Trace Augmentation: What Can Be Done Even Before Preprocessing in a Profiled SCA?

Sihang Pu[1] Yu Yu[1] Weijia Wang[1] Zheng Guo[1]
Junrong Liu[1] Dawu Gu[1] Lingyun Wang[2] Jie Gan[3]

Shanghai Jiao Tong University, Shanghai, China
Email: {push.beni, yyuu, aawwjaa, guozheng, liujr, dwgu}@sjtu.edu.cn

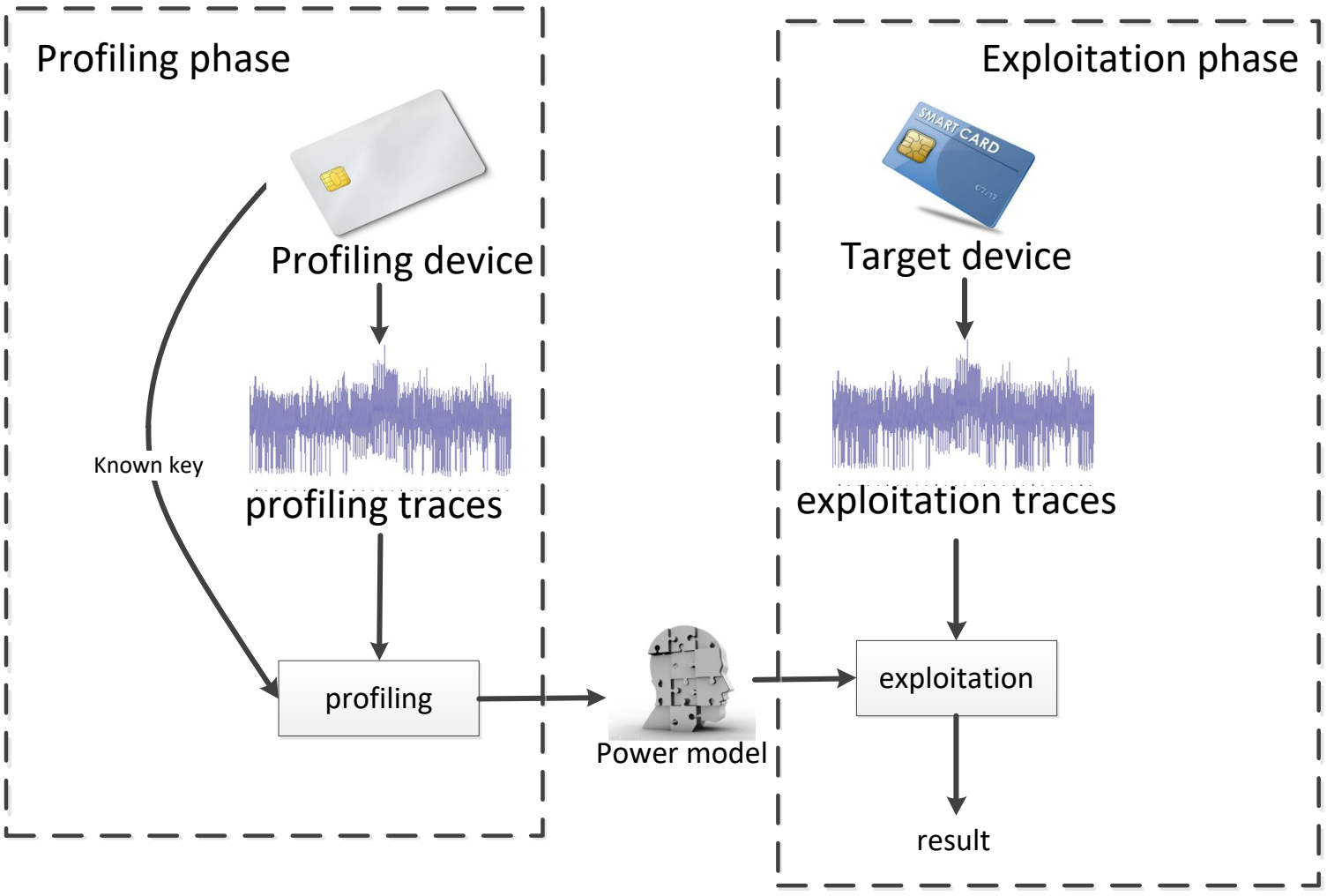
Shanghai Viewsource Information Science and Technology Company, China
Email: lingyun.wang@viewsources.com

Beijing Smart-Chip Microelectronics Technology Co., Ltd. Beijing, China
Email: ganjie@sgitg.sgcc.com.cn

CARDIS, 2017

Outline

Profiled SCA



Profiled SCA

Profiled side-channel attack:

- Adds a profiling phase to the original SCA
- Can be considered as the powerful class of power analysis

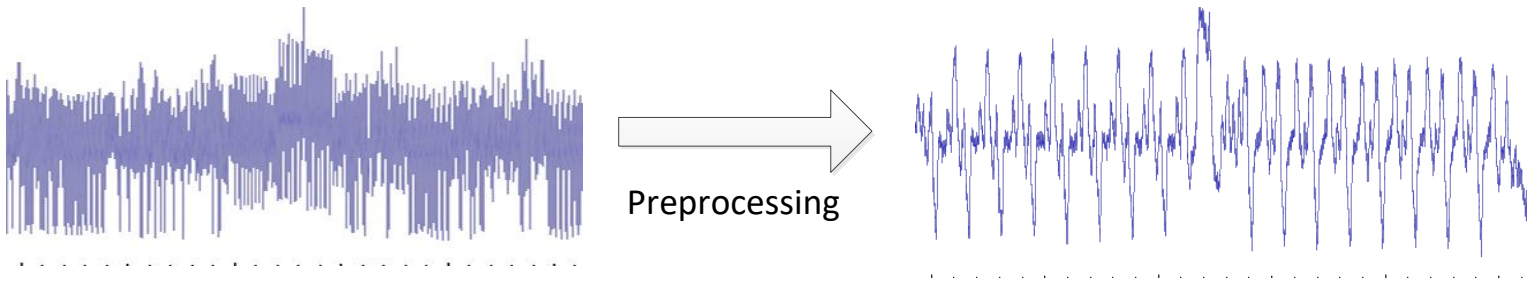
But:

- It presumes high similarity between profiling device and target device
- Otherwise, result in less desirable performance
- All existing solutions [1, 2] mainly focus on the dedicated designing of profiling method

 [1] C. Whitnall, E. Oswald and etl.
Robust Profiled DPA, *CHES 2015*.

