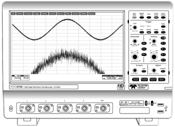
Motivation: we need to change how we view the outcome of a (non-invasive) side-channel attack

1. How we view side-channel attacks at the moment;
2. The current evaluation strategy;
3. Changing our view to include the rank of a side-channel attack;
4. This work: how do we appropriately modify our evaluation methodology.

CURRENT MODEL FOR A SIDE-CHANNEL ATTACK

Gather *n* measurements (traces)

Configure attack
- Pick model for leakage
- Choose distinguisher



Run attack

Candidate key k

Check using a known plaintext/ciphertext pair
$c == Enc_k(p)$?

FACTORS AFFECTING SUCCESS

The quality of an attack is affected by:

1. The nature of the ‘true’ underlying leakage signal;
2. The quality of the adversary’s model for that leakage;
3. The statistical technique used to assign scores to key candidates;
4. Noise: environmental, countermeasures, measurement quality;
5. The number of measurements available.

CURRENT EVALUATION APPROACH

Attack-based evaluation approach:

- Run a battery of attacks, and see what happens.

Judge impact of attack outcomes:

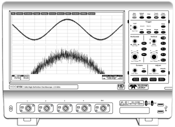
1. Does the adversary recover the secret key?
2. If yes, how many measurements were needed?
3. (other properties assessed: time, expense, ...)

SIDE-CHANNEL ATTACKS: WITH KEY RANK

Veyrat-Charvillion (SAC 2012) noticed that the adversary doesn't need the attack to be "perfect":

Gather *n* measurements (traces)

Configure attack
- Pick model for leakage
- Choose distinguisher



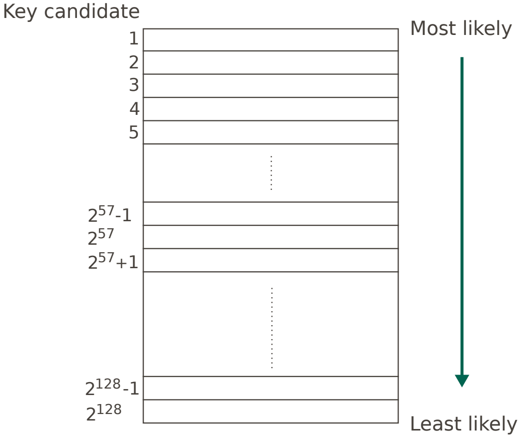
Run attack

Auxiliary information assigning 'scores' to **all** key candidates

Enumerate and check the key candidates in order of their score

KEY RANK: A DEFINITION

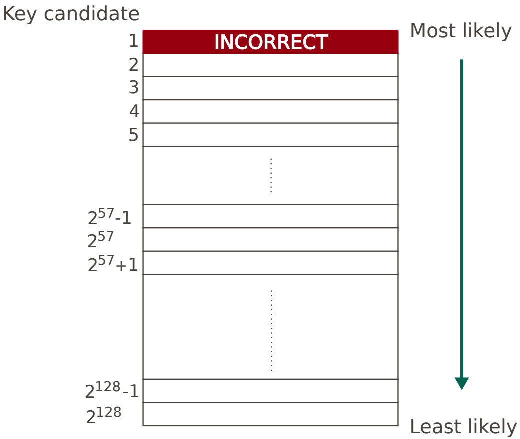
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

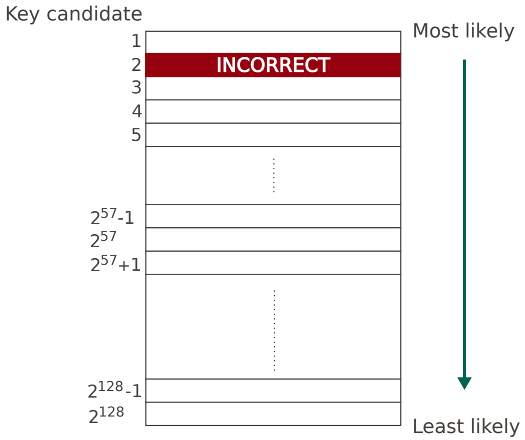
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

