A Comprehensive Study of Deep Learning for Side-Channel Analysis

Loïc Masure[1,3] Cécile Dumas[1] Emmanuel Prouff[2, 3]

[1]Univ. Grenoble Alpes, CEA, LETI, DSYS, CESTI, F-38000 Grenoble
loic.masure@cea.fr

[2]ANSSI, France

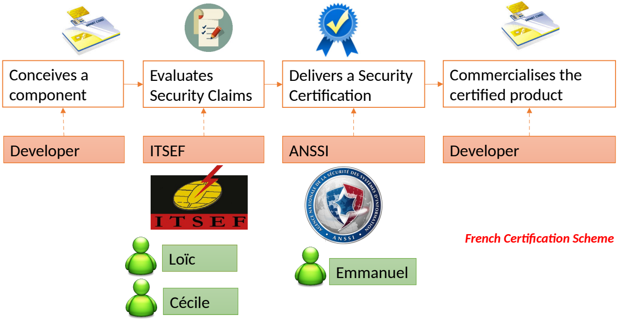
[3]Sorbonne Université, UPMC Univ Paris 06, POLSYS, UMR 7606, LIP6, F-75005, Paris, France

17/09/2020, CHES

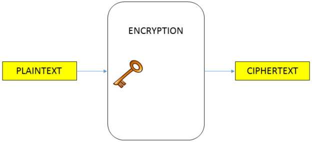
Outline

Who am I

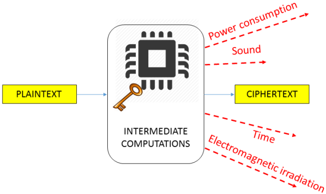
- PhD student, studying Deep Learning (DL) for Side-Channel Analysis (SCA)

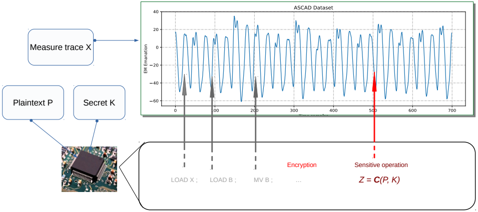


What is SCA?

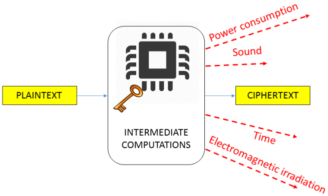


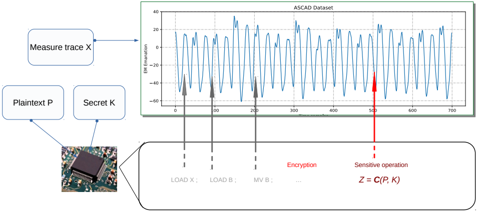
What is SCA?


PLAINTEXT
INTERMEDIATE COMPUTATIONS
CIPHERTEXT
Power consumption
Sound
Time
Electromagnetic irradiation


Measure trace X
Plaintext P
Secret K
ASCAD Dataset
EM Emanation
LOAD X ;
LOAD B ;
MV B ;
...
Encryption
Sensitive operation
Z = C(P, K)

What is SCA?





Profiling Attack

Attack using *open samples* similar to the target device – same code, same chip, *etc.* – with full knowledge of the secret key

Two steps:

- Profiling phase: P, K known $\implies$ Z known, $\mathbf{X}$ acquired on an open sample
- Attack phase: P known, $\mathbf{X}$ acquired on the *target* device, K guessed

FROM RESEARCH TO INDUSTRY

cea tech

leti

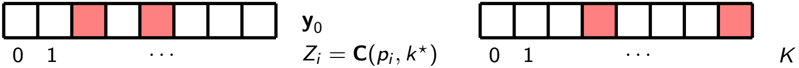
Outline

1. Context
2. SCA Optimization Problem versus Deep Learning Based SCA
3. NLL Minimization is PI Maximization
4. Simulation results
5. Experimental results

Profiling Attacks

Key Recovery (*i.e.* attack step)

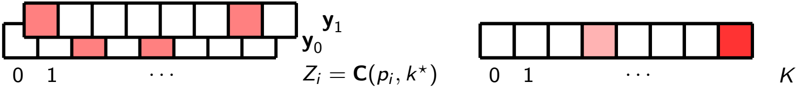
Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Profiling Attacks

Key Recovery (*i.e.* attack step)

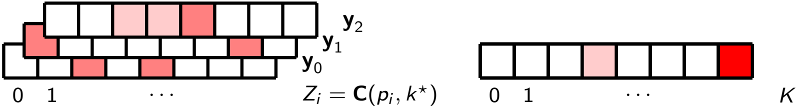
Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Profiling Attacks

Key Recovery (*i.e.* attack step)

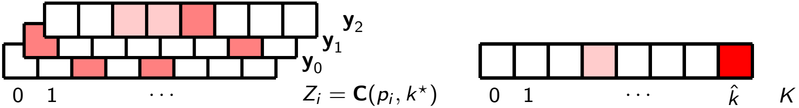
Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Profiling Attacks

Key Recovery (*i.e.* attack step)

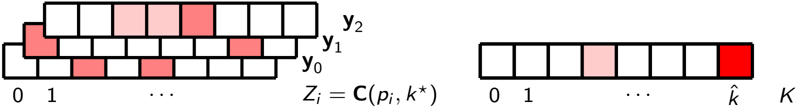
Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Profiling Attacks

Key Recovery (*i.e.* attack step)

Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$

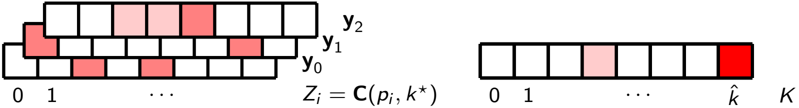


Goal: find F that minimizes N_a s.t. $\hat{k} = k^\star$ with probability $\geq \beta$ (*e.g.* 0.9)

Profiling Attacks

Key Recovery (*i.e.* attack step)

Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$

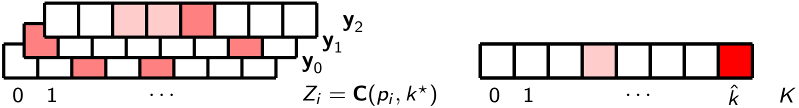


Goal: find F that minimizes N_a s.t. $\hat{k} = k^\star$ with probability $\geq \beta$ (*e.g.* 0.9)
Optimal model: $F^\star$, with $N_a^\star$ traces

Profiling Attacks

Key Recovery (*i.e.* attack step)

Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Goal: find F that minimizes N_a s.t. $\hat{k} = k^\star$ with probability $\geq \beta$ (*e.g.* 0.9)
Optimal model: $F^\star$, with $N_a^\star$ traces

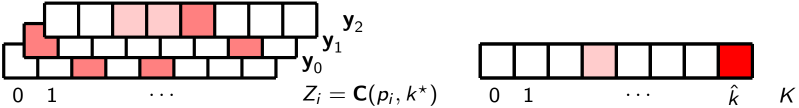
How to find $F^\star \implies$ profiling step

Requires to know the probability distribution $F^\star = \Pr[Z|\mathbf{X}]$

Profiling Attacks

Key Recovery (*i.e.* attack step)

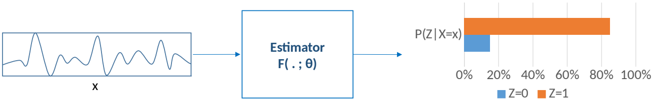
Given N_a attack traces $\mathbf{x}_i$ with plaintext p_i, calculate *scores* $\mathbf{y}_i = F(\mathbf{x}_i)$



Goal: find F that minimizes N_a s.t. $\hat{k} = k^\star$ with probability $\geq \beta$ (*e.g.* 0.9)
Optimal model: $F^\star$, with $N_a^\star$ traces

How to find $F^\star \implies$ profiling step

Requires to know the probability distribution $F^\star = \Pr[Z|\mathbf{X}]$
Reality: unknown for the evaluator/attacker. Estimation with parametric models $F(., \theta)$:



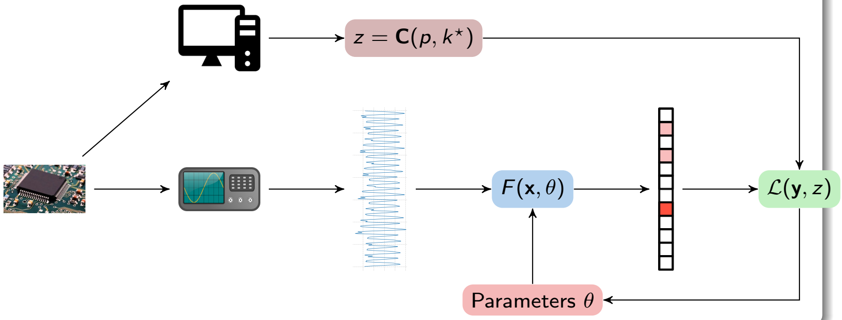
Deep Learning (DL) based SCA is a hot topic currently

Recent milestones about its effectiveness: more robust against counter-measures like masking [MPP16], jitter (misalignment) [CDP17], whether on software or FPGA [Kim+19]

Deep Learning (DL) based SCA is a hot topic currently

Recent milestones about its effectiveness: more robust against counter-measures like masking [MPP16], jitter (misalignment) [CDP17], whether on software or FPGA [Kim+19]

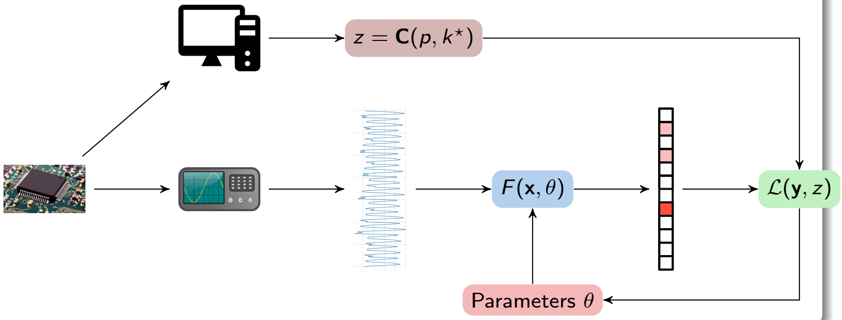
Training a Neural Network



Deep Learning (DL) based SCA is a hot topic currently

Recent milestones about its effectiveness: more robust against counter-measures like masking [MPP16], jitter (misalignment) [CDP17], whether on software or FPGA [Kim+19]

Training a Neural Network



$\mathcal{L}$: performance metric (accuracy, recall, ...) or loss function (Mean Square Error, NLL, ...)

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\Longrightarrow$ successful key recovery
- *Low* accuracy $\Longrightarrow$ nothing

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery
- *Low* accuracy $\implies$ nothing, problem: often happens (*e.g.* highly noisy leakages)

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery
- *Low* accuracy $\implies$ nothing, problem: often happens (*e.g.* highly noisy leakages)
- *Apparently, no other machine learning metric related to SCA metrics*

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery
- *Low* accuracy $\implies$ nothing, problem: often happens (*e.g.* highly noisy leakages)
- *Apparently, no other machine learning metric related to SCA metrics*

Accuracy: find β s.t. $N_a^\star = 1$

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery
- *Low* accuracy $\implies$ nothing, problem: often happens (*e.g.* highly noisy leakages)
- *Apparently, no other machine learning metric related to SCA metrics*

Accuracy: find β s.t. $N_a^\star = 1$ $\neq$ SCA: fix β and find $N_a^\star$ instead

[1] Picek *et al.*, CHES 2019 [Pic+18]

Open issue with Machine Learning based SCA[1]

"How to evaluate the quality of a model during training?"

- Accuracy: probability to recover the secret key with one trace

Their observations

"Accuracy does not seem to be the right performance metric in SCA"

- *High* accuracy $\implies$ successful key recovery
- *Low* accuracy $\implies$ nothing, problem: often happens (*e.g.* highly noisy leakages)
- *Apparently, no other machine learning metric related to SCA metrics*

Accuracy: find β s.t. $N_a^\star = 1$ $\neq$ SCA: fix β and find $N_a^\star$ instead

Our claim: we can accurately estimate $N_a^\star$ with DL !