 [2] W. Wang, Y. Yu, F.-X. Standaert and etl.
Ridge-Based Profiled Differential Power Analysis, *CT-RSA 2017*.

Preprocessing techniques



Good examples:

- Lowpass
- Principal component analysis (PCA)
- Linear discriminant analysis (LDA)
- Singular spectrum analysis (SSA)

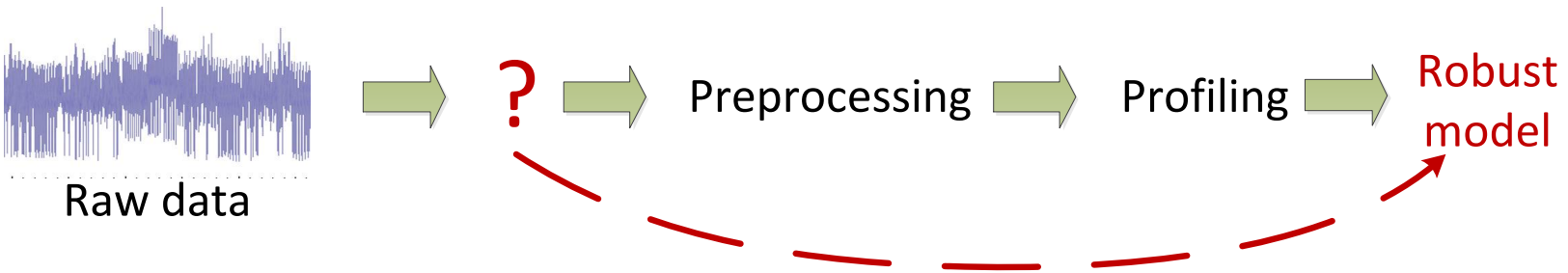
These techniques mainly focus on ameliorating the SNR.

But:

- They may not work well with deviated target devices;

Motivation

What can be done during (or even before)
the preprocessing stage to make the subsequent attacks more robust?

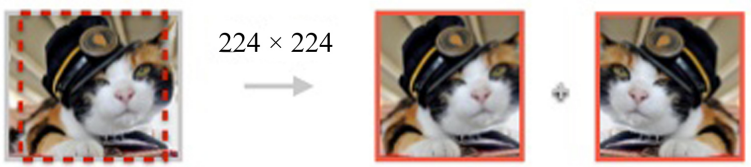


Data augmentation

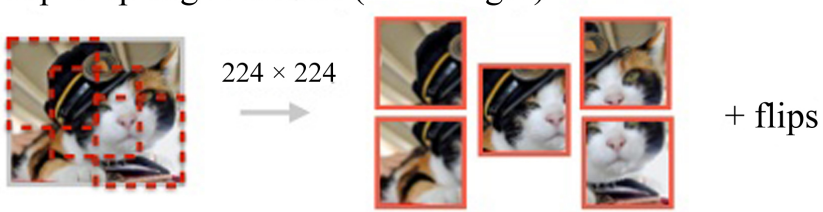
- It is popular in the deep learning community
- It is used to mitigate overfitting and build a more robust model

a. No augmentation (=1 image)

b. Flip augmentation (=2 images)



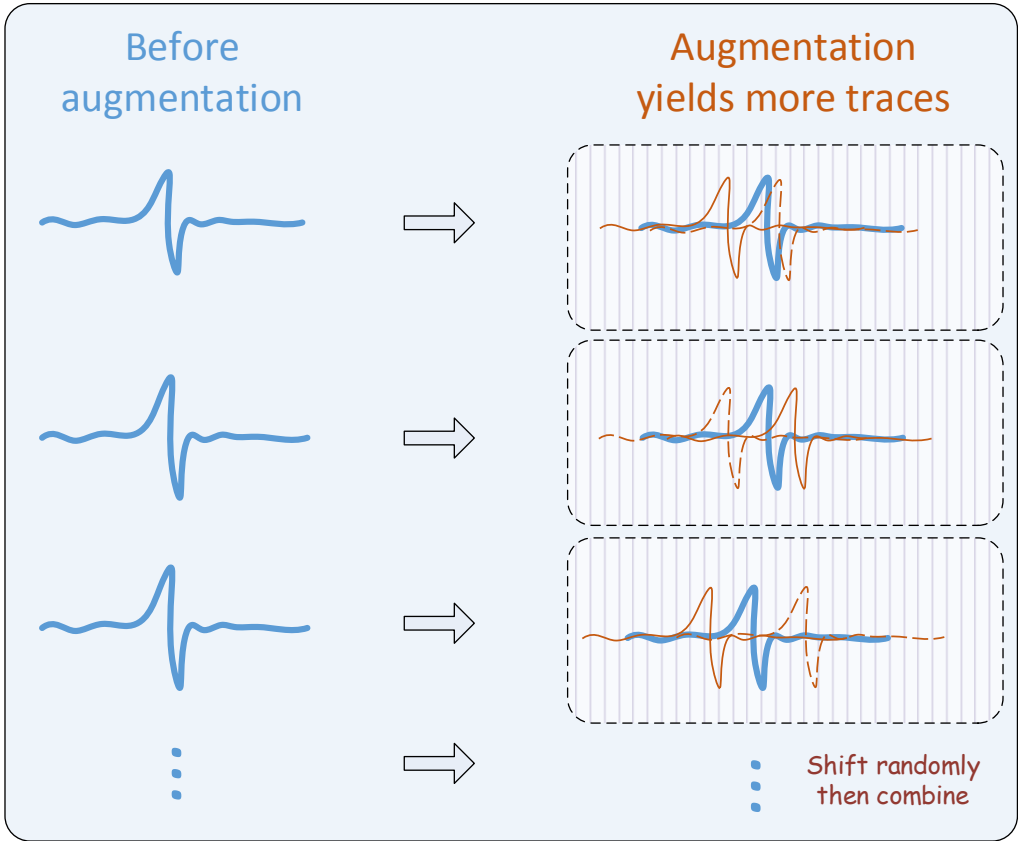
c. Crop+Flip augmentation (=10 images)



Trace augmentation through random shift

1. Shift each trace horizontally at random.
2. Repeat step 1 several times to yield many perturbed traces.
3. Append these new perturbed traces to original set.