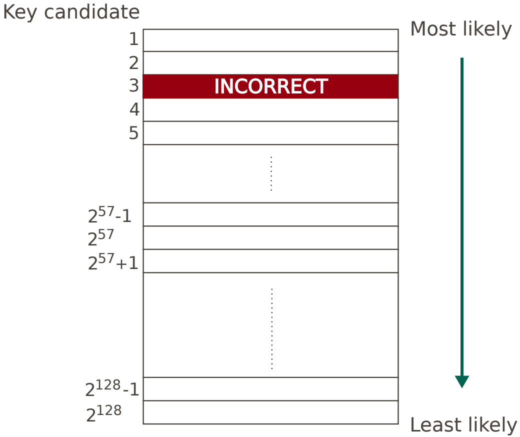
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

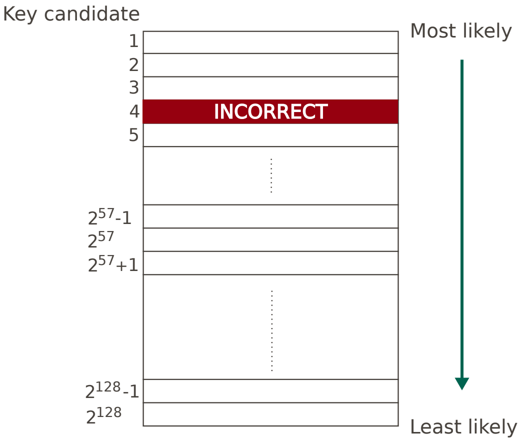
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

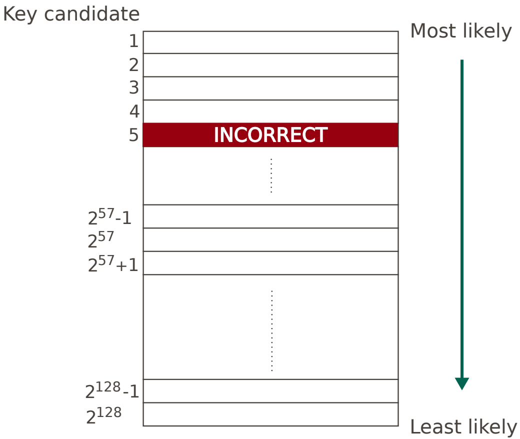
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

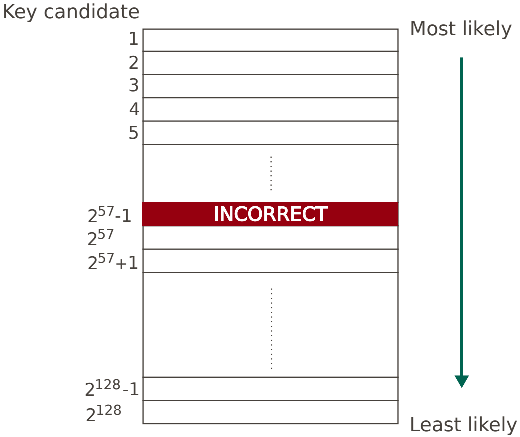
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

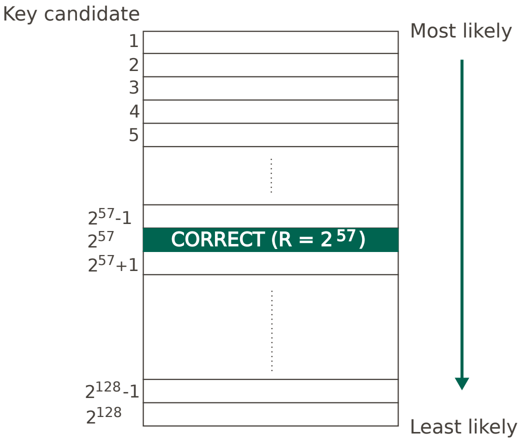
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