[1] Picek *et al.*, CHES 2019 [Pic+18]

FROM RESEARCH TO INDUSTRY

cea tech

Outline

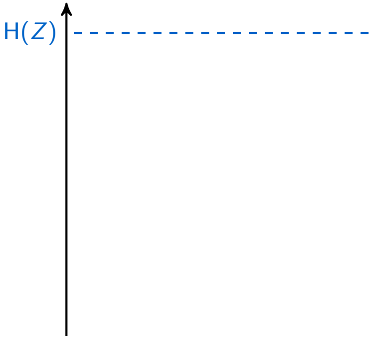
1. Context
2. SCA Optimization Problem versus Deep Learning Based SCA
3. NLL Minimization is PI Maximization
4. Simulation results
5. Experimental results

leti

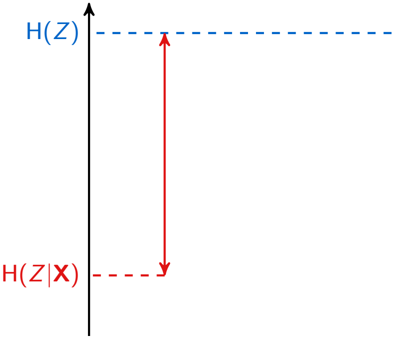
Bridging the gap between the loss function and the SCA metric

Bridging the gap between the loss function and the SCA metric

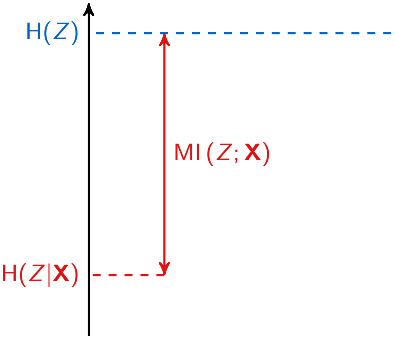
Bridging the gap between the loss function and the SCA metric



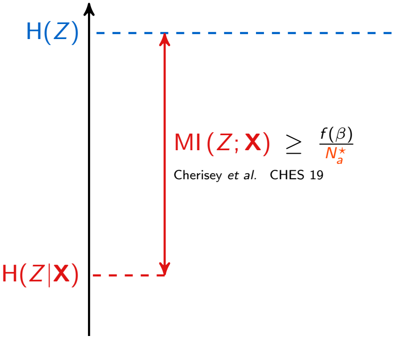
Bridging the gap between the loss function and the SCA metric



Bridging the gap between the loss function and the SCA metric

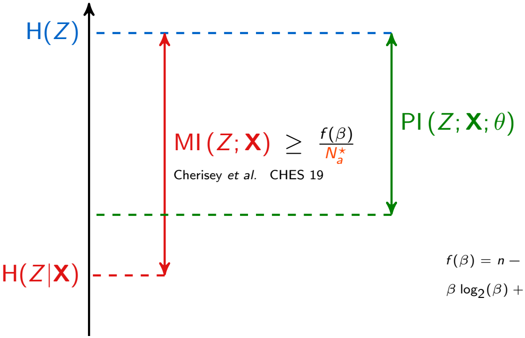


Bridging the gap between the loss function and the SCA metric



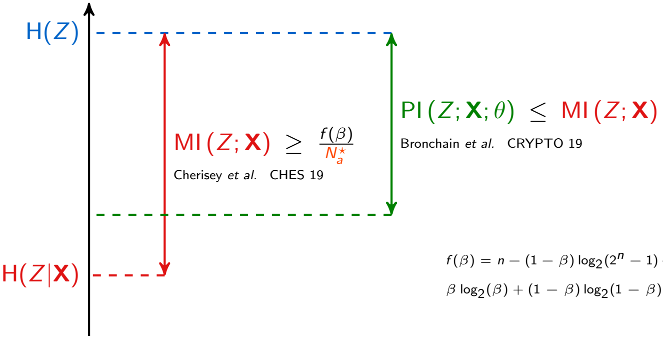
$$f(\beta) = n - (1-\beta)\log_2(2^n - 1) + \beta\log_2(\beta) + (1-\beta)\log_2(1-\beta)$$

Bridging the gap between the loss function and the SCA metric



$f(\beta) = n - (1-\beta)\log_2(2^n - 1) +$
$\beta \log_2(\beta) + (1-\beta)\log_2(1-\beta)$

Bridging the gap between the loss function and the SCA metric

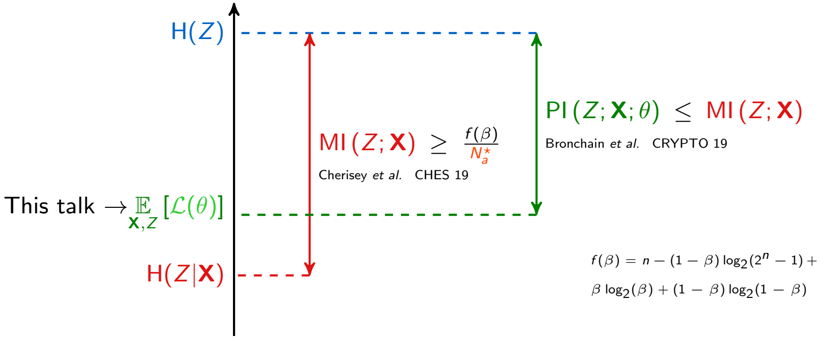


$$f(\beta) = n - (1-\beta)\log_2(2^n - 1) + \beta\log_2(\beta) + (1-\beta)\log_2(1-\beta)$$

Bridging the gap between the loss function and the SCA metric

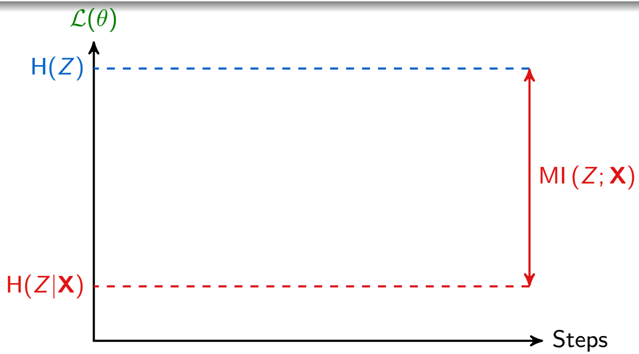
Training: minimization of the NLL *a.k.a.* Cross Entropy

$$\mathcal{L}(\theta) = \frac{1}{N_p}\sum_{i=1}^{N_p} -\log_2 F(\mathbf{x}_i, \theta)[z_i] = \mathrm{H}(Z) - \widehat{\mathrm{PI}_{N_p}}(Z; \mathbf{X}; \theta)$$