A visual illustration of trace augmentation



Parameters

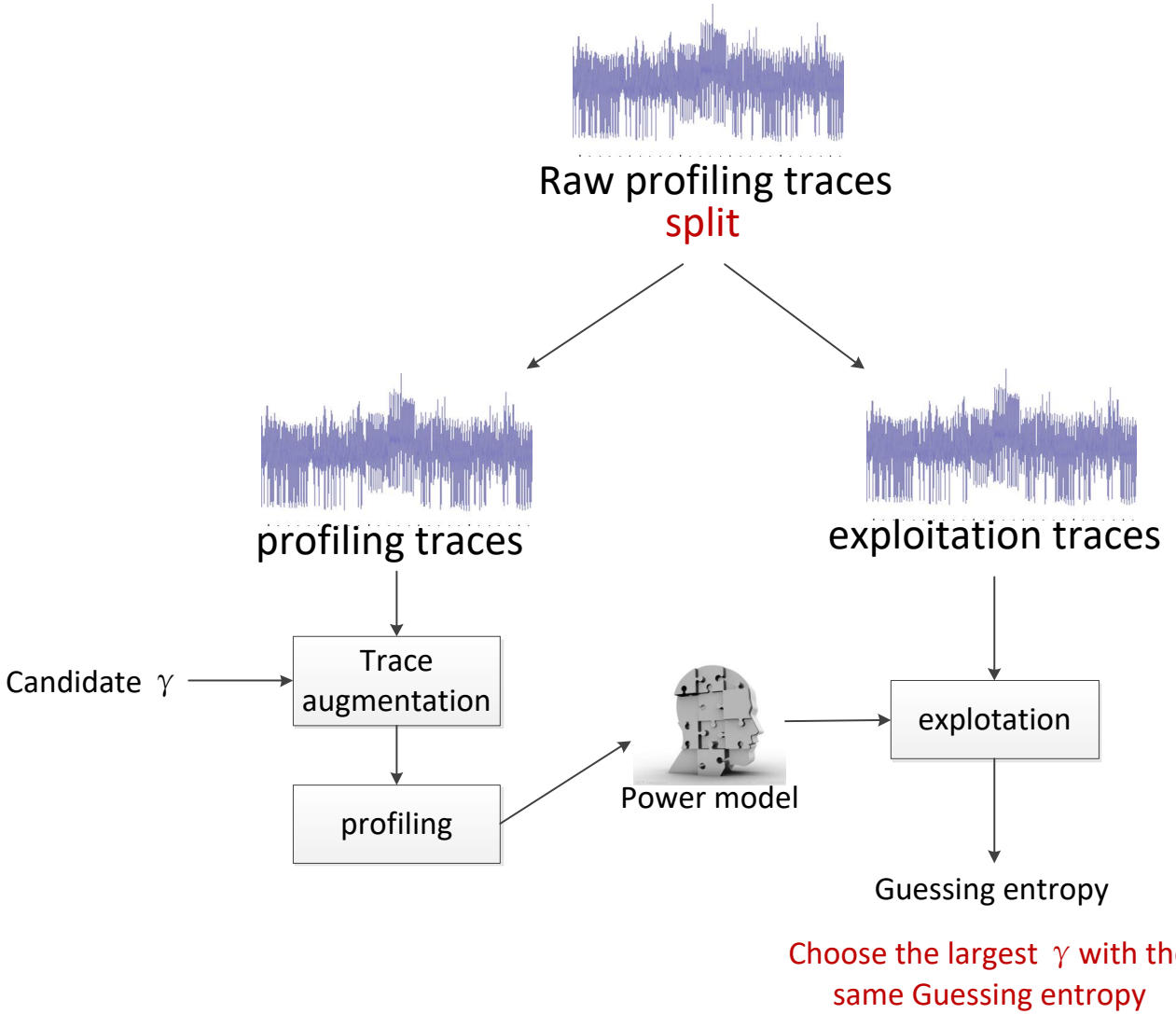
- Shift ratio, γ: quantifies the extent of random shift: perturb each trace with a horizontal random shift drawn uniformly from $[-\gamma \cdot d, \gamma \cdot d]$, where d denotes the number of leakage points
- Augmentation ratio, C: quantifies The number of expanded traces, $N_{new} = \gamma C \cdot N_{original}$

Trace augmentation through random shift

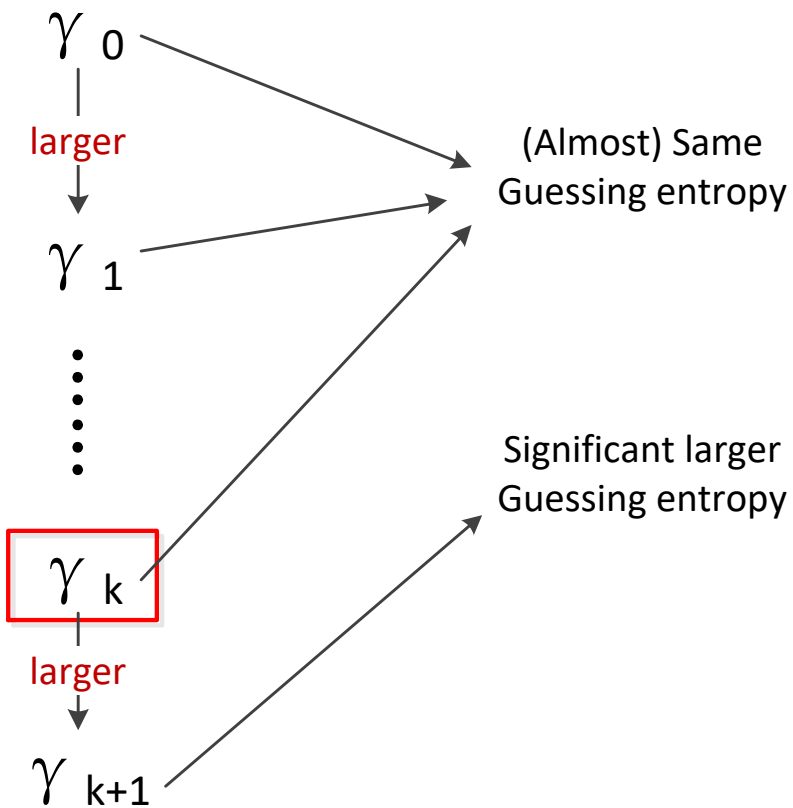
It is tricky to define a general rule to select the two undetermined parameters (i.e., γ and C).
But we noted that:

1. The augmentation ratio, C, does not depend much on the attack scenario, thus we simply choose $C = 10$;
2. The shift ratio, γ, affect the improvement of trace augmentation significantly, thus we design the lazy-validation.