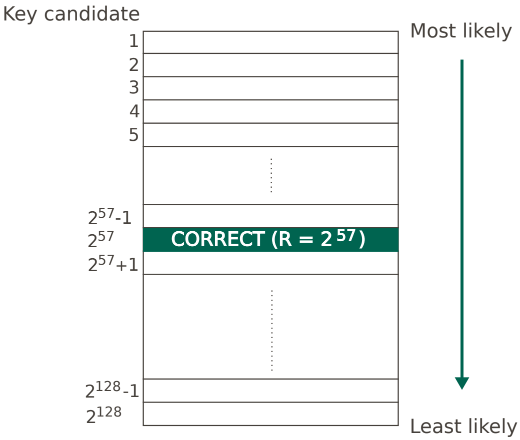
Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

KEY RANK: A DEFINITION

Rank **R**: the number R of candidate keys an adversary must enumerate and check before generating the correct key.



Note: enumeration of candidate keys is expensive!
(see ePrint 2016/609)

Check each candidate key *k* by encrypting a known plaintext ciphertext pair *(p,c)*

EVALUATION: WITH KEY RANK

Which is more powerful:

An attack requiring 10,000 measurements with a rank of 2^{55}?

or

An attack requiring 50,000 measurements with a rank of 2^{53}?

KEY RANK AS A RANDOM VARIABLE

Key rank R is a random variable defined over the randomness in (a fixed number of) measurements

Can we analytically compute the distribution of R?

Answer: no, in practice we don't know all the distributions involved.

Later: is looking at the expectation $\mathbb{E}(R)$ a good idea?

ESTIMATING THE RANK DISTRIBUTION

The only usable approach is to estimate the rank distribution using repeated sampling.

1. Fix an attack strategy, and a number of measurements n.
2. Capture a fresh set of n measurements, and run the attack.
3. Compute or estimate the rank for that attack.
4. Repeat.

Questions we wanted to answer:

1. What is the shape of the distribution?
2. Is there consistency across the spectrum of SCA scenarios?

COMPUTING RANK

Need a non-naive method for approximating rank when key is known.

Care about speed and minimising the error in bits b: if the true rank is 2^x, then the estimate is within $2^{x \pm b}$.

Majority of existing attempts provide an estimate for an interval:

- Veyrat-Charvillon et al. (Eurocrypt 2013)
- Glowacz et al. (FSE 2015) (*)
- Duc et al. (Eurocrypt 2015)
- Bernstein et al. (ePrint 2015/221)

We chose to look at optimising:

- Martin et al. (Asiacrypt 2015)

IMPROVING RANK ESTIMATION

Made several observations to reduce the run-time of Martin et al. rank estimation algorithm.

- Able to achieve $\sim 8 - 10$ orders of magnitude more precision at no additional cost
- $\Rightarrow$ can get a very accurate point estimate in a few seconds on a workstation CPU.

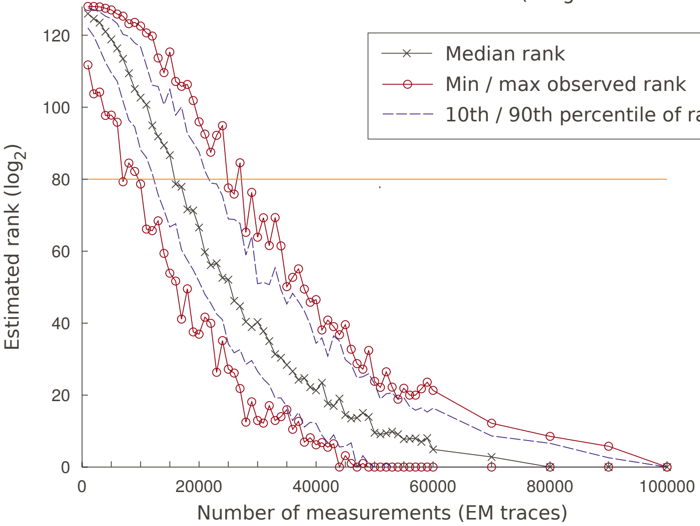
RESULTS: DISTRIBUTION CONSISTENCY

In general, distribution and shape of R is very consistent.