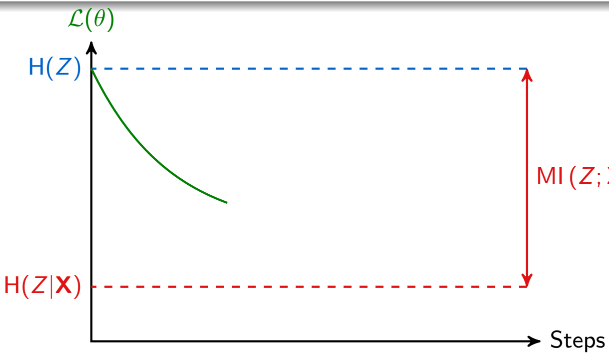
Main Result

Proposition



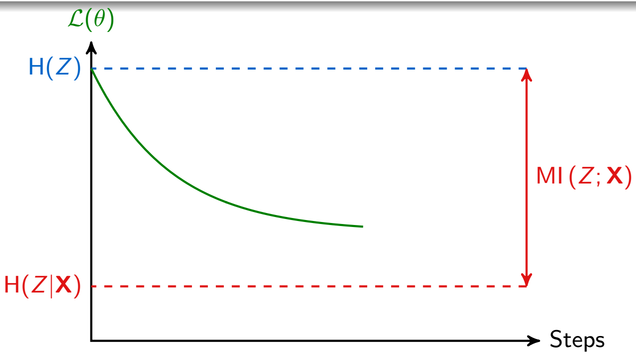
Main Result

Proposition



Main Result

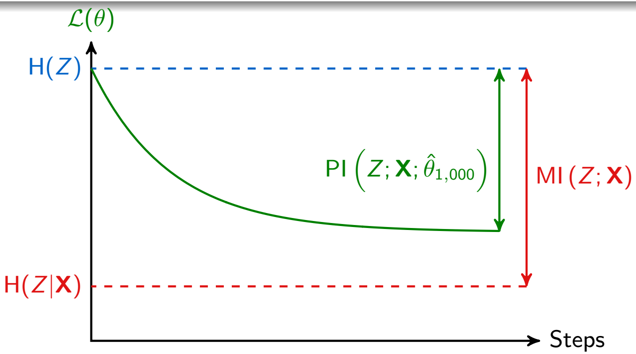
Proposition



Main Result

Proposition

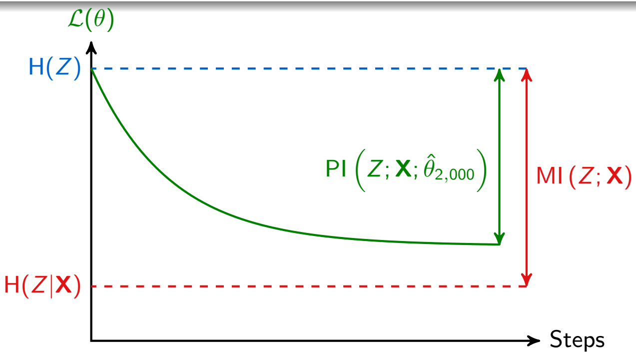
Let $\hat{\theta}_{N_p} = \text{argmin}_\theta \mathcal{L}(\theta) = \text{argmax}_\theta \widehat{\text{PI}_{N_p}}(Z; \mathbf{X}; \theta)$.



Main Result

Proposition

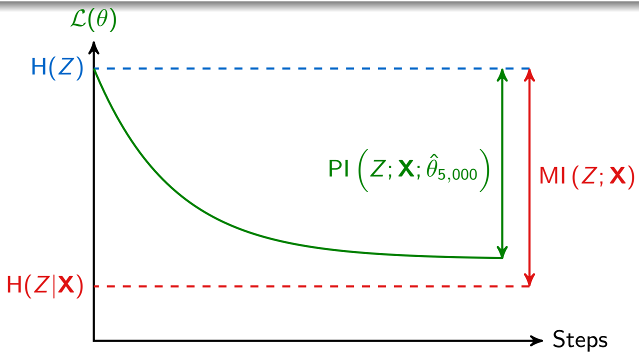
Let $\hat{\theta}_{N_p} = \mathrm{argmin}_\theta\, \mathcal{L}(\theta) = \mathrm{argmax}_\theta\, \widehat{\mathsf{PI}_{N_p}}\,(Z; \mathbf{X}; \theta)$.



Main Result

Proposition

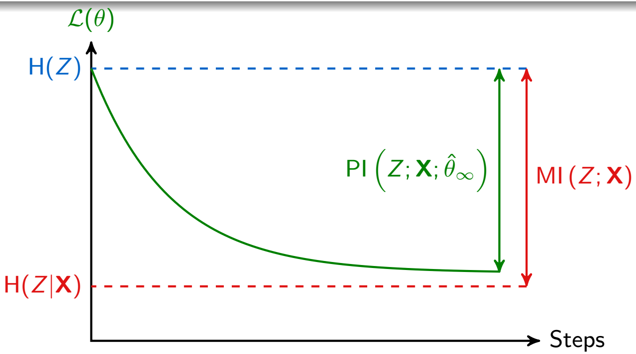
Let $\hat{\theta}_{N_p} = \text{argmin}_\theta\, \mathcal{L}(\theta) = \text{argmax}_\theta\, \widehat{\mathsf{PI}_{N_p}}\,(Z; \mathbf{X}; \theta)$.



Main Result

Proposition

Let $\hat{\theta}_{N_p} = \mathrm{argmin}_\theta\, \mathcal{L}(\theta) = \mathrm{argmax}_\theta\, \widehat{\mathrm{PI}_{N_p}}\,(Z; \mathbf{X}; \theta)$.

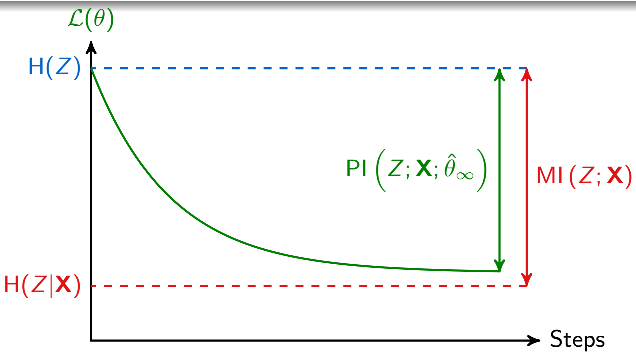


Main Result

Proposition

Let $\hat{\theta}_{N_p} = \mathrm{argmin}_\theta\, \mathcal{L}(\theta) = \mathrm{argmax}_\theta\, \widehat{\mathrm{PI}_{N_p}}\,(Z; \mathbf{X}; \theta)$. *Then:*

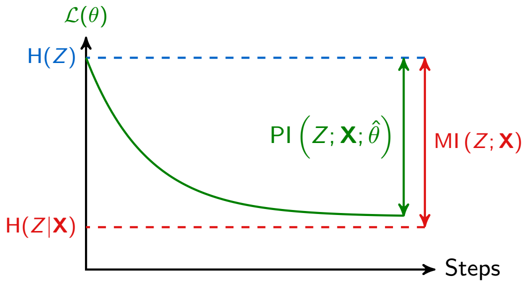
$$\mathrm{PI}\left(Z; \mathbf{X}; \hat{\theta}_{N_p}\right) \xrightarrow[N_p \to \infty]{\mathcal{P}} \sup_\theta \mathrm{PI}\,(Z; \mathbf{X}; \theta) \leq \mathrm{MI}\,(Z; \mathbf{X})$$



Tightness of the Lower Bound

To what extent the gap PI/MI is negligible?