Lazy-Validation



Lazy-Validation



Intuition: it is safe to perturb traces to some certain extent as long as it does not affect the performance in the ideal setting

Experiments

- Simulation-based experiment
- Practical experiments:
  - Software-based experiments
  - FPGA-based experiments

Simulating Settings

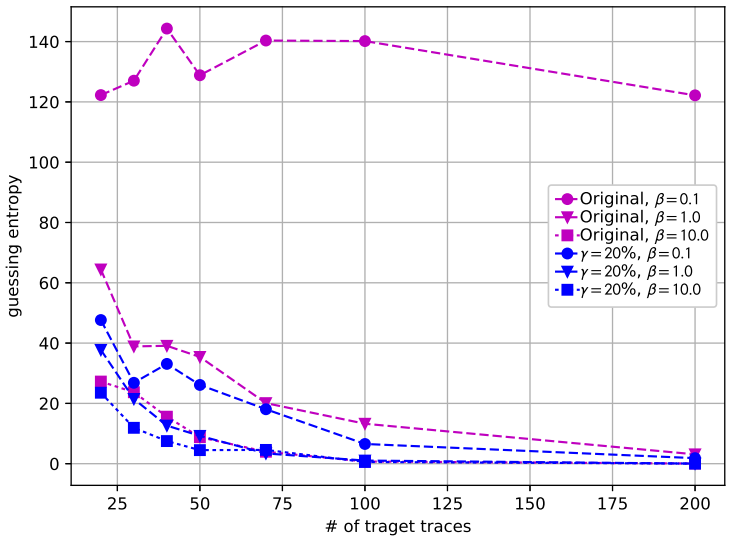
- Leakage is subject to the multi-variant Gaussian distribution
- Choosing means of the distribution by picking random real numbers
- Generating of covariance matrix: rely on a refined method named 'vine'
  - Parameter β: control the correlations between leakage points;
  - Smaller β value corresponds to the more dependencies among points of each trace
- Perfect case: the (simulated) traces of profiling and exploitation are subject to the similar distribution (same values of mean and parameter β).
- Imperfect case: perturb the (standardized) leakage function of the exploitation trace by imposing a 'noise' of uniform distribution $\mathcal{U}(-5, 5)$

Attack Settings

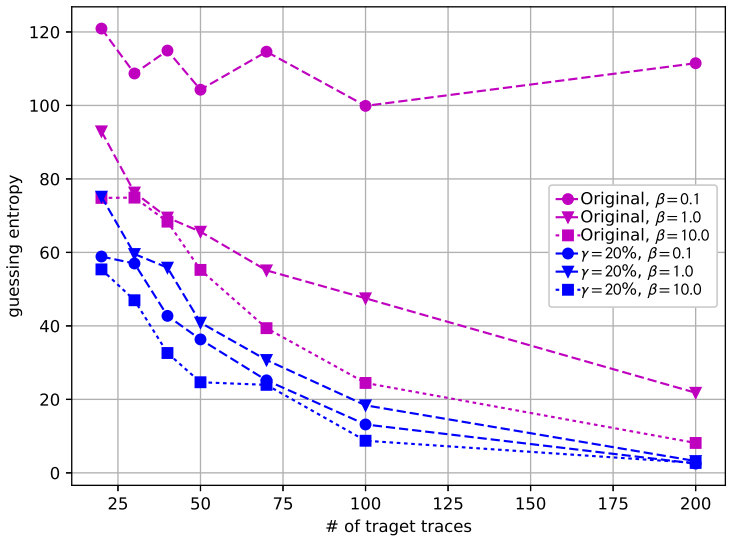
1. Trace augmentation (optional)
2. LDA to 1 point
3. Template building:
   - Gaussian template building
   - K-means

Simulating Experimental Results

Gaussian template building:



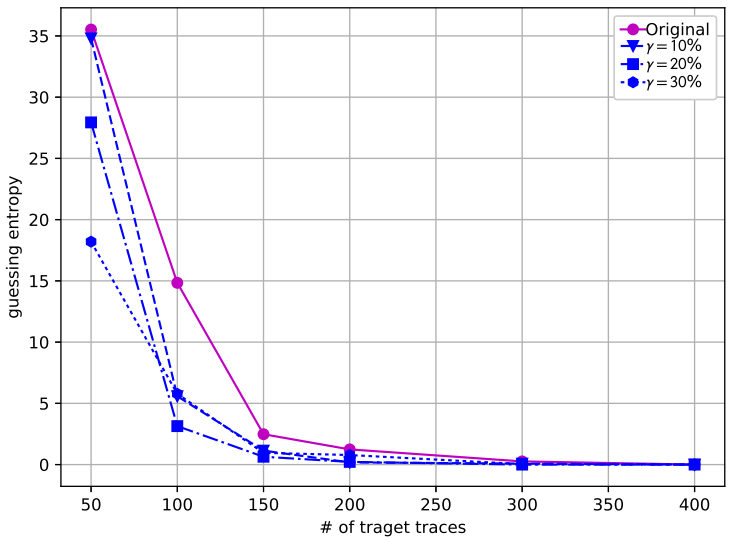
(a) perfect case (without discrepancy)



