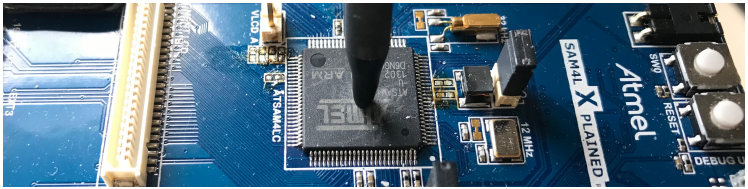


Deep-Learning for Side-Channel Analysis

Where are We ?

Loïc Masure

Saint-Étienne, November 18th

European Research Council
Established by the European Commission

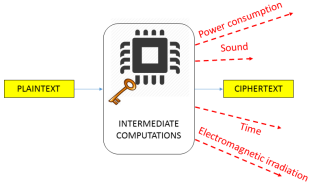
UCLouvain

Table of Contents

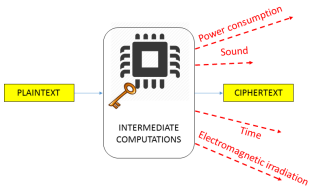
Content

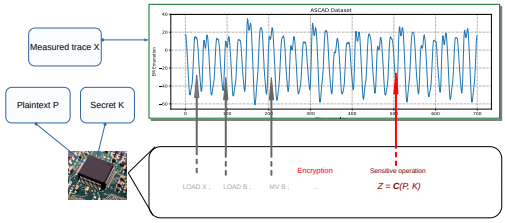
What is Side-Chanel Analysis (SCA)?

What is Side-Chanel Analysis (SCA)?

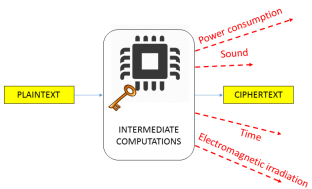


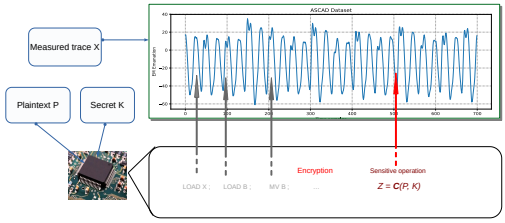
What is Side-Chanel Analysis (SCA)?





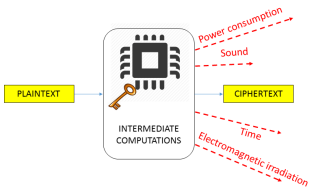
What is Side-Chanel Analysis (SCA)?

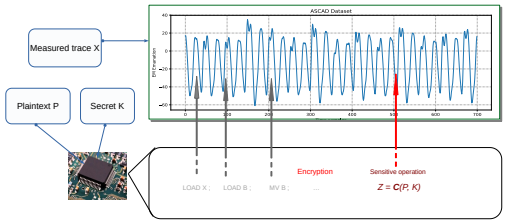




Leakages on intermediate computations allows **divide & conquer** strategy.

What is Side-Chanel Analysis (SCA)?



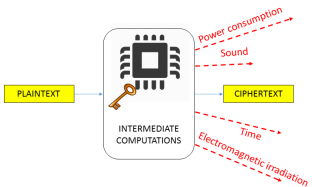


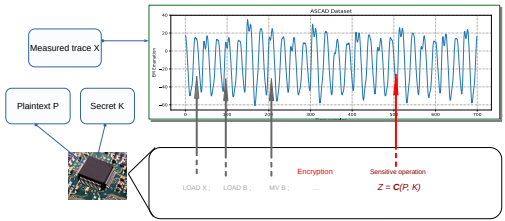
Leakages on intermediate computations allows **divide & conquer** strategy.
Example for AES with 8-bit architecture, with key length $N = 128$:

Classical cryptanalysis (brute force): $2^{128} \gg$ #atoms in the Sun

SCA: 16×256

What is Side-Chanel Analysis (SCA)?



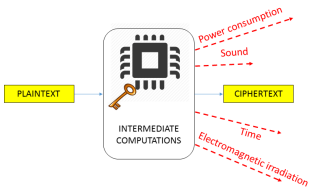


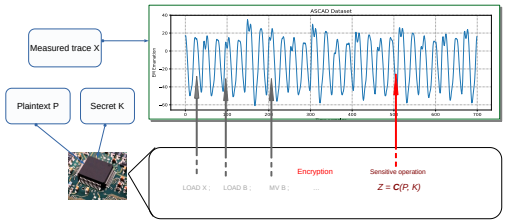
Leakages on intermediate computations allows **divide & conquer** strategy.
Example for algorithm designed for n-bit architecture, with key length N:

Classical cryptanalysis (brute force): $2^{128} \gg$ #atoms in the Sun

SCA: 16×256

What is Side-Chanel Analysis (SCA)?



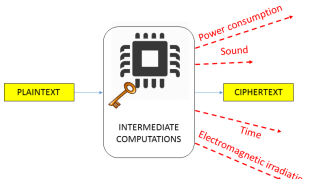


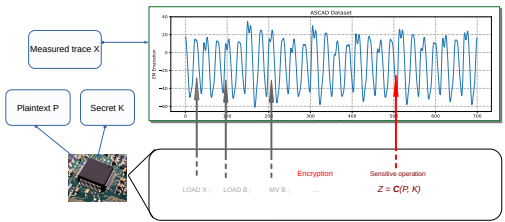
Leakages on intermediate computations allows **divide & conquer** strategy.
Example for algorithm designed for n-bit architecture, with key length N:

Classical cryptanalysis (brute force): 2^N

SCA: 16×256

What is Side-Chanel Analysis (SCA)?



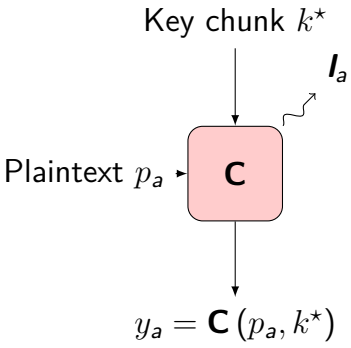


Leakages on intermediate computations allows **divide & conquer** strategy.
Example for algorithm designed for n-bit architecture, with key length N:

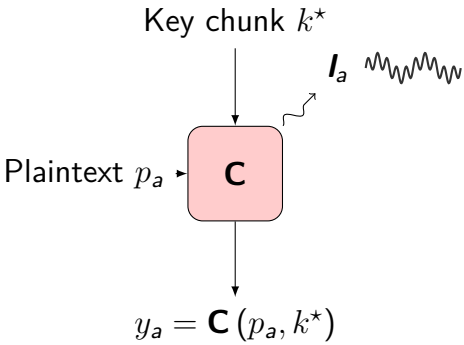
Classical cryptanalysis (brute force): 2^N