- Performed hundreds of thousands of repeat experiments across a variety of:
-
  - Noise levels;
  - Distinguisher types;
  - Leakage distributions;
  - Quantities of measurements.
- ... estimating and recording the rank after each attack.

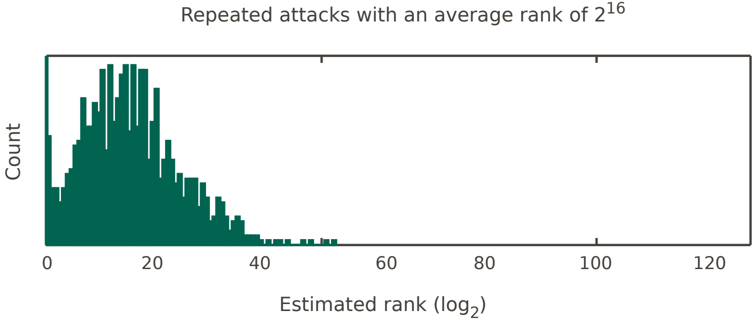
RESULTS: DISTRIBUTION SHAPE -- REAL WORLD EXPERIMENT

Repeated DPA attacks on a BeagleBone Black implementing AES-128 in hardware
(Longo et al. CHES 2015)



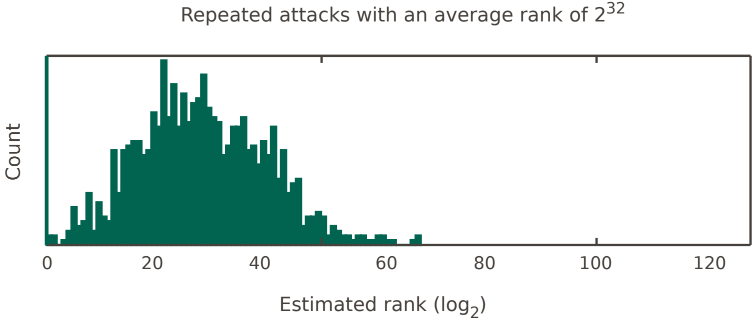
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



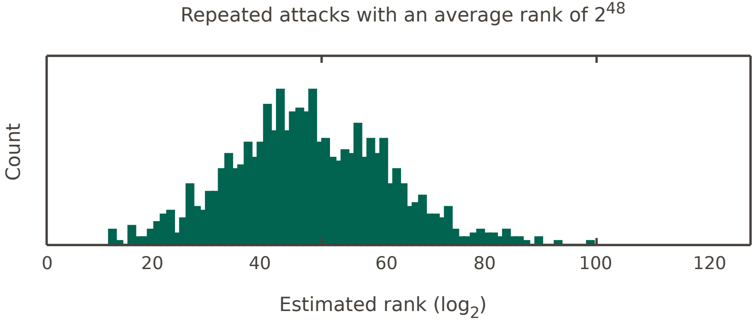
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



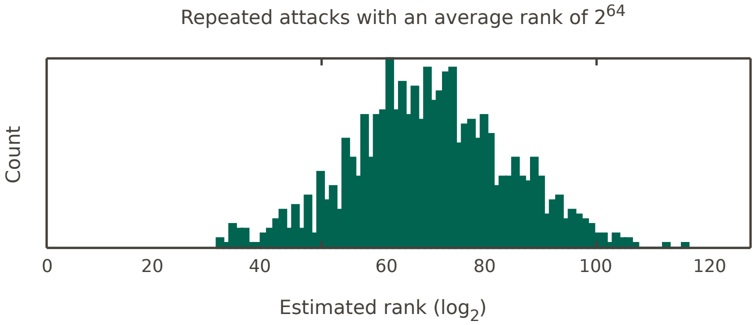
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