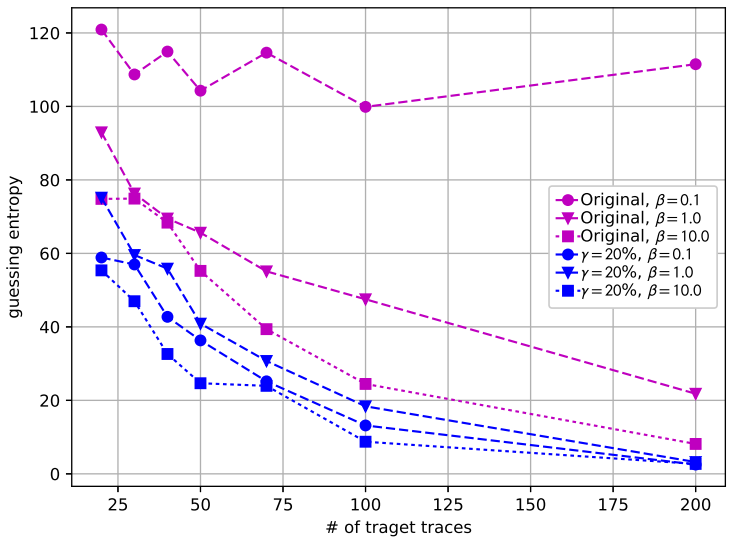
(b) introducing noise as discrepancy

Simulating Experimental Results

K-means:



(c) perfect case (without discrepancy)



(d) introducing noise as discrepancy

Target Settings

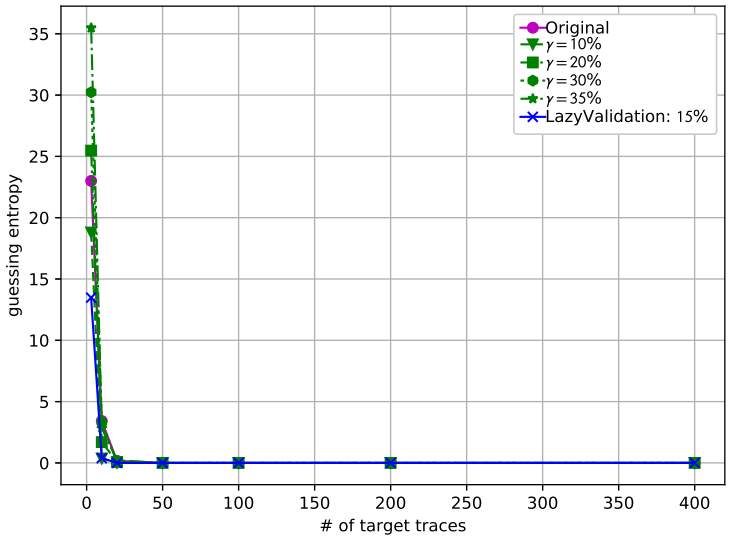
- Atmel ATMega-163
- Consist of 54 leakage points.

- Perfect case: the points of profiling and exploitation come from the same positions.
- Imperfect case: adding misalignment between profiling and target traces