Gap composed of three kinds of errors:

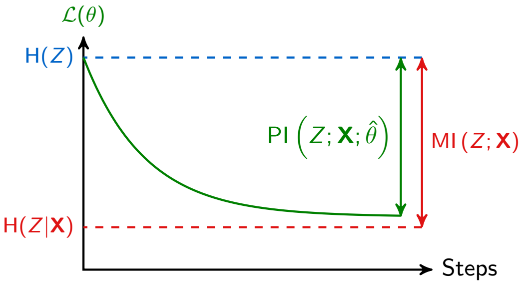


Tightness of the Lower Bound

To what extent the gap PI/MI is negligible?

Gap composed of three kinds of errors:

- Approximation error: $\sup_{\theta \in \Theta} \mathrm{PI}(Z; \mathbf{X}; \theta) - \mathrm{MI}(Z; \mathbf{X}) \leq 0$

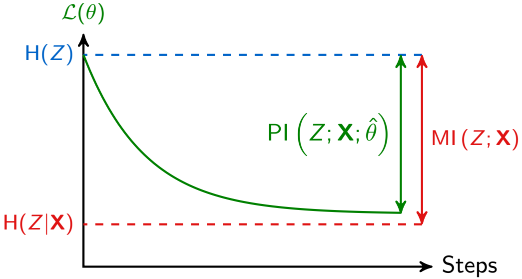


Tightness of the Lower Bound

To what extent the gap PI/MI is negligible?

Gap composed of three kinds of errors:

- Approximation error: $\sup_{\theta \in \Theta} \mathrm{PI}\left(Z; \mathbf{X}; \theta\right) - \mathrm{MI}\left(Z; \mathbf{X}\right) \leq 0$
- Estimation error: $N_p < \infty \implies \cancel{\sup_{\theta \in \Theta} \mathrm{PI}\left(Z; \mathbf{X}; \theta\right)} \rightarrow \widehat{\mathrm{PI}_{N_p}}\left(Z; \mathbf{X}; \hat{\theta}_{N_p}\right)$

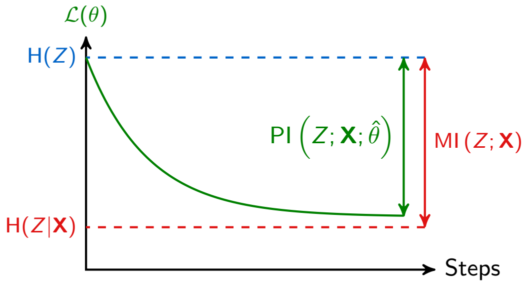


Tightness of the Lower Bound

To what extent the gap PI/MI is negligible?

Gap composed of three kinds of errors:

- Approximation error: $\sup_{\theta \in \Theta} \mathrm{PI}(Z; \mathbf{X}; \theta) - \mathrm{MI}(Z; \mathbf{X}) \leq 0$
- Estimation error: $N_p < \infty \implies \cancel{\sup_{\theta \in \Theta} \mathrm{PI}(Z; \mathbf{X}; \theta)} \to \widehat{\mathrm{PI}_{N_p}}\left(Z; \mathbf{X}; \hat{\theta}_{N_p}\right)$
- Optimization error: $\hat{\theta}_{N_p}$ unknown, θ_{SGD} instead, by SGD



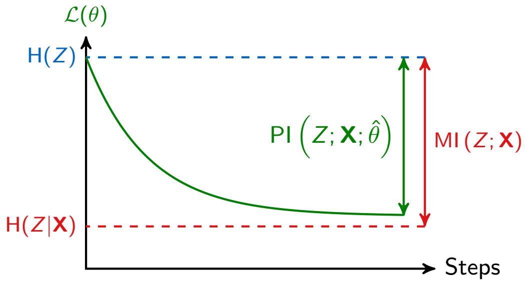
Tightness of the Lower Bound

To what extent the gap PI/MI is negligible?

Gap composed of three kinds of errors:

- Approximation error: $\sup_{\theta \in \Theta} \mathrm{PI}(Z; \mathbf{X}; \theta) - \mathrm{MI}(Z; \mathbf{X}) \leq 0$
- Estimation error: $N_p < \infty \implies \cancel{\sup_{\theta \in \Theta} \mathrm{PI}(Z; \mathbf{X}; \theta)} \rightarrow \widehat{\mathrm{PI}_{N_p}}\left(Z; \mathbf{X}; \hat{\theta}_{N_p}\right)$
- Optimization error: $\hat{\theta}_{N_p}$ unknown, θ_{SGD} instead, by SGD

$\implies$ Ideally each error must be discussed through simulations and experiments



Outline

Settings of the simulations

Leakage model

- Hamming weight with additive gaussian noise ($\sigma \in [0.01; 3.2]$)
- Draw an *Exhaustive* dataset: estimation error negligible

Settings of the simulations

Leakage model

- Hamming weight with additive gaussian noise ($\sigma \in [0.01; 3.2]$)
- Draw an *Exhaustive* dataset: estimation error negligible

PI/MI computation

- Computation of $\mathrm{MI}\left(\mathbf{X}; Z\right)$ with Monte-Carlo simulations

Settings of the simulations

Leakage model

- Hamming weight with additive gaussian noise ($\sigma \in [0.01; 3.2]$)
- Draw an *Exhaustive* dataset: estimation error negligible

PI/MI computation

- Computation of $\mathrm{MI}(\mathbf{X}; Z)$ with Monte-Carlo simulations
- Training of a one layer MLP with $1,000$ neurons to maximize $\mathrm{PI}(Z; \mathbf{X}; \theta) = n - \mathcal{L}(\theta)$, where $n = 4$ bits

Settings of the simulations

Leakage model

- Hamming weight with additive gaussian noise ($\sigma \in [0.01; 3.2]$)
- Draw an *Exhaustive* dataset: estimation error negligible

PI/MI computation

- Computation of $\mathsf{MI}(\mathbf{X}; Z)$ with Monte-Carlo simulations
- Training of a one layer MLP with 1,000 neurons to maximize $\mathsf{PI}(Z; \mathbf{X}; \theta) = n - \mathcal{L}(\theta)$, where $n = 4$ bits

