CS230

Deep Learning: Project Final Report

Casey Long[*]
Department of Computer Science
Stanford University
clong80@stanford.edu

Abstract

Deep learning neural networks aid in side channel cryptanalysis against AES-128 implementations running on an 8-bit RISC (AVR) CPU architecture.

1 Introduction

Power Analysis side-channel attacks correlate power consumption of crytographic operations to estimate both a key value and its timestamp in a power *trace*. A *trace* refers to a set of power consumption measurements taken across a crytographic operation. Often, this is depicted as an X-Y plot, with current or voltage on the Y-axis and time on the X-axis.

A Simple Power Analysis (SPA) involves directly interpreting the visual trace. Because block cipher encryption algorithms like AES are deterministic and public, correlating the power consumption to certain crytographic operations can reveal execution and data path points of the algorithm. The role of the crytographic engineer is to prevent *leakage* of crytographic operations in *traces* to an adversary. Often, this employs the use of power reduction to minimizes signal strength or introduction of noise to minimizes measurement strength.

More advanced power analysis side-channel attack takes advantage of large datasets of *traces* to measure small variations of power consumption. These variations are not intuitively obvious and differences are expressed in terms of covariances.

2 Background

A more thorough investigation of the AES-128 algorithm is deferred to other papers. The salient points of the AES-128 algorithm important to this paper are expressed:

- There are four main functions; `add_round_key`, `substitute_bytes`, `shift_rows`, and `mix_columns`. They are permutations of each other; i.e., they are chained together.
- The `add_round_key` and `substitute_bytes` functions are of particular interest to side-channel analysis, and involve a bitwise `XOR` and constant-time table lookup, respectively. This table is known as the Rijndael S-Box.
- `substitute_bytes` represents a non-linear mapping that breaks the 128-bit key produced by `add_round_key` into 16 bytes, which serve as the index to the S-Box lookup. Because of its non-linearity, it is difficult to alter or safeguard this function while preserving this mapping. In addition to the small 8-bit index, `substitute_bytes` **is a source of weakness.**

[*]Stanford CS230 student.

CS230: Deep Learning, Spring 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

During a DPA attack, an attacker targets a single 8-bit block. This is what gives DPA attacks their strength; brute force complexity is reduced from $\mathcal{O}(2^{128})$ to $\mathcal{O}(16 \cdot 2^8) \approx \mathcal{O}(2^8)$.

The *key schedule* for the AES-128 algorithm is provided.

Here is a general procedure of a differential attack.

- We assume *a priori* knowledge of the plaintext or ciphertext (or both) values for a fixed key value and the algorithm used. Intuitively, this means we have access to the device under attack (DUT), and are able to monitor the encryption or decryption process. This is possible with things such as encrypted bootloaders where we can continually reset the device, or commercial-off-the-shelf equipment that is not unique.
- We want to calculate the maximum likelihood estimate (MLE) for the key-byte. Because it is only 8-bits, we can brute force this. Thus, we have $2^8 = 256$ possible "classes" that we can bin each estimate into.
- For each key estimate, we correlate it to the power *trace*. This correlation uses some *leakage function* that maps the key-byte to a power intensity. A common choice for this *leakage function* is the *Hamming Weight*. The intuition here, is that an "on" or "1" bit is related to power consumption.

3 Related work

Many power analysis algorithms can provide an MLE. The ones discussed in this paper include a Correlation Power Analysis (CPA), Linear Regression Analysis (LRA), a Multilayer Perceptron (MLP), and a Convolutional Neural Network (CNN).

29 one of the most important works in the field of side-channel analysis. Published in 1996, this seminal work introduces one of the first feasible concepts of statistical analysis of *trace* datasets to attack a microcontroller, called a Differential Power Analysis. Many other power analysis algorithms such as CPA and LRA are directly derived from this work. While DPA in this work refers to a specific Difference of Means algorithm to measure differences, the word "differential power analysis" is often interchangably used in any power analysis attack that uses some statistical difference measurement to gain inference about the MLE key-byte.

28 introduces a Correlational Power Analysis.

26 introduces a Linear Regression Analysis.

- Additionally, a literature review indicates sucess using Support Vector Machines (SVM) [14][15][16][17], Random Forests [17][18], Multilayer Perceptrons (MLP) [19][20][21], and Convolutional Neural Networks (CNN) [22][23]. Due to relevancy of the last two in deep learning, more attention will be emphasized there.