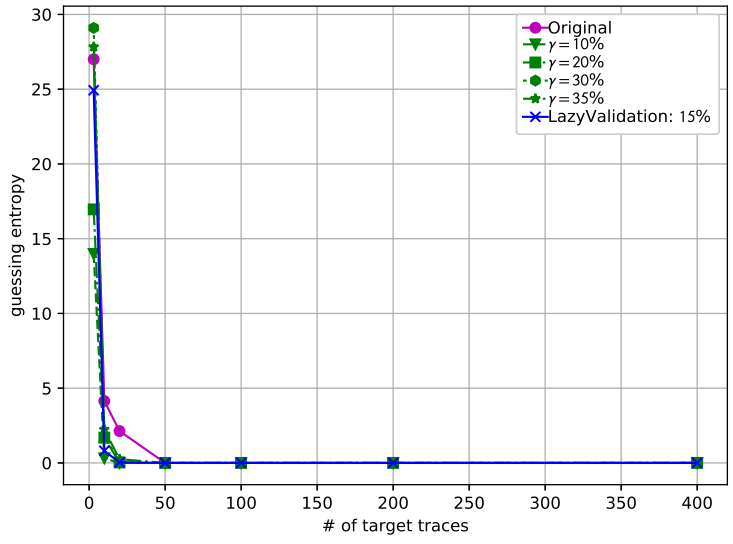
Attack Settings

1. Trace augmentation (optional)
2. PCA to 1 point
3. Gaussian template building

Software-based Experimental Results

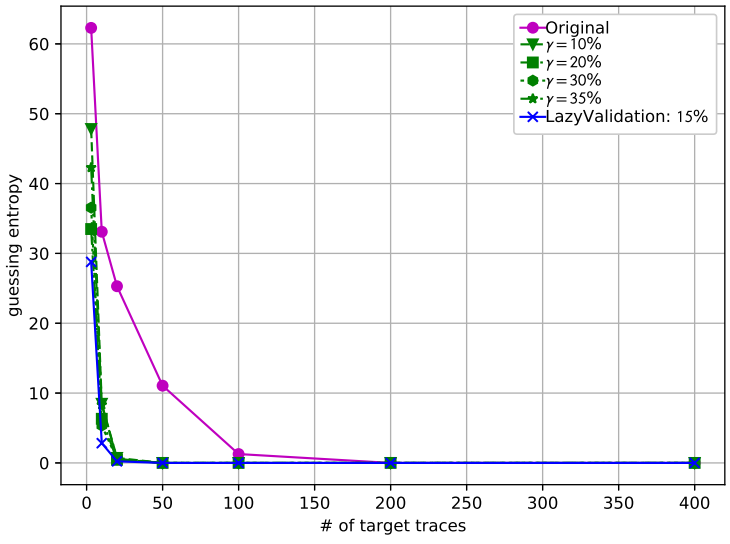


(e) no misalignment

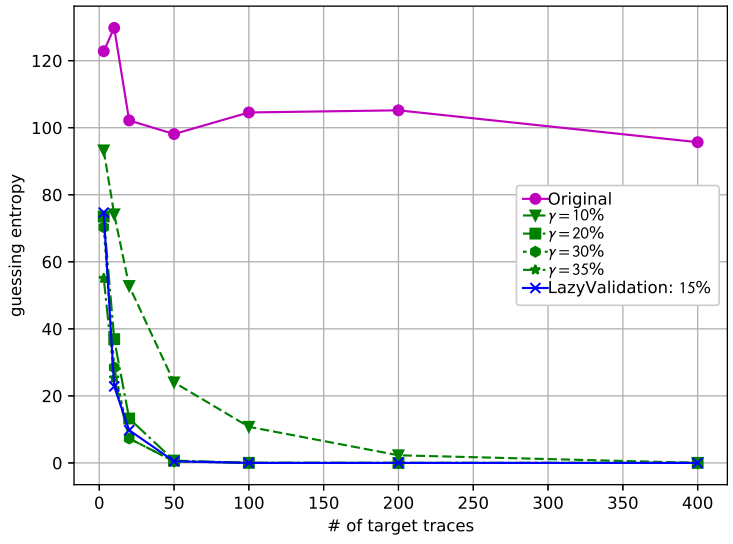


(f) misalignment = 5%

Software-based Experimental Results

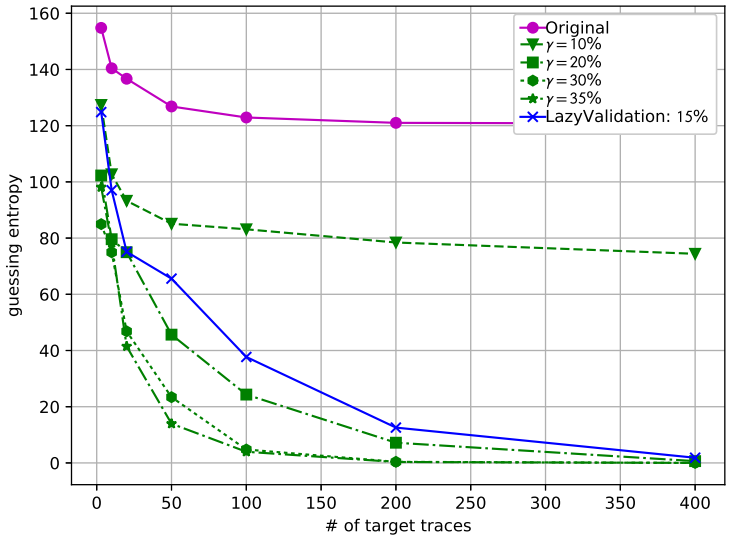


(g) misalignment = 10%

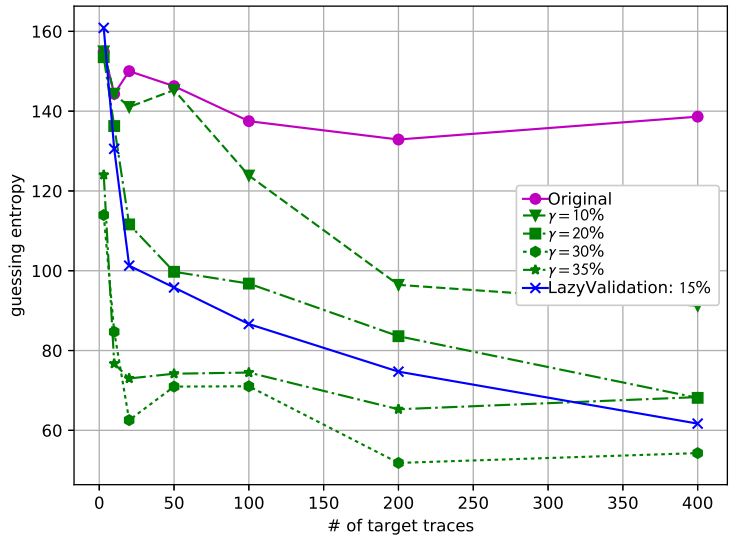


(h) misalignment = 15%

Software-based Experimental Results



(i) misalignment = 20%



(j) misalignment = 25%

Target Settings

- SAKURA-X board
- 20 leakage points

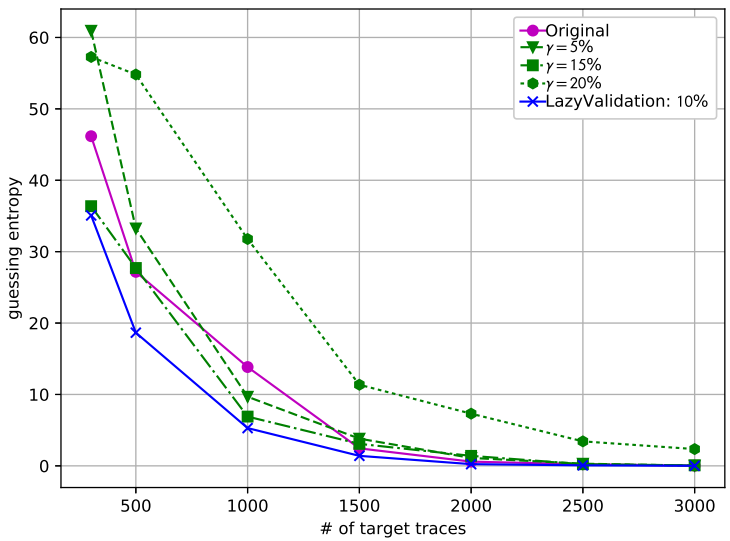
- Perfect case: the points of profiling and exploitation come from the same positions.
- Imperfect case: adding misalignment between profiling and target traces

Attack Settings

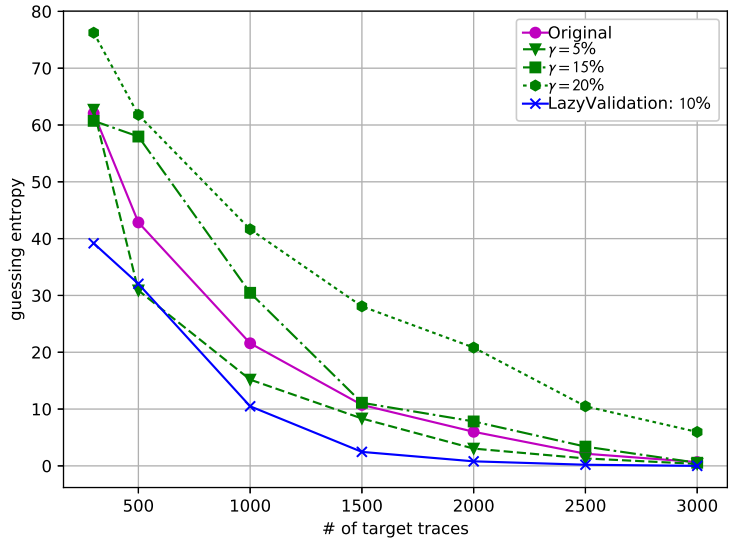
1. Trace augmentation (optional)
2. PCA to 1 point
3. Linear-regression based profiling, ridge-based profiling