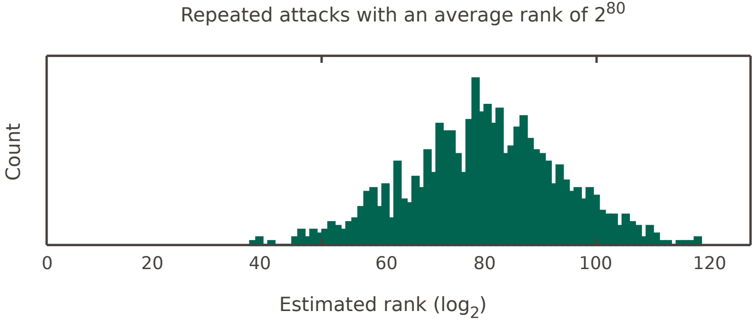
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



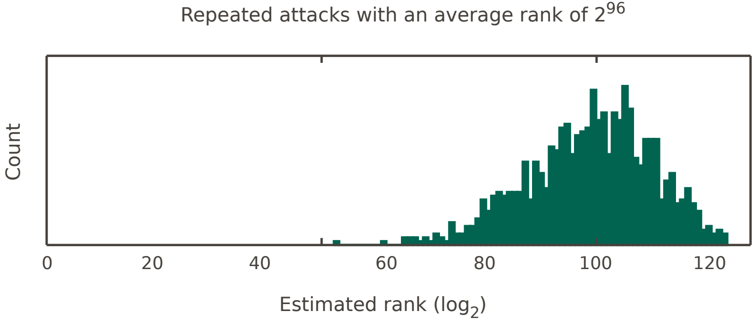
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



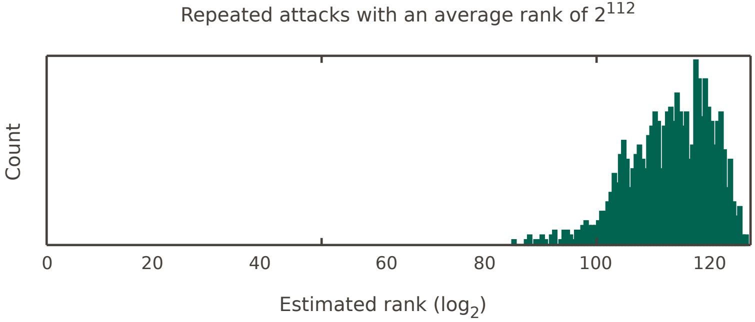
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



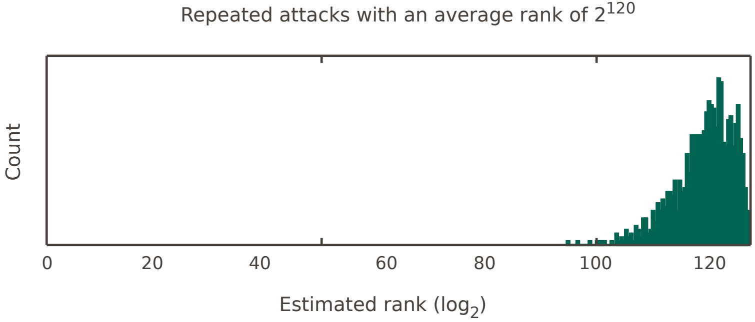
RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



RESULTS: DISTRIBUTION SHAPE -- IN DETAIL

Histograms of repeated attacks grouped by mean rank:



EVALUATION PROPOSAL

- Repeated sampling from the rank distribution of an attack is the only approach.

Statistic choice:

- Large variance in distribution means averages are not particularly useful statistics;
- Non-parametric order statistics such as percentiles are ideal: e.g estimated 10% chance my devices are vulnerable to an attack of rank $\leq 2^{80}$.

Estimation stability (discussed in paper):

- Need to run at least 30 repeat experiments.

This put stress on the measurement gathering phase:

- If you're careful, you can be clever with measurement collection.

CONCLUSION

Thanks for listening!

Rank estimation and enumeration code (C++11):

`https://github.com/bristol-sca/labynkyr`

MIT-style licence

An analysis of enumeration capabilities:

`http://eprint.iacr.org/2016/609`