SCA: $\frac{N}{n} \times 2^n$

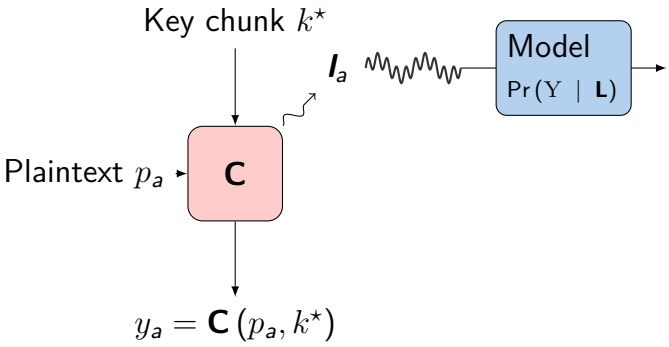
How does an SCA work



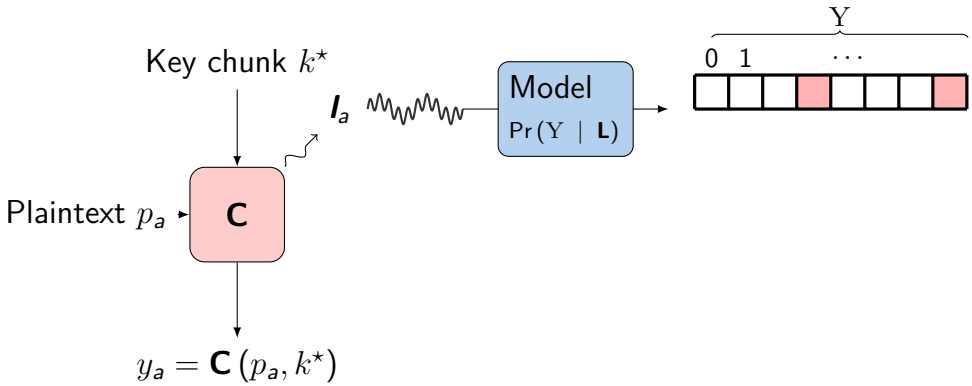
How does an SCA work



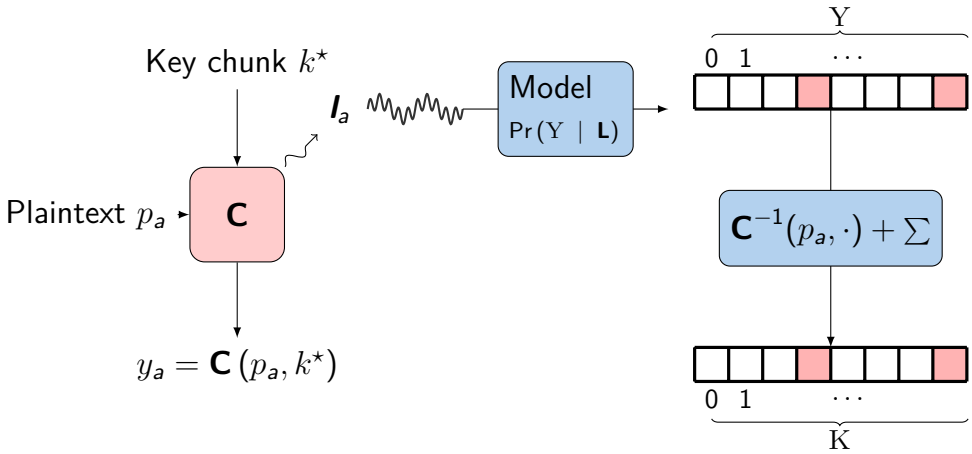
How does an SCA work



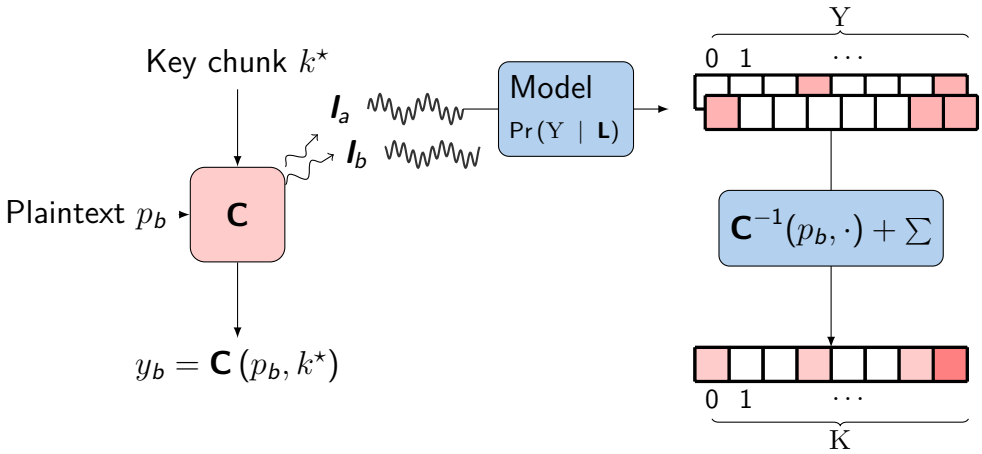
How does an SCA work



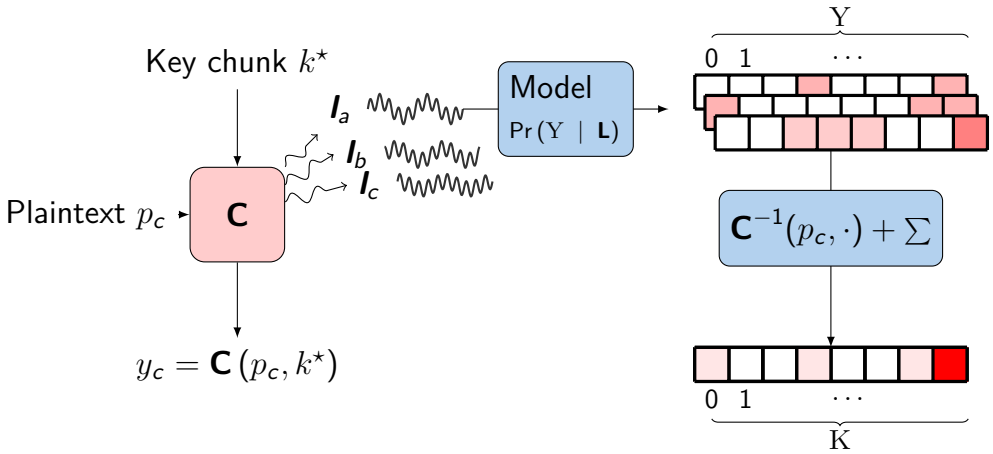
How does an SCA work



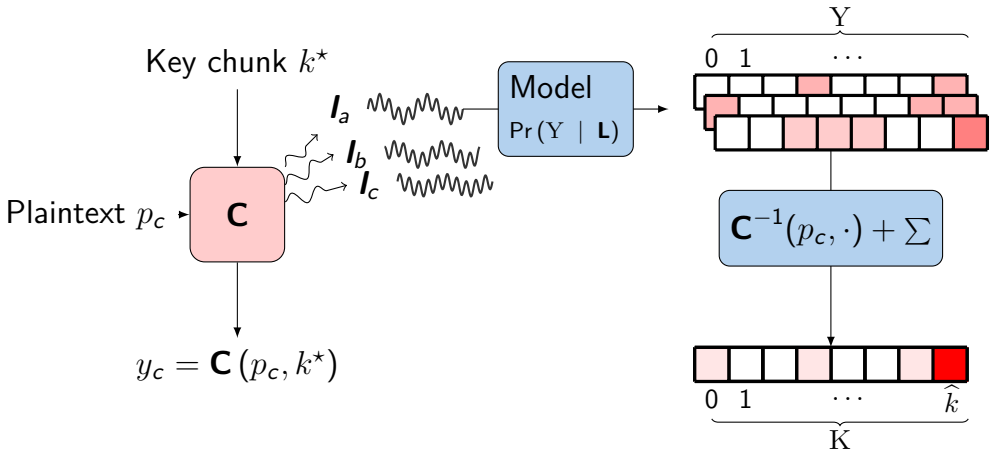
How does an SCA work



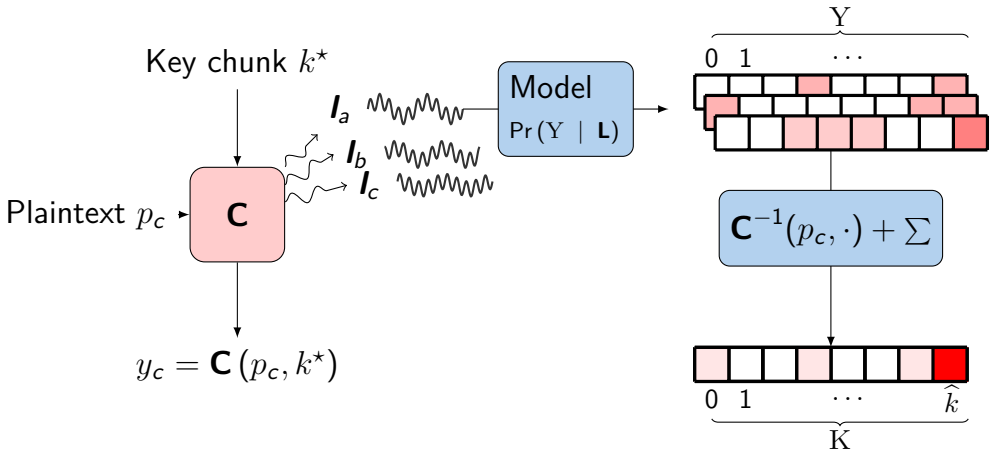
How does an SCA work



How does an SCA work



How does an SCA work



Successful attack iff $\widehat{k} = k^\star$

From scores to Metrics

If, the adversary gets:

From scores to Metrics

If, the adversary gets:

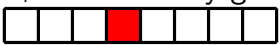
Sensitive computation unpredictable
SCA not more powerful than cryptanalysis
Device fully secure

From scores to Metrics

If, the adversary gets:

Sensitive computation unpredictable
SCA not more powerful than cryptanalysis
Device fully secure

If, the adversary gets:

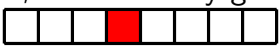


From scores to Metrics

If, the adversary gets:

Sensitive computation unpredictable
SCA not more powerful than cryptanalysis
Device fully secure

If, the adversary gets:



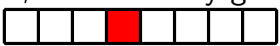
Exact prediction of the sensitive computation
Success rate of 100% with *one* trace
Device not secure at all

From scores to Metrics

If, the adversary gets:

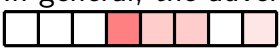
Sensitive computation unpredictable
SCA not more powerful than cryptanalysis
Device fully secure

If, the adversary gets:



Exact prediction of the sensitive computation
Success rate of 100% with *one* trace
Device not secure at all

In general, the adversary gets:

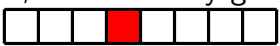


From scores to Metrics

If, the adversary gets:

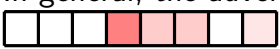
Sensitive computation unpredictable
SCA not more powerful than cryptanalysis
Device fully secure

If, the adversary gets:



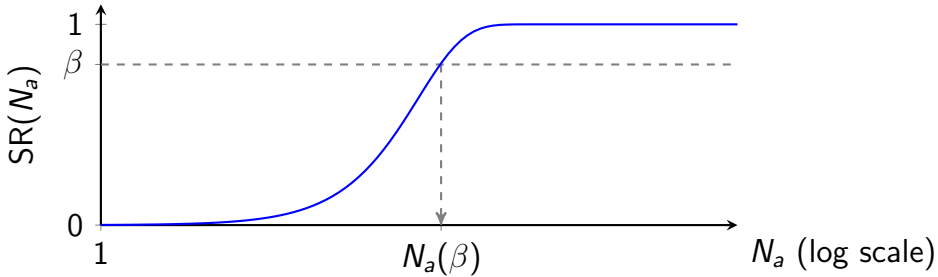
Exact prediction of the sensitive computation
Success rate of 100% with *one* trace
Device not secure at all

In general, the adversary gets:



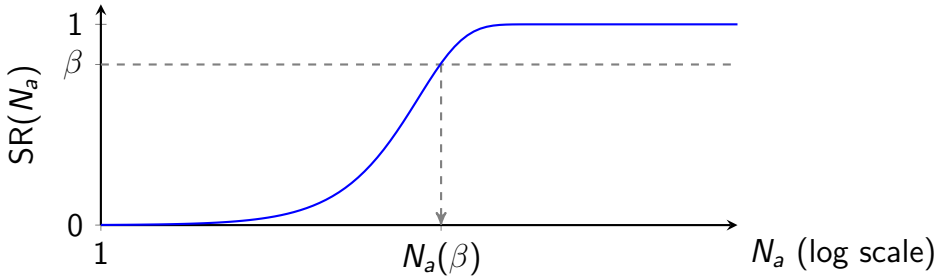
How does this translate into SCA security metrics ?

Concrete SCA Metrics: the Success Rate (SR)



SR: probability to succeed the attack within N_a queries to the target

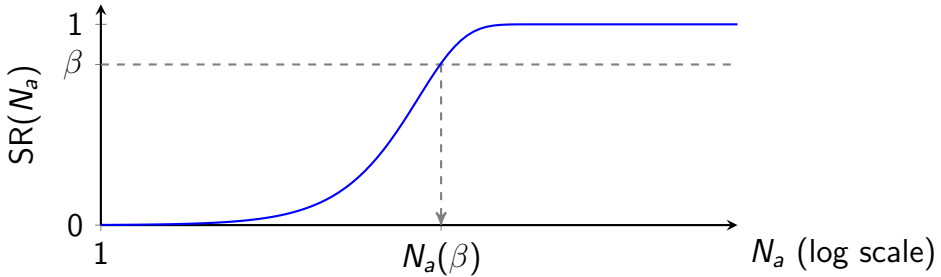
Concrete SCA Metrics: the Success Rate (SR)



SR: probability to succeed the attack within N_a queries to the target

Secured device with prob. $\geq 1 - \beta$, $\implies$ refresh secret every $N_a(\beta)$ use ✓

Concrete SCA Metrics: the Success Rate (SR)



SR: probability to succeed the attack within N_a queries to the target

Secured device with prob. $\geq 1-\beta$, $\implies$ refresh secret every $N_a(\beta)$ use ✓

Naive est. of $N_a(\beta)$ is expensive: complexity depends on $N_a(\beta)$ itself ✗

Circumventing the Drawbacks of the Success Rate (SR)

Can we find surrogate metrics characterizing $N_a(\beta)$?

[1] Mangard, Oswald, and Popp, *Power analysis attacks - revealing the secrets of smart cards*

[2] Chérisey et al., "Best Information is Most Successful: Mutual Information and Success Rate in Side-Channel Analysis"

Circumventing the Drawbacks of the Success Rate (SR)

Can we find surrogate metrics characterizing $N_a(\beta)$?

CPA [1]

Using correlation coeff.

$$N_a(\beta) \approx \frac{f(\beta)}{\rho^2}$$

Easy to estimate ρ ✓
Only for univariate, linear ✗

[1] Mangard, Oswald, and Popp, *Power analysis attacks - revealing the secrets of smart cards*
[2] Chérisey et al., "Best Information is Most Successful: Mutual Information and Success Rate in Side-Channel Analysis"

Circumventing the Drawbacks of the Success Rate (SR)

Can we find surrogate metrics characterizing $N_a(\beta)$?

CPA [1]

Using correlation coeff.

$$N_a(\beta) \approx \frac{f(\beta)}{\rho^2}$$

Easy to estimate ρ ✓
Only for univariate, linear ✗

General case [2]

Using the Mutual Information (MI),

$$N_a(\beta) \geq \frac{f(\beta)}{\mathrm{MI}(\mathrm{Y}; \mathbf{L})}$$

MI generalizes ρ ✓
MI hard to estimate ✗

[1] Mangard, Oswald, and Popp, *Power analysis attacks - revealing the secrets of smart cards*

[2] Chérisey et al., "Best Information is Most Successful: Mutual Information and Success Rate in Side-Channel Analysis"

Why Estimating MI is hard ?

No unbiased estimator [3] ✗
Curse of dim. if bias > 0 ✗
The lower the MI, the harder ✗
$\implies$ need for good estimators with bias < 0

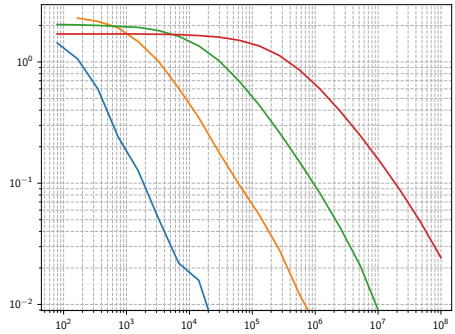
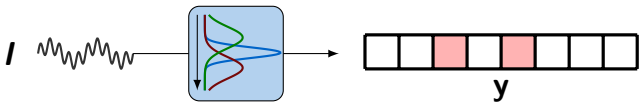


Figure: Bias (y-axis) vs. #traces (x-axis) for $D = 1, 2, 3, 4$. Taken from[4].

[3] Paninski, "Estimation of Entropy and Mutual Information"
[4] Masure et al., *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*

Content

Gaussian Templates (GTs)[6]



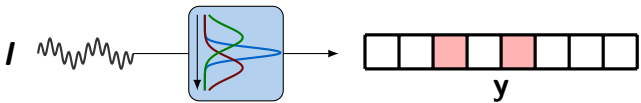
Gaussian Templates (GTs)

Generative model built from Gaussian generative laws

$$\Pr(\mathrm{Y} = s \mid \mathbf{L} = \boldsymbol{l}) \approx \frac{\mathcal{N}(\boldsymbol{l}, \mu_s, \Sigma_s)}{\sum_{s'} \mathcal{N}(\boldsymbol{l}, \mu_{s'}, \Sigma_{s'})} \tag{1}$$

[5] Masure et al., *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*.
[6] Chari, Rao, and Rohatgi, "Template Attacks", Ches 2002 (2022 Test of Time Award)

Gaussian Templates (GTs)[6]



Gaussian Templates (GTs)

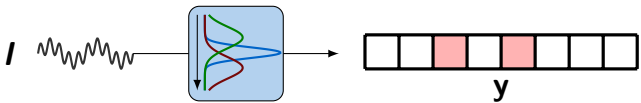
Generative model built from Gaussian generative laws

$$\Pr(\mathrm{Y} = s \mid \mathbf{L} = \boldsymbol{l}) \approx \frac{\mathcal{N}(\boldsymbol{l}, \mu_s, \Sigma_s)}{\sum_{s'} \mathcal{N}(\boldsymbol{l}, \mu_{s'}, \Sigma_{s'})} \quad (1)$$

Does not scale well to *multivariate* leakage[5] ✗

[5] Masure et al., *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*.
[6] Chari, Rao, and Rohatgi, "Template Attacks", Ches 2002 (2022 Test of Time Award)

Gaussian Templates (GTs)[6]



Gaussian Templates (GTs)

Generative model built from Gaussian generative laws

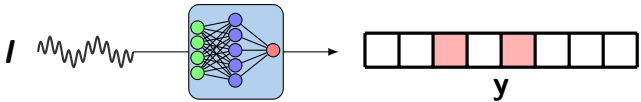
$$\Pr(\mathrm{Y} = s \mid \mathbf{L} = \boldsymbol{l}) \approx \frac{\mathcal{N}(\boldsymbol{l}, \mu_s, \Sigma_s)}{\sum_{s'} \mathcal{N}(\boldsymbol{l}, \mu_{s'}, \Sigma_{s'})} \tag{1}$$

Does not scale well to *multivariate* leakage[5] ✗

Relies on Gaussian hypothesis ✓

[5] Masure et al., *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*.
[6] Chari, Rao, and Rohatgi, "Template Attacks", Ches 2002 (2022 Test of Time Award)

Deep Learning (DL) for SCA



Deep Neural Network (DNN)

More general model

$$F : \left| \begin{array}{ccc} \mathcal{L} & \longrightarrow & \mathcal{P}(\mathcal{Y}) \\ \boldsymbol{l} & \longmapsto & \mathbf{y} = F(\boldsymbol{l}) \approx \Pr\left(\mathrm{Y} \mid \mathbf{L} = \boldsymbol{l}\right) \end{array} \right. \tag{2}$$

$F(\boldsymbol{l})$: output of a Directed Acyclic Graph (DAG) of computation:

Each node: elementary function $f_i(\cdot, \theta_i)$

θ_i: *parameters* fully describing f_i

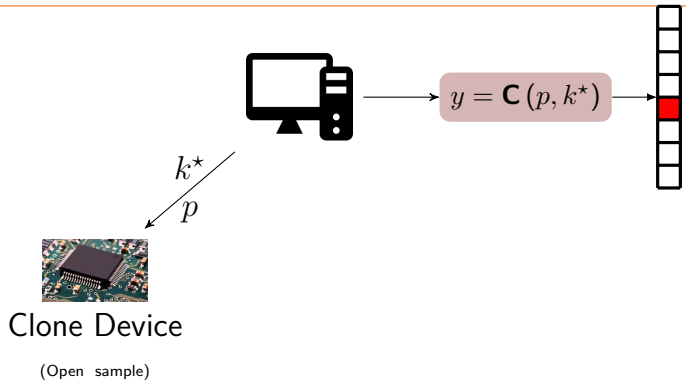
Shape of the DAG, nature of the classes of functions: *architecture* of the DNN.