Several case studies

- Higher-order masking: sensitive variable split into d independent parts
- Shuffling: independent operations (*e.g.* 16 SBoxes in SubBytes) randomly shuffled

Simulation results

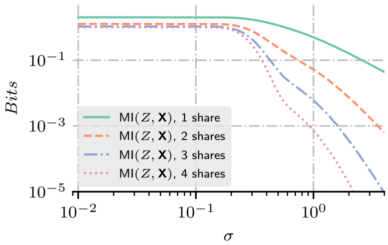


Figure: H-O masking, w.r.t. level of noise

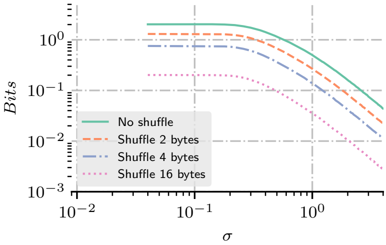


Figure: Shuffling, w.r.t. level of noise

Simulation results

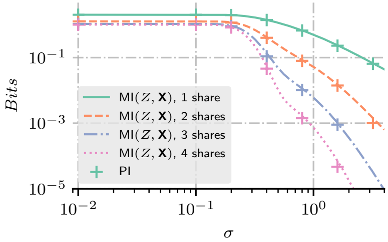


Figure: H-O masking, w.r.t. level of noise

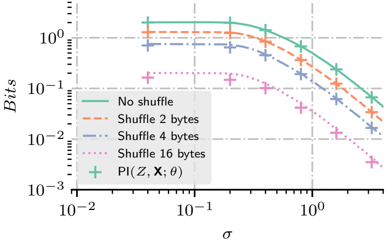


Figure: Shuffling, w.r.t. level of noise

What to interpret

- No matter the masking order, $\mathrm{PI}(Z; \mathbf{X}; \theta_{SGD}) \approx \mathrm{MI}(Z; \mathbf{X})$

Simulation results

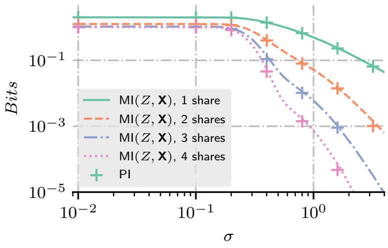


Figure: H-O masking, w.r.t. level of noise

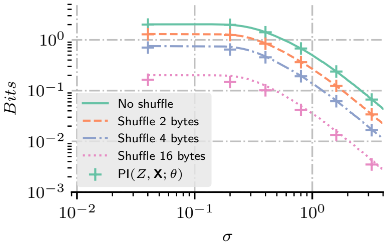


Figure: Shuffling, w.r.t. level of noise

What to interpret

- No matter the masking order, $\mathsf{PI}\left(Z; \mathbf{X}; \theta_{SGD}\right) \approx \mathsf{MI}\left(Z; \mathbf{X}\right)$
- For a simple MLP, the approximation error and the optimization error are negligible

Simulation results

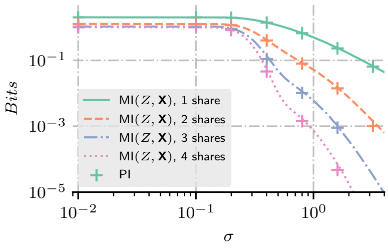


Figure: H-O masking, w.r.t. level of noise

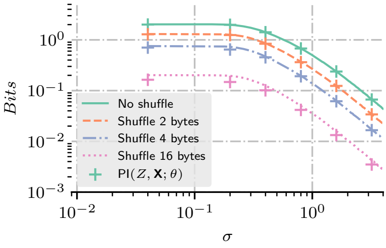


Figure: Shuffling, w.r.t. level of noise

What to interpret

- No matter the masking order, $\mathrm{PI}(Z; \mathbf{X}; \theta_{SGD}) \approx \mathrm{MI}(Z; \mathbf{X})$
- For a simple MLP, the approximation error and the optimization error are negligible
- Any more *complex* model should have a negligible approximation error too
- Empirical verifications: see appendix

Outline

1. Context

2. SCA Optimization Problem versus Deep Learning Based SCA

3. NLL Minimization is PI Maximization

4. Simulation results

5. Experimental results

Application on Public Datasets

- $N_a(\theta)$: number of traces obtained with key recovery.

Application on Public Datasets

- $N_a(\theta)$: number of traces obtained with key recovery.
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathrm{MI}(Z;\mathbf{X})}$ and $\mathrm{PI}\left(Z;\mathbf{X};\theta_{SGD}\right) \approx \mathrm{MI}\left(Z;\mathbf{X}\right)$

Application on Public Datasets

- $\cancel{N_a(\theta)}\ \frac{f(\beta)}{\mathrm{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n-\mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathrm{MI}(Z;\mathbf{X})}$ and $\mathrm{PI}(Z;\mathbf{X};\theta_{SGD}) \approx \mathrm{MI}(Z;\mathbf{X})$

Application on Public Datasets

- $\cancel{N_a(\theta)}$ $\frac{f(\beta)}{\mathrm{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n - \mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathrm{MI}(Z;\mathbf{X})}$ and $\mathrm{PI}(Z;\mathbf{X};\theta_{SGD}) \approx \mathrm{MI}(Z;\mathbf{X})$
- Tests on public datasets, using architectures proposed in recent papers [MDP19; Kim+19]

Application on Public Datasets

- $\cancel{N_a(\theta)}\ \frac{f(\beta)}{\mathrm{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n-\mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathrm{MI}(Z;\mathbf{X})}$ and $\mathrm{PI}\,(Z;\mathbf{X};\theta_{SGD}) \approx \mathrm{MI}\,(Z;\mathbf{X})$
- Tests on public datasets, using architectures proposed in recent papers [MDP19; Kim+19]
- Relative error ϵ computed at final epoch

Application on Public Datasets

- $\cancel{N_a(\theta)}$ $\frac{f(\beta)}{\mathrm{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n-\mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathrm{MI}(Z;\mathbf{X})}$ and $\mathrm{PI}\,(Z;\mathbf{X};\theta_{SGD}) \approx \mathrm{MI}\,(Z;\mathbf{X})$
- Tests on public datasets, using architectures proposed in recent papers [MDP19; Kim+19]
- Relative error ϵ computed at final epoch