FPGA-based Experimental Results

Linear-regression based profiling:



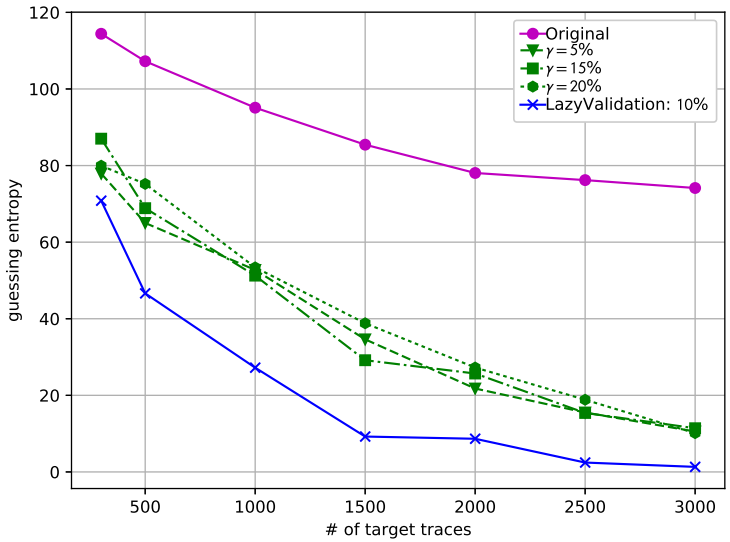
(k) no misalignment



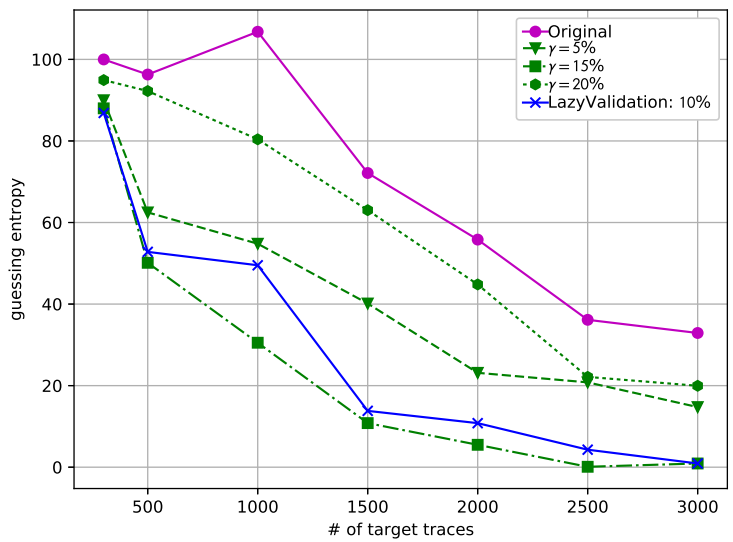
(l) misalignment = 5%

FPGA-based Experimental Results

Linear-regression based profiling:



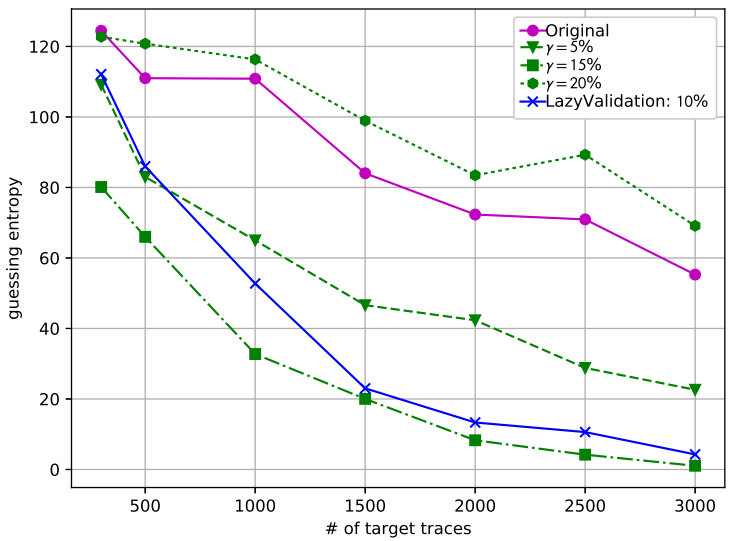
(m) misalignment = 10%



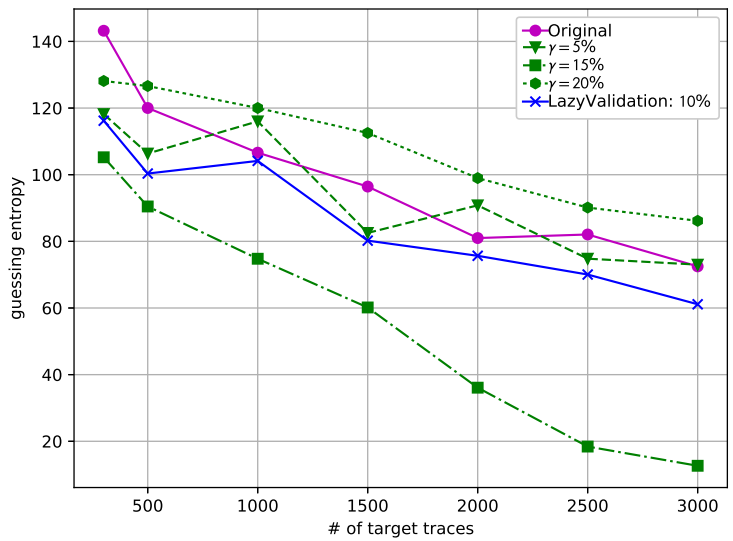
(n) misalignment = 15%

FPGA-based Experimental Results

Linear-regression based profiling:



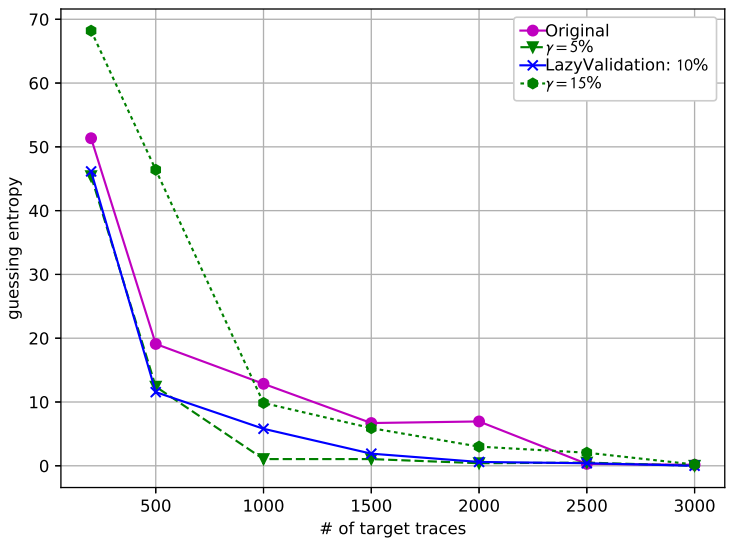
(o) misalignment = 20%



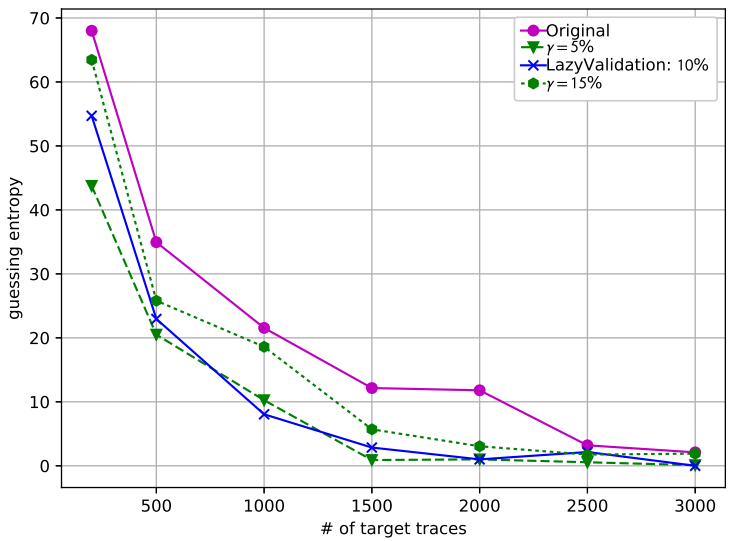
(p) misalignment = 25%

FPGA-based Experimental Results

Ridge-based profiling:



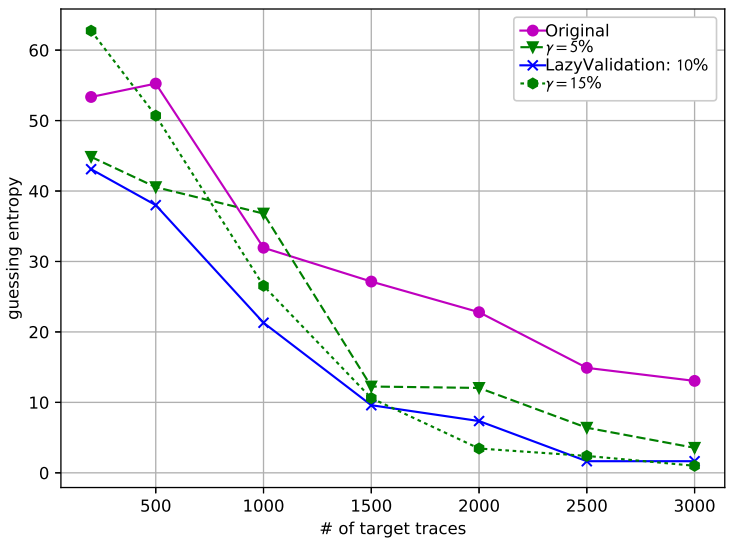
(q) no misalignment



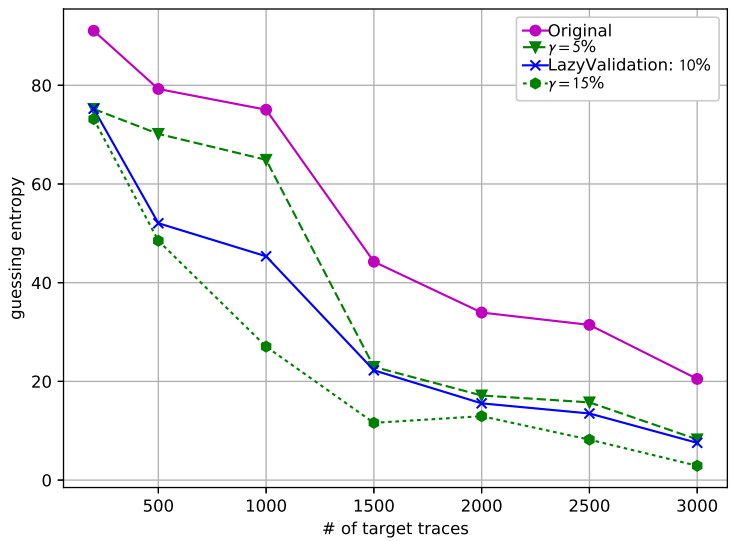
(r) misalignment = 5%

FPGA-based Experimental Results

Ridge-based profiling:



(s) misalignment = 10%



(t) misalignment = 15%

Summary

- Trace augmentation based preprocessing can effectively improve the performance of profiled SCA, in both of the scenarios where:
  - Profiling device behaves close to the target device
  - Profiling device deviates significantly from the target device
- Lazy-validation to obtain conservative but appropriate parameters
- Simulation-based and practical experiments
  - Confirm the effectiveness of our approach
  - The improvement depends on the correlation among points of each trace.

Future works

- Explore other possible ways to augment the trace.
- Why? How much improvement? When?

Questions?

Thanks for your consideration!
Questions?