Training a DNN for Profiled SCA

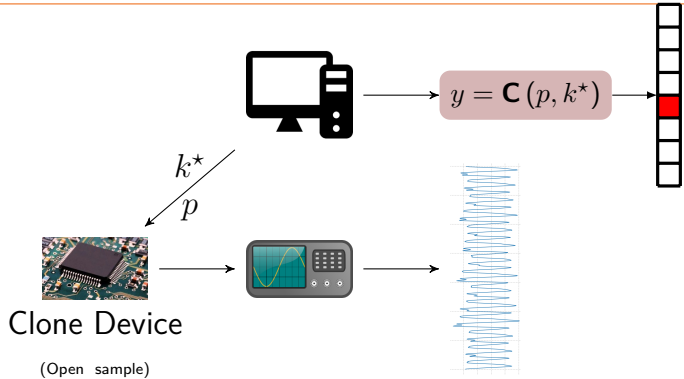
Clone Device

(Open sample)

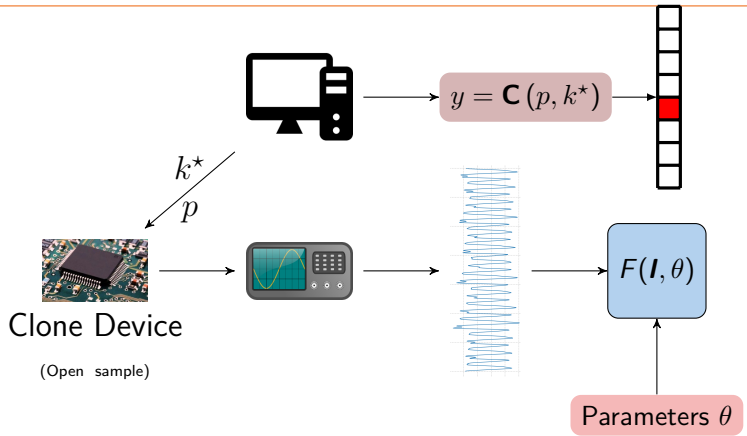
Training a DNN for Profiled SCA



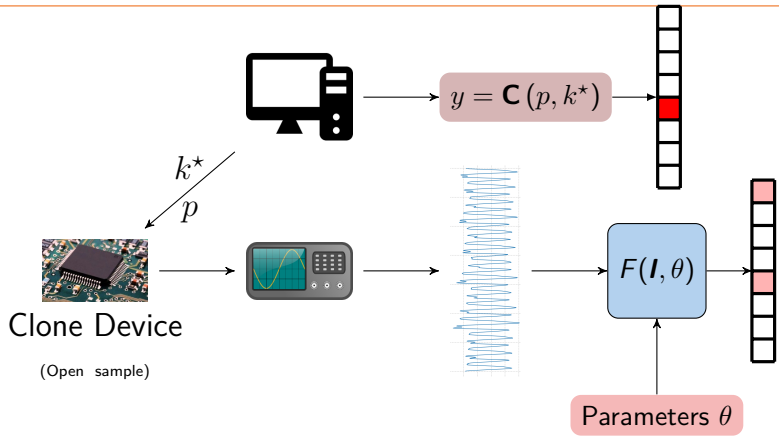
Training a DNN for Profiled SCA



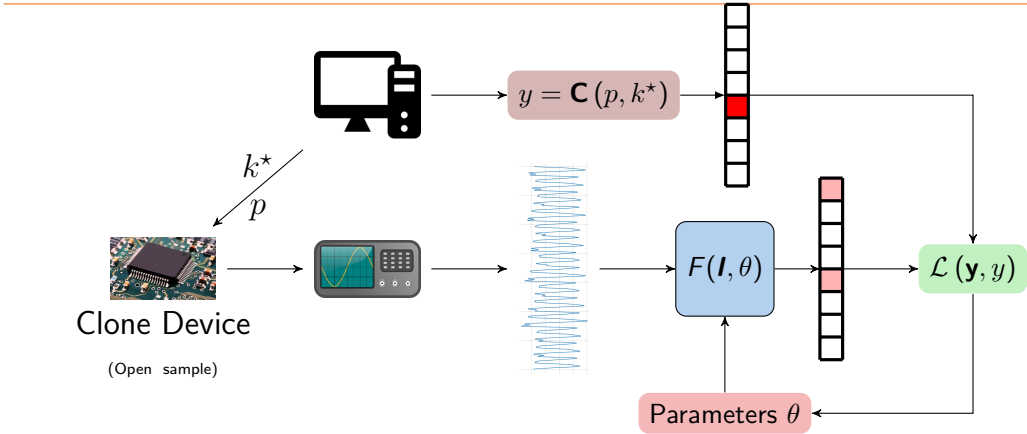
Training a DNN for Profiled SCA



Training a DNN for Profiled SCA

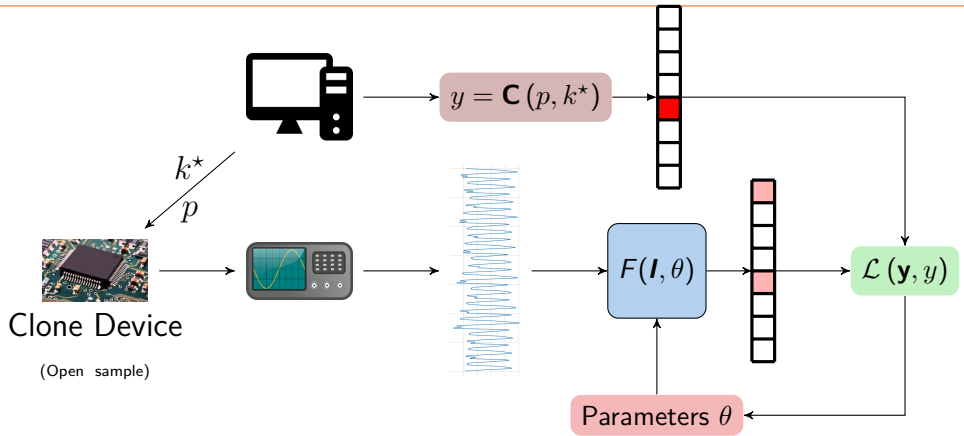


Training a DNN for Profiled SCA



$\mathcal{L}()$: loss function to minimize

Training a DNN for Profiled SCA

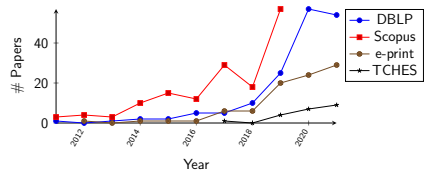


$\mathcal{L}()$: loss function to minimize

Use of *gradient descent* algorithm for minimization

The Deep Learning (DL) hype in SCA

- ➔ **Space 2016:** DL breaks *masking*[7]
- ➔ **Ches 2017:** CNNs efficiently tackles *misalignment*[8]
- ➔ **Ches 2019:** non-profiled attacks[9]
- ➔ *De facto* standard for evaluations
- ➔ Dedicated sessions in conferences



[7] Maghrebi, Portigliatti, and Prouff, "Breaking Cryptographic Implementations Using Deep Learning Techniques"

[8] Cagli, Dumas, and Prouff, "Convolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures - Profiling Attacks Without Pre-processing"

[9] Timon, "Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis"

Predicting the Online Attack Complexity

Back to the link between MI and $N_a(\beta)$...

[10] Masure, Dumas, and Prouff, "Gradient Visualization for General Characterization in Profiling Attacks"; Kim et al., "Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled

Predicting the Online Attack Complexity

Back to the link between MI and $N_a(\beta)$...

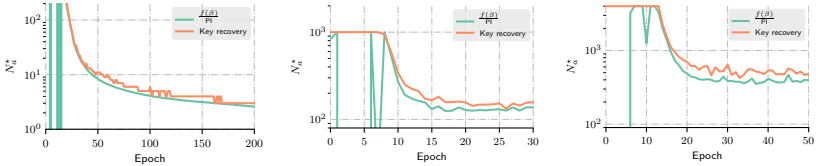
Tests on public datasets, using SOTA architectures[10]

[10] Masure, Dumas, and Prouff, "Gradient Visualization for General Characterization in Profiling Attacks"; Kim et al., "Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled

Predicting the Online Attack Complexity

Back to the link between MI and $N_a(\beta)$...