Micro-controller protected with misalignment

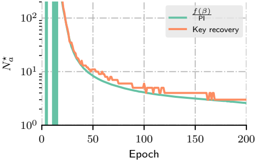


Figure: AES-RD: $\epsilon = 0.16$

Application on Public Datasets

- $\cancel{N_a(\theta)}$ $\frac{f(\beta)}{\mathsf{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n-\mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathsf{MI}(Z;\mathbf{X})}$ and $\mathsf{PI}\,(Z;\mathbf{X};\theta_{SGD}) \approx \mathsf{MI}\,(Z;\mathbf{X})$
- Tests on public datasets, using architectures proposed in recent papers [MDP19; Kim+19]
- Relative error ϵ computed at final epoch

Micro-controller protected with misalignment

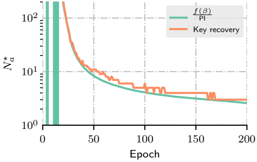


Figure: AES-RD: $\epsilon = 0.16$

Micro-controller protected with masking

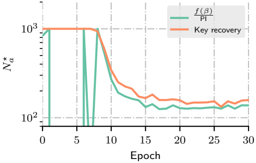


Figure: ASCAD: $\epsilon = 0.16$

Application on Public Datasets

- $\cancel{N_a(\theta)}$ $\frac{f(\beta)}{\mathsf{PI}(Z;\mathbf{X};\theta)} \approx \frac{f(\beta)}{n-\mathcal{L}(\theta)}$: number of traces obtained with key recovery?
- So far: $N_a^\star \geq \frac{f(\beta)}{\mathsf{MI}(Z;\mathbf{X})}$ and $\mathsf{PI}\left(Z;\mathbf{X};\theta_{SGD}\right) \approx \mathsf{MI}\left(Z;\mathbf{X}\right)$
- Tests on public datasets, using architectures proposed in recent papers [MDP19; Kim+19]
- Relative error ϵ computed at final epoch

Micro-controller protected with misalignment

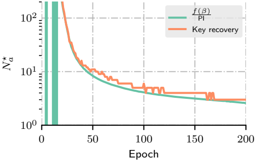


Figure: AES-RD: $\epsilon = 0.16$

Micro-controller protected with masking

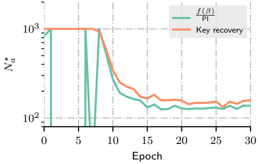


Figure: ASCAD: $\epsilon = 0.16$

Implementation on FPGA (no counter-measure)

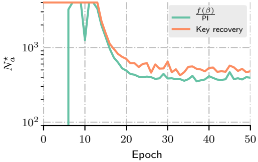


Figure: AES-HD: $\epsilon = 0.18$

Conclusion

Takeaway messages

Conclusion

Takeaway messages

1. Minimizing the NLL loss $\equiv$ maximizing the PI $\implies$ tight lower bound of the MI $\implies$ accurate estimation of $N_a^\star$

Conclusion

Takeaway messages

1. Minimizing the NLL loss $\equiv$ maximizing the PI $\implies$ tight lower bound of the MI $\implies$ accurate estimation of $N_a^\star$
2. NLL as a loss function is sound from an evaluator point of view

Conclusion

Takeaway messages

1. Minimizing the NLL loss $\equiv$ maximizing the PI $\implies$ tight lower bound of the MI $\implies$ accurate estimation of $N_a^\star$
2. NLL as a loss function is sound from an evaluator point of view
3. Enables to quantitatively measure the impact of counter-measures

Thank You!

Questions?

Looking for a postdoc candidate in machine-learning-based SCA? Hire me!

References I

[CDP17] Eleonora Cagli, Cécile Dumas, and Emmanuel Prouff. "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures - Profiling Attacks Without Pre-processing". In: *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*. Ed. by Wieland Fischer and Naofumi Homma. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 45–68. ISBN: 978-3-319-66786-7. DOI: 10.1007/978-3-319-66787-4_3. URL: https://doi.org/10.1007/978-3-319-66787-4_3.

[Kim+19] Jaehun Kim et al. "Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled Side-channel Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.3 (2019), pp. 148–179. DOI: 10.13154/tches.v2019.i3.148-179. URL: https://tches.iacr.org/index.php/TCHES/article/view/8292.

References II

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. “Breaking Cryptographic Implementations Using Deep Learning Techniques”. In: *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*. Ed. by Claude Carlet, M. Anwar Hasan, and Vishal Saraswat. Vol. 10076. Lecture Notes in Computer Science. Springer, 2016, pp. 3–26. ISBN: 978-3-319-49444-9. DOI: 10.1007/978-3-319-49445-6_1. URL: https://doi.org/10.1007/978-3-319-49445-6_1.

[MDP19] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. “Gradient Visualization for General Characterization in Profiling Attacks”. In: *Constructive Side-Channel Analysis and Secure Design - 10th International Workshop, COSADE 2019, Darmstadt, Germany, April 3-5, 2019, Proceedings*. Ed. by Ilia Polian and Marc Stöttinger. Vol. 11421. Lecture Notes in Computer Science. Springer, 2019, pp. 145–167. ISBN: 978-3-030-16349-5. DOI: 10.1007/978-3-030-16350-1_9. URL: https://doi.org/10.1007/978-3-030-16350-1_9.

References III

[Pic+18] Stjepan Picek et al. "The Curse of Class Imbalance and Conflicting Metrics with Machine Learning for Side-channel Evaluations". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.1 (2018), pp. 209–237. DOI: 10.13154/tches.v2019.i1.209-237. URL: https://tches.iacr.org/index.php/TCHES/article/view/7339.

[Vey+12] Nicolas Veyrat-Charvillon et al. "Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note". In: *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. Lecture Notes in Computer Science. Springer, 2012, pp. 740–757. ISBN: 978-3-642-34960-7. DOI: 10.1007/978-3-642-34961-4_44. URL: https://doi.org/10.1007/978-3-642-34961-4_44.

Our home dataset

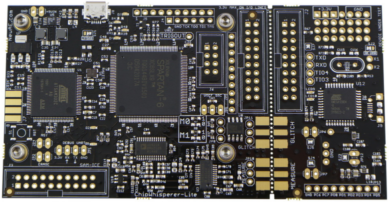


Figure: ChipWhisperer-Lite board

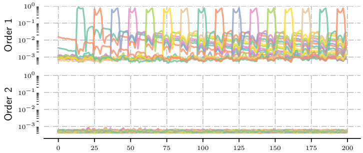


Figure: SNR at orders $d = 1, 2$

Algorithm 1 loadData

```
LD r0, X       ▷ Loads the first byte in r0
CLR r0         ▷ Clears the register
ST X, r0       ▷ Stores 0 in the plaintext array
LD r0, X       ▷ Do it again to clear the bus
CLR r0
ST X, r0
LD r0, X       ▷ One more time to be sure
CLR r0
ST X+, r0
```

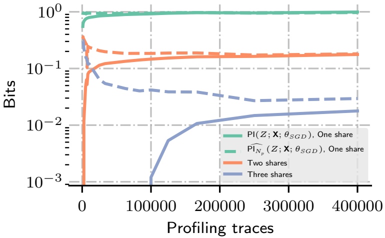
Loads sequentially an array of 16 bytes into one register and clears it $\implies$ no joint leakage at order $d \geq 2$.
500, 000 traces acquired.
We only work on $n = 4$ bits, $|\mathcal{Z}| = 2^n = 16$.

Experiment on ChipWhisperer-Lite: masking

- Emulation of order d leakages: $Z = \bigoplus_{i \in [\![0,d]\!]} plain[i]$ for $d \in \{0, 1, 2\}$
- Extraction of PoIs according to SNR.
- Learning curve: $\mathrm{PI}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ and $\widehat{\mathrm{PI}_{N_p}}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ w.r.t. N_p $\Longrightarrow$ measures the estimation error.

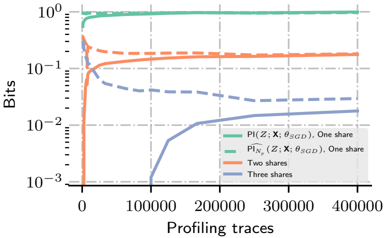
Experiment on ChipWhisperer-Lite: masking

- Emulation of order d leakages: $Z = \bigoplus_{i \in [\![0,d]\!]} plain[i]$ for $d \in \{0,1,2\}$
- Extraction of PoIs according to SNR.
- Learning curve: $\mathrm{PI}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ and $\widehat{\mathrm{PI}_{N_p}}\left(Z; \mathbf{X}; \theta_{SGD}\right)$ w.r.t. N_p $\Longrightarrow$ measures the estimation error.



Experiment on ChipWhisperer-Lite: masking

- Emulation of order d leakages: $Z = \bigoplus_{i \in [\![0,d]\!]} plain[i]$ for $d \in \{0, 1, 2\}$
- Extraction of PoIs according to SNR.
- Learning curve: $\mathrm{PI}(Z; \mathbf{X}; \theta_{SGD})$ and $\widehat{\mathrm{PI}_{N_p}}(Z; \mathbf{X}; \theta_{SGD})$ w.r.t. N_p $\Longrightarrow$ measures the estimation error.

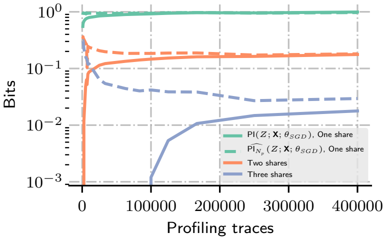


What to interpret

- $\approx$ one decade lost for each new masking order $\Longrightarrow$ masking remains sound

Experiment on ChipWhisperer-Lite: masking

- Emulation of order d leakages: $Z = \bigoplus_{i \in [\![0,d]\!]} plain[i]$ for $d \in \{0, 1, 2\}$
- Extraction of PoIs according to SNR.
- Learning curve: $\mathrm{PI}(Z; \mathbf{X}; \theta_{SGD})$ and $\widehat{\mathrm{PI}_{N_p}}(Z; \mathbf{X}; \theta_{SGD})$ w.r.t. N_p $\implies$ measures the estimation error.

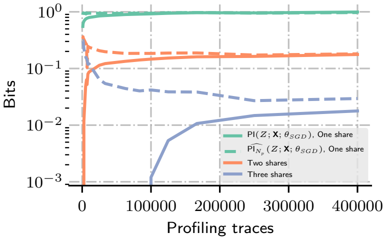


What to interpret

- $\approx$ one decade lost for each new masking order $\implies$ masking remains sound
- Masking has an effect on the estimation error

Experiment on ChipWhisperer-Lite: masking

- Emulation of order d leakages: $Z = \bigoplus_{i \in [\![0,d]\!]} plain[i]$ for $d \in \{0, 1, 2\}$
- Extraction of PoIs according to SNR.
- Learning curve: $\mathrm{PI}(Z; \mathbf{X}; \theta_{SGD})$ and $\widehat{\mathrm{PI}_{N_p}}(Z; \mathbf{X}; \theta_{SGD})$ w.r.t. N_p $\implies$ measures the estimation error.



What to interpret

- $\approx$ one decade lost for each new masking order $\implies$ masking remains sound
- Masking has an effect on the estimation error
- For $d = 3$, $N_p < 100,000$, no information !

Experiment 5: shuffling

- Emulation of order c shuffling: $Z = plain[i]$ where i is randomly drawn from a subset of c indices
- Complete trace: $D = 250$

Experiment 5: shuffling

- Emulation of order c shuffling: $Z = plain[i]$ where i is randomly drawn from a subset of c indices
- Complete trace: $D = 250$

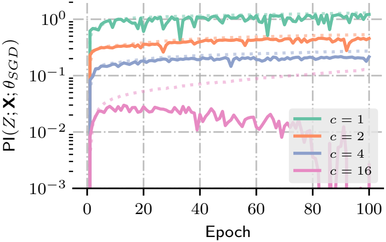


Figure: Exp. 5, shuffling

Experiment 5: shuffling

- Emulation of order c shuffling: $Z = plain[i]$ where i is randomly drawn from a subset of c indices
- Complete trace: $D = 250$

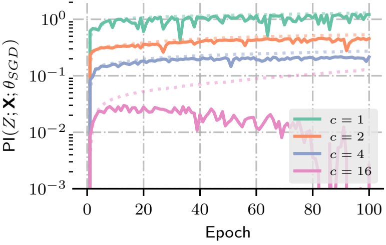


Figure: Exp. 5, shuffling

What to interpret

- Linear decrease of PI, as expected [Vey+12]

Experiment 5: shuffling

- Emulation of order c shuffling: $Z = plain[i]$ where i is randomly drawn from a subset of c indices
- Complete trace: $D = 250$

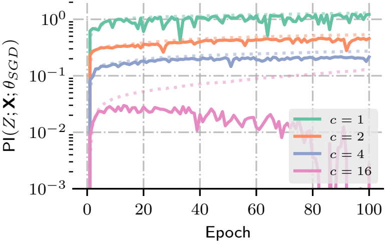


Figure: Exp. 5, shuffling

What to interpret

- Linear decrease of PI, as expected [Vey+12]
- Clearly over-fitting: the estimation error non-negligible