Tests on public datasets, using SOTA architectures[10]



MCU, random delay: $\epsilon = 16\%$ MCU, masking: $\epsilon = 16\%$ FPGA (unprotected): $\epsilon = 18\%$

T-table 32-bit (polymorphic): $\epsilon < 1\%$ AES 8-bit (polymorphic): $\epsilon = 15\%$

[10] Masure, Dumas, and Prouff, "Gradient Visualization for General Characterization in Profiling Attacks"; Kim et al., "Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled

Content

Towards SCA dedicated Architectures

DL-SCA mostly inspired by computer vision
Usually uses **many** layers, with **small** filters [11]

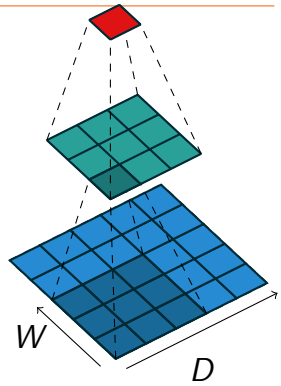


Figure: A 2D receptive field.[12]

[11] Simonyan and Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition"
[12] Inspired by Dumoulin and Visin, *A guide to convolution arithmetic for deep learning*

Convolution Arithmetics with 2D Data

A 2D receptive field of size $D \times D$, captured by two different settings.
\# parameters: $\frac{DW^2}{W-1} \approx DW$
minimized by setting W small

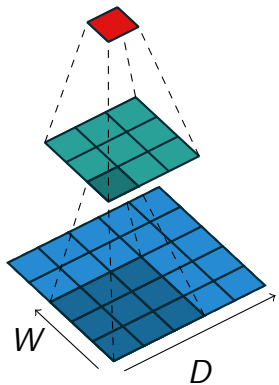


Figure: 2 layers, $W = 3$.

Convolution Arithmetics with 2D Data

A 2D receptive field of size $D \times D$, captured by two different settings.
\# parameters: $\frac{DW^2}{W-1} \approx DW$
minimized by setting W small

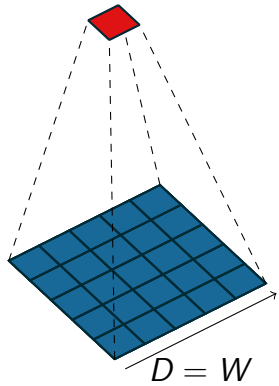


Figure: 1 layer, $W = 5$.

Convolution Arithmetics with 1D Data

A 1D receptive field of size $D = 5$, captured either by one or two convolution layers.
parameters: $D\frac{W}{W-1} \approx D$, independent of W
We don't necessarily need to stack many layers in SCA ![13] [14]

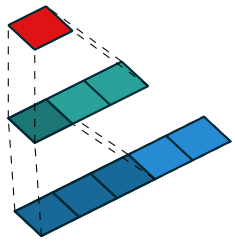


Figure: 2 layers, $W = 3$.

[13] Zaid et al., "Methodology for Efficient CNN Architectures in Profiling Attacks", Ches 2020
[14] Masure et al., "Deep Learning Side-Channel Analysis on Large-Scale Traces - A Case Study on a Polymorphic AES", Esorics 2020

Convolution Arithmetics with 1D Data

A 1D receptive field of size $D = 5$, captured either by one or two convolution layers.
parameters: $D\frac{W}{W-1} \approx D$, independent of W
We don't necessarily need to stack many layers in SCA ![13] [14]

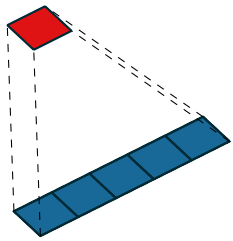
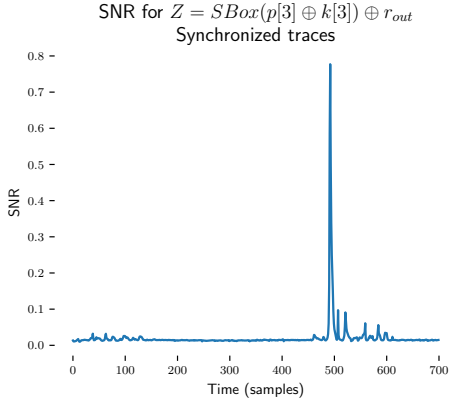


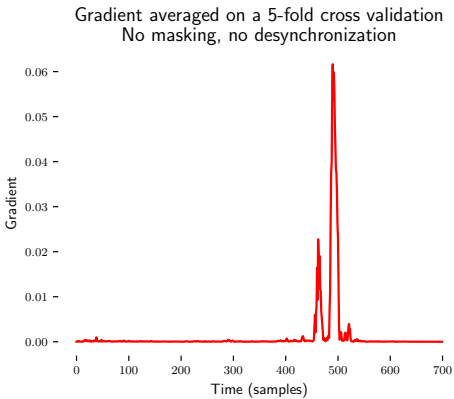
Figure: 1 layer, $W = 5$.

[13] Zaid et al., "Methodology for Efficient CNN Architectures in Profiling Attacks", Ches 2020
[14] Masure et al., "Deep Learning Side-Channel Analysis on Large-Scale Traces - A Case Study on a Polymorphic AES", Esorics 2020

Sensitivity Analysis for P.o.Is Selection[15]



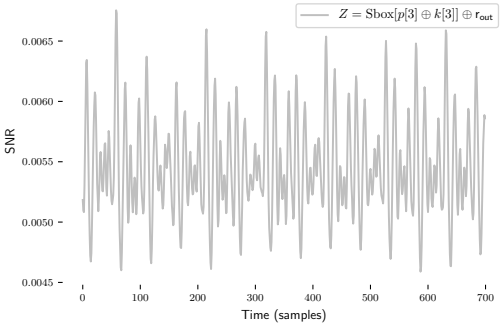
(a) Characterization with SNR



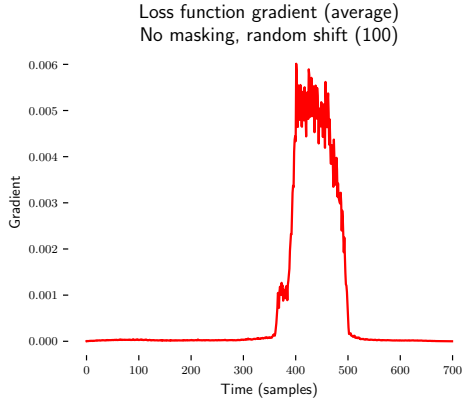
(b) Gradient Visualization

[15] Masure, Dumas, and Prouff, "Gradient Visualization for General Characterization in Profiling Attacks"; Hettwer, Gehrer, and Güneysu, "Deep Neural Network Attribution Methods for Leakage Analysis and Symmetric Key Recovery".

Sensitivity Analysis with misalignment

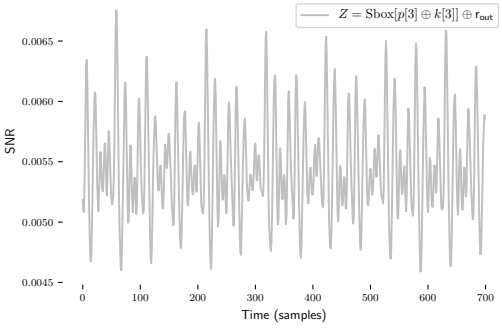


(a) No PoI emphasized ✗

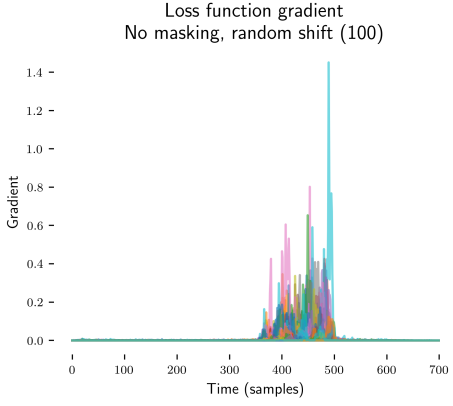


(b) Band of PoIs ✓

Sensitivity Analysis with misalignment



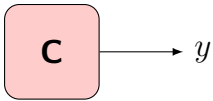
(a) No PoI emphasized ✗



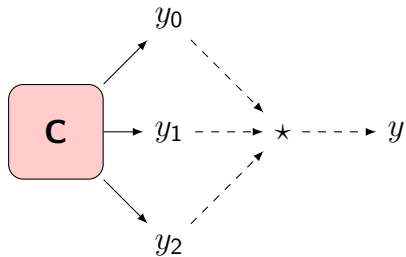
(b) Characterization for each trace ✓

Content

How to protect against SCA: Masking



(a) Unprotected

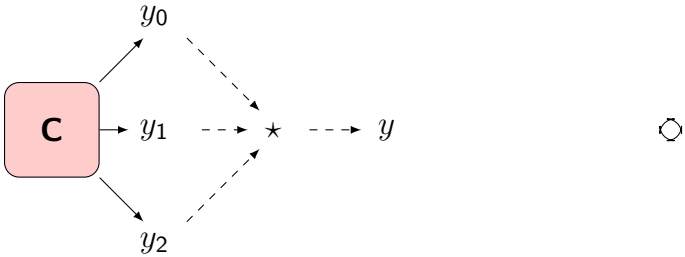


(b) Masking with $d + 1 = 3$ shares

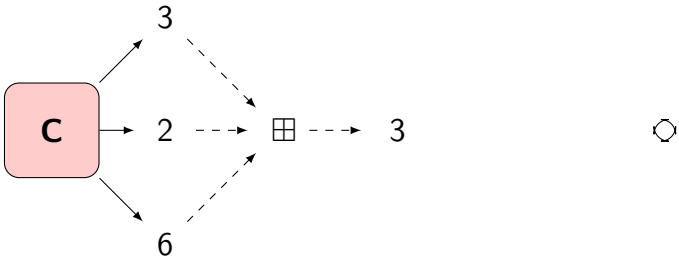
$y = y_0 \star \ldots \star y_d$

Each non-trivial subset of share: independent of y

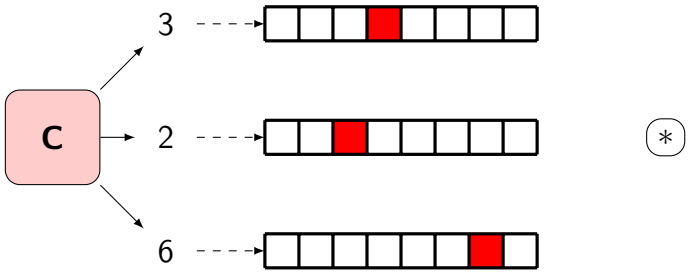
The Effect of Masking



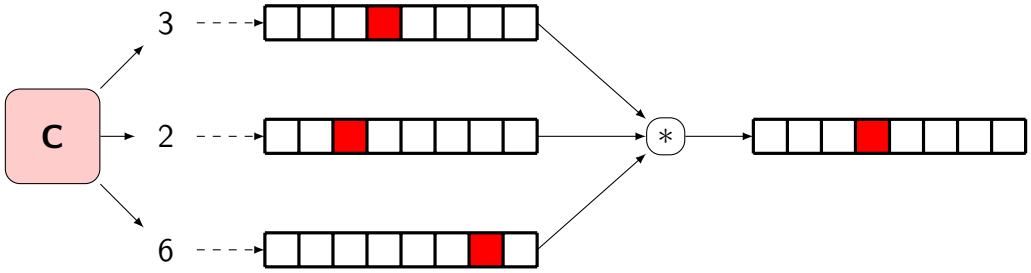
The Effect of Masking



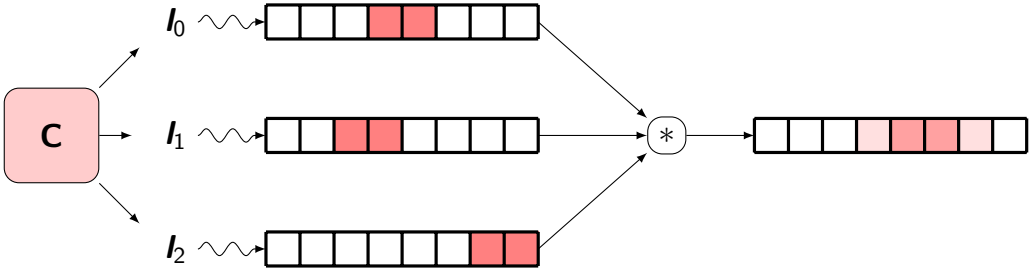
The Effect of Masking



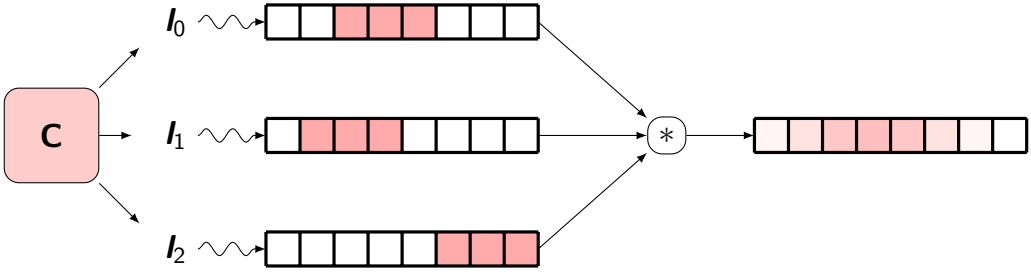
The Effect of Masking



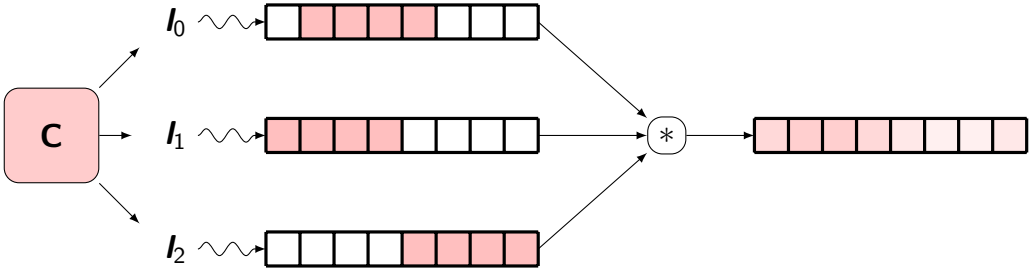
The Effect of Masking



The Effect of Masking



The Effect of Masking



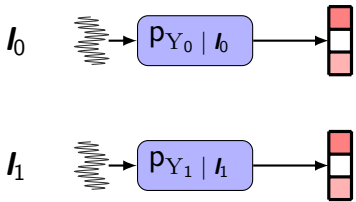
Masking amplifies the noise . . . exponentially with #shares

Independence $\Longrightarrow$ each separate leakage l_i **statistically neutral** w.r.t. the secret y

The Elephant in the Room

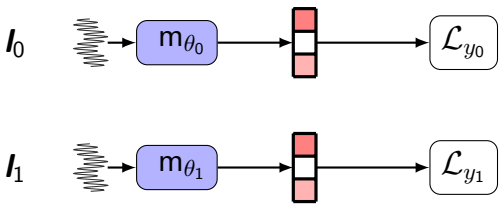
How to profile masked implementations ?

The natural way: divide & conquer
$\rightarrow \Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$



The Elephant in the Room

How to profile masked implementations ?



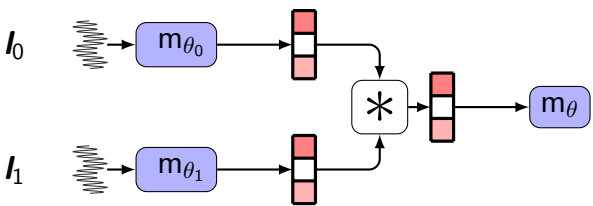
The natural way: divide & conquer

$\rightarrow$ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$

$\rightarrow$ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$

The Elephant in the Room

How to profile masked implementations ?



The natural way: divide & conquer

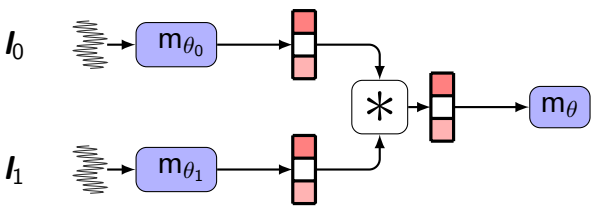
$\rightarrow$ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$

$\rightarrow$ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$

$\rightarrow$ Use $\circledast$ to recombine

The Elephant in the Room

How to profile masked implementations ?



The natural way: divide & conquer

$\rightarrow$ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$

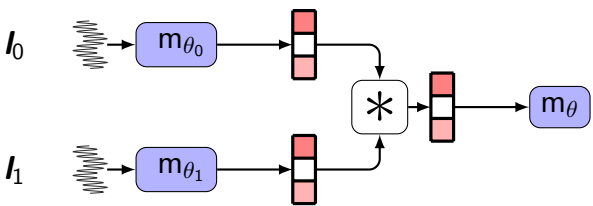
$\rightarrow$ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$

$\rightarrow$ Use $\circledast$ to recombine

Worst-case adversary:

The Elephant in the Room

How to profile masked implementations ?

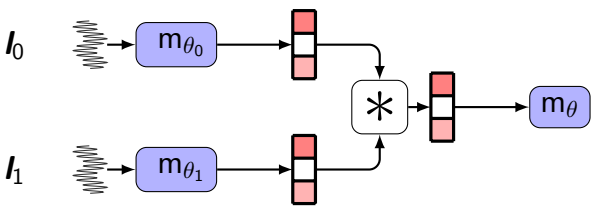


The natural way: divide & conquer
→ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$
→ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$
→ Use $\circledast$ to recombine

Worst-case adversary:
→ Aware of masking scheme

The Elephant in the Room

How to profile masked implementations ?

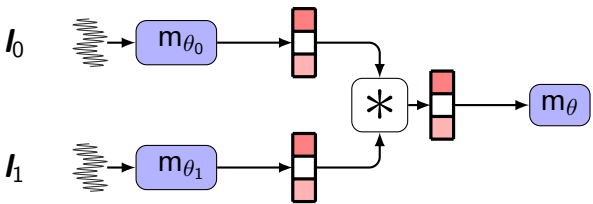


The natural way: divide & conquer
→ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$
→ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$
→ Use $\circledast$ to recombine

Worst-case adversary:
→ Aware of masking scheme
→ Access to random nonces

The Elephant in the Room

How to profile masked implementations ?

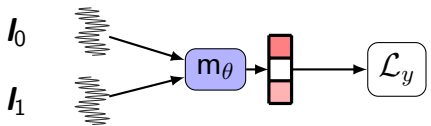


The natural way: divide & conquer
→ $\Pr(\mathrm{Y} \mid \mathbf{L})$ decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$
→ Each, modeled by m_{θ_i}, trained with $\mathcal{L}_{y_i}$
→ Use $\circledast$ to recombine

Worst-case adversary:
→ Aware of masking scheme
→ Access to random nonces

Too conservative ✗
Not realistic ✗

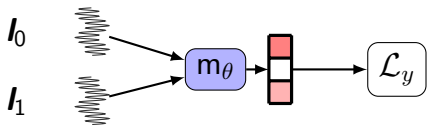
The Elephant in the Room II



The End-to-End Way:

$\rightarrow \Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

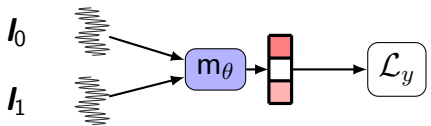
The Elephant in the Room II



The End-to-End Way:

$\rightarrow \Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

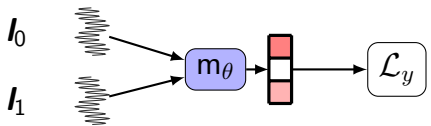
The Elephant in the Room II



The End-to-End Way:
$\rightarrow \Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

Uninformed Adversary:

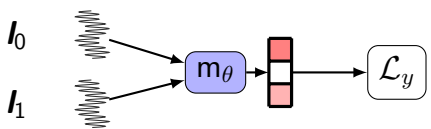
The Elephant in the Room II



The End-to-End Way:
$\rightarrow \Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

Uninformed Adversary:
$\rightarrow$ Not aware of masking scheme

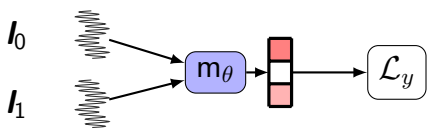
The Elephant in the Room II



The End-to-End Way:
→ $\Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

Uninformed Adversary:
→ Not aware of masking scheme
→ No access to random nonces

The Elephant in the Room II



The End-to-End Way:
→ $\Pr(Y \mid \mathbf{L})$ directly modeled by m_θ, trained with $\mathcal{L}_y$

Uninformed Adversary:
→ Not aware of masking scheme
→ No access to random nonces

More realistic ✓
Maybe sub-optimal ✗

Simulated Experiments

Learning Curves: MI estimation vs. data complexity

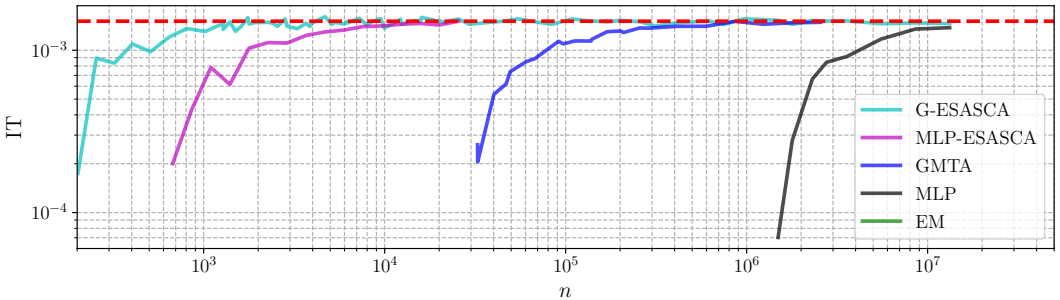


Figure: First-order masking, SNR $= 0.1$

Simulated Experiments

Learning Curves: MI estimation vs. data complexity

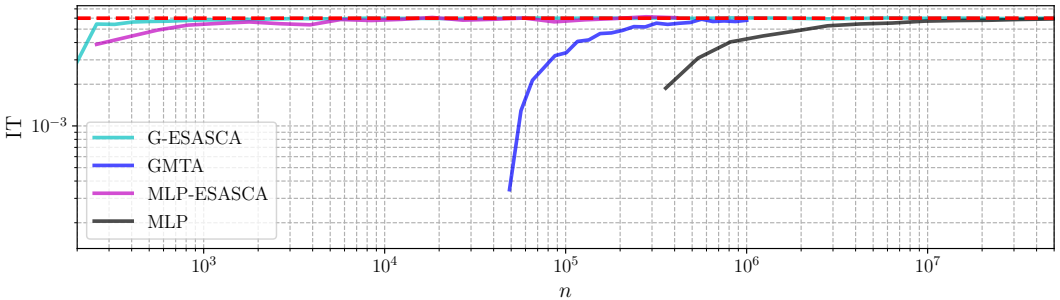


Figure: Second-order masking, SNR $= 1$

Recap

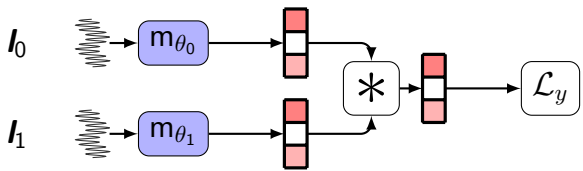
Divide & conquer approach:

- converges faster than black-box counter-parts[16]
- data complexity $\perp$ #shares, **not for uninformed adversaries**[17]!

Can we find a trade-off between both approaches ?

[16] Bronchain et al., "Efficient Profiled Side-Channel Analysis of Masked Implementations, Extended".
[17] Masure et al., *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*.

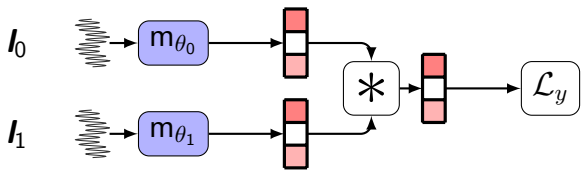
Don't Learn what You Already Know !



→ Model still decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

Don't Learn what You Already Know !

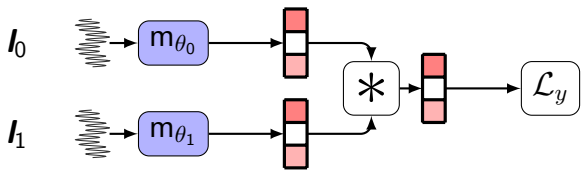


→ Model still decomposed as collection of $\Pr(Y_i \mid \mathbf{L}_i)$

→ Still recombined with $\circledast$ but ...

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

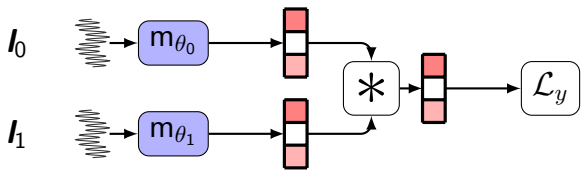
Don't Learn what You Already Know !



→ Model still decomposed as collection of $\Pr(\mathrm{Y}_i \mid \mathbf{L}_i)$
→ Still recombined with $\circledast$ but ...
→ ... Training done *jointly* with $\mathcal{L}_y$

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

Don't Learn what You Already Know !

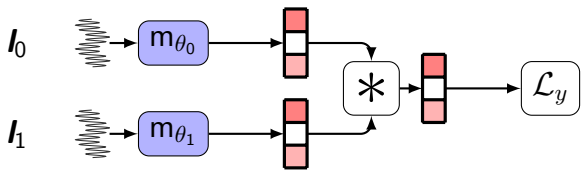


→ Model still decomposed as collection of $\Pr(Y_i \mid \mathbf{L}_i)$
→ Still recombined with $\circledast$ but ...
→ ... Training done *jointly* with $\mathcal{L}_y$

Scheme-aware adversary:[18]

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

Don't Learn what You Already Know !

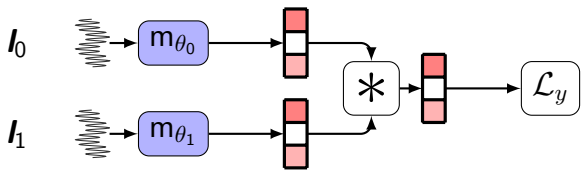


→ Model still decomposed as collection of $\Pr(Y_i \mid \mathbf{L}_i)$
→ Still recombined with $\circledast$ but ...
→ ... Training done *jointly* with $\mathcal{L}_y$

Scheme-aware adversary:[18]
→ Aware of masking scheme

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

Don't Learn what You Already Know !

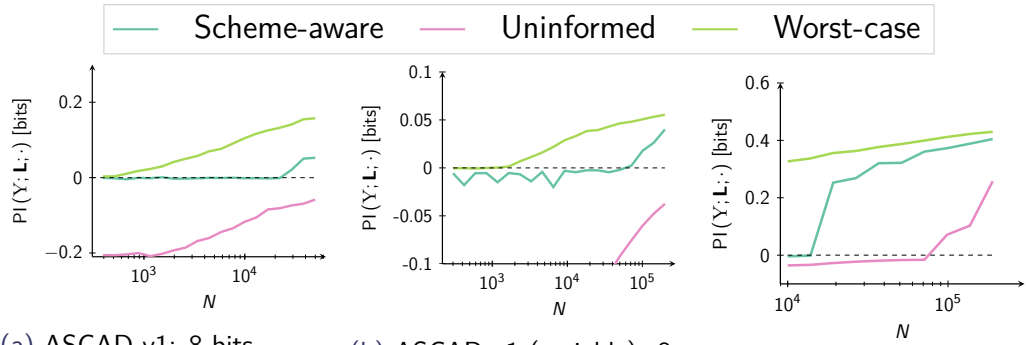


→ Model still decomposed as collection of $\Pr(Y_i \mid \mathbf{L}_i)$
→ Still recombined with $\circledast$ but ...
→ ... Training done *jointly* with $\mathcal{L}_y$

Scheme-aware adversary:[18]
→ Aware of masking scheme
→ No access to random nonces

[18] Masure et al., "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking", Ches 2023

Scheme-Aware Spares Some Data Complexity



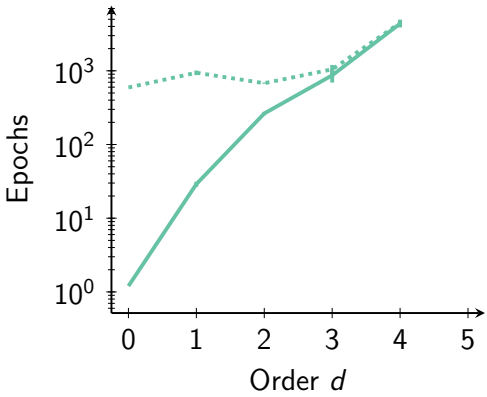
(a) ASCAD-v1: 8 bits, $d = 1$.

(b) ASCAD-v1 (variable): 8 bits, $d = 1$.

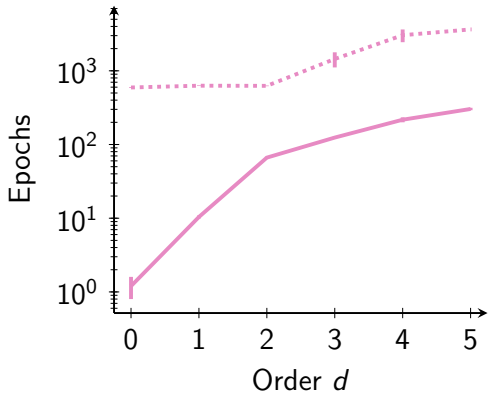
(c) Spook: 2 bits, $d = 2$.

Figure: Learning curves: MI estimation vs. data complexity.

Drawback: the Plateau Effect of Masking



(a) Scheme-aware.



(b) Uninformed.

Figure: Number of epochs required to properly train a model.

An Explanation

Theorem (informal[19])

Assume that each $\mathbf{L}_i$ *is i.i.d. standard Gaussian in* $\mathbb{R}^p$*. Define the target function* $h_{\boldsymbol{u}}(\boldsymbol{l}) = \prod_{i=1}^{d} \operatorname{sign}(\boldsymbol{u}^\intercal \boldsymbol{l}_i)$*, for some normalized hyperplane* $\boldsymbol{u}$*. Let* m_θ *be a model, such that* $\mathop{\mathbb{E}}_{\mathbf{L}} \left[\|\nabla_\theta \mathrm{m}_\theta\|^2 \right] \leq G(\theta)^2$*. Then,*

$$\mathop{\mathbb{E}}_{\boldsymbol{u}} \left[\left\| \nabla_\theta \mathcal{L}(\theta) - \mathop{\mathbb{E}}_{\boldsymbol{u}} \left[\nabla_\theta \mathcal{L}(\theta) \right] \right\|^2 \right] \leq G(\theta)^2 \cdot \mathcal{O}\left(\sqrt{\frac{d \log(p)}{p}} \right)^d . \qquad (3)$$

The gradient almost takes the same direction, regardless of y !

[19] Shalev-Shwartz, Shamir, and Shammah, "Failures of Gradient-Based Deep Learning", p. ICML 2017.

Content

Epilogue

How to tackle masking with DL remains unclear:

Gradient descent not suitable for higher orders

Efficient surrogate to gradient descent ?

$\Longrightarrow$ Then current evaluator run suboptimal attacks

No efficient surrogate to gradient descent ?

$\Longrightarrow$ Then intrinsic gap between worst-case approach and others

References I

 Bronchain, O. et al. “Efficient Profiled Side-Channel Analysis of Masked Implementations, Extended”. In: *IEEE Trans. Inf. Forensics Secur.* 17 (2022), pp. 574–584. DOI: 10.1109/TIFS.2022.3144871. URL: https://doi.org/10.1109/TIFS.2022.3144871.

References II

Cagli, E., C. Dumas, and E. 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 W. Fischer and N. 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.

References III

- Chari, S., J. R. Rao, and P. Rohatgi. "Template Attacks". In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by B. S. K. Jr., Ç. K. Koç, and C. Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 13–28. ISBN: 3-540-00409-2. DOI: 10.1007/3-540-36400-5_3. URL: https://doi.org/10.1007/3-540-36400-5_3.
- Chérisey, E. de et al. "Best Information is Most Successful: Mutual Information and Success Rate in Side-Channel Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 (2019), pp. 49–79. DOI: 10.13154/tches.v2019.i2.49-79. URL: https://tches.iacr.org/index.php/TCHES/article/view/7385.

References IV

- Dumoulin, V. and F. Visin. *A guide to convolution arithmetic for deep learning*. 2016. arXiv: 1603.07285 [stat.ML].
- Hettwer, B., S. Gehrer, and T. Güneysu. "Deep Neural Network Attribution Methods for Leakage Analysis and Symmetric Key Recovery". In: *Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers*. Ed. by K. G. Paterson and D. Stebila. Vol. 11959. Lecture Notes in Computer Science. Springer, 2019, pp. 645–666. DOI: 10.1007/978-3-030-38471-5_26. URL: https://doi.org/10.1007/978-3-030-38471-5_26.

References V

 Kim, J. 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 VI

- Maghrebi, H., T. Portigliatti, and E. 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 C. Carlet, M. A. Hasan, and V. 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.
- Mangard, S., E. Oswald, and T. Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007. ISBN: 978-0-387-30857-9.

References VII

Masure, L., C. Dumas, and E. 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 I. Polian and M. 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 VIII

- Masure, L. et al. "Deep Learning Side-Channel Analysis on Large-Scale Traces - A Case Study on a Polymorphic AES". In: *Computer Security - ESORICS 2020 - 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14-18, 2020, Proceedings, Part I*. Ed. by L. Chen et al. Vol. 12308. Lecture Notes in Computer Science. Springer, 2020, pp. 440–460. DOI: 10.1007/978-3-030-58951-6_22. URL: https://doi.org/10.1007/978-3-030-58951-6_22.
- Masure, L. et al. "Don't Learn What You Already Know: Grey-Box Modeling for Profiling Side-Channel Analysis against Masking". In: *IACR Cryptol. ePrint Arch.* (2022), p. 493. URL: https://eprint.iacr.org/2022/493.

References IX

- Masure, L. et al. *Information Bounds and Convergence Rates for Side-Channel Security Evaluators*. Cryptology ePrint Archive, Paper 2022/490. https://eprint.iacr.org/2022/490. 2022. URL: https://eprint.iacr.org/2022/490.
- Paninski, L. "Estimation of Entropy and Mutual Information". In: *Neural Comput.* 15.6 (2003), pp. 1191–1253.
- Shalev-Shwartz, S., O. Shamir, and S. Shammah. "Failures of Gradient-Based Deep Learning". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by D. Precup and Y. W. Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 3067–3075. URL: http://proceedings.mlr.press/v70/shalev-shwartz17a.html.

References X

- Simonyan, K. and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015. URL: http://arxiv.org/abs/1409.1556.
- Timon, B. "Non-Profiled Deep Learning-based Side-Channel attacks with Sensitivity Analysis". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.2 (2019), pp. 107–131. DOI: 10.13154/tches.v2019.i2.107-131. URL: https://tches.iacr.org/index.php/TCHES/article/view/7387.

References XI

Zaid, G. et al. "Methodology for Efficient CNN Architectures in Profiling Attacks". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2020.1 (2019), pp. 1–36. DOI: 10.13154/tches.v2020.i1.1-36. URL: https://tches.iacr.org/index.php/TCHES/article/view/8391.