4 Dataset and Features

4.1 Description

The dataset consists of 60,000 AES-128 power traces extracted from ATMega8515 (AVR architecture) microcontroller, partitioned into 10,000 test and 50,000 train cases. It is a time series dataset. Each data point consists of three groups of information:

- **traces**: contains an index number, with a timestamp and raw power measurement
- **labels**: the AES substitution box (i.e., a Rijndael S-box) values. We denote $S(\mathtt{p} \oplus \mathtt{k})$ as the substition box, where $\mathtt{p}$ is our plaintext value $\mathtt{k}$ is our key value.
- **metadata**: associated with every timestamp is the truth values for the plaintext, ciphertext, key, and mask used during that timestamp. A mask is an obfuscation technique to protect AES implementations by randomizing the intermediate results, thus creating noise to power traces. Not all traces are masked.

This 5Gb dataset is freely available from the National Cybersecurity Agency of France (Agence nationale de la sécurité des systèmes d'information, ANSSI). The ANSSI Side Channel Attack

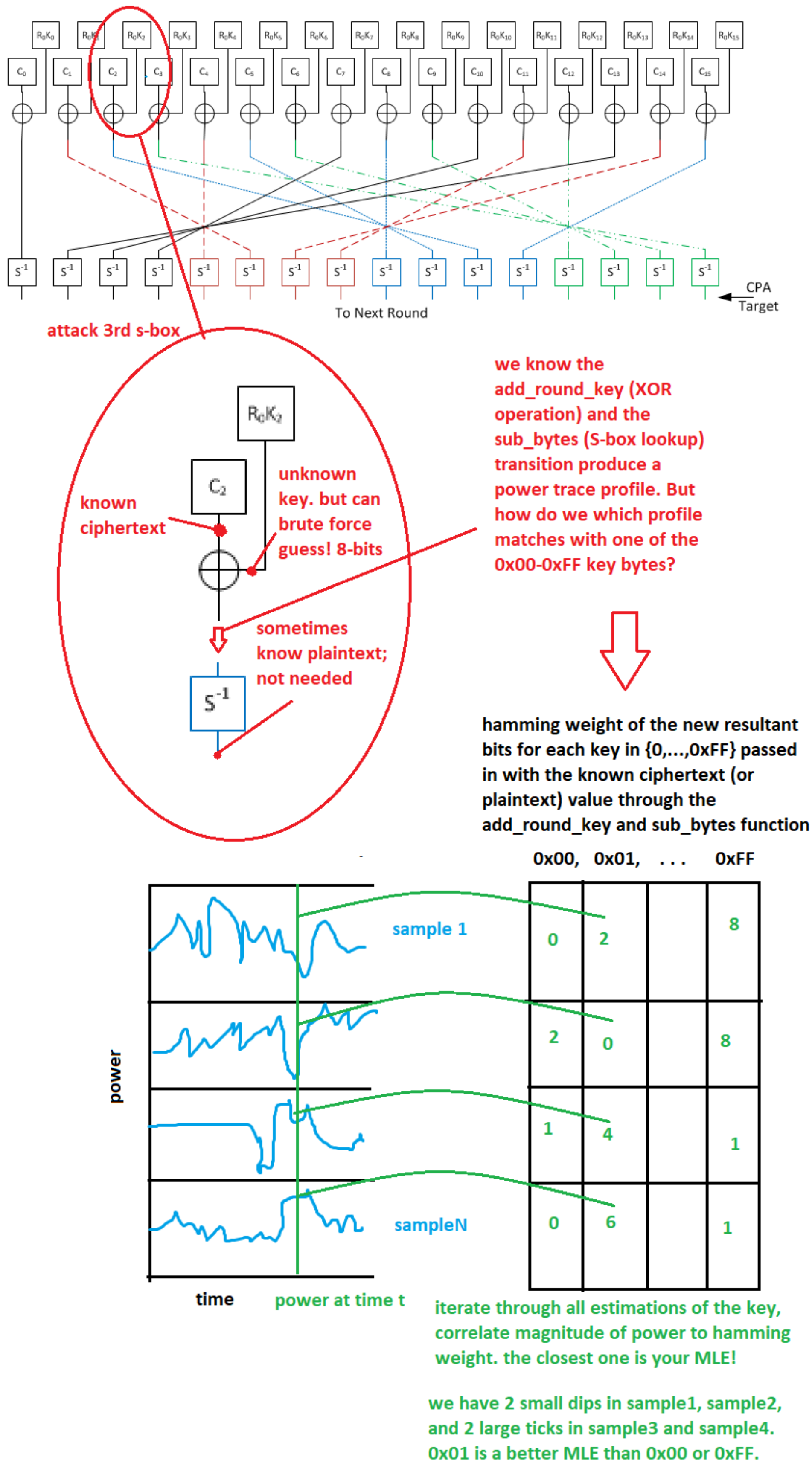

Figure 1: *snr4*

Database (ASCAD) is in HDF5 format, which can be parsed with Python's `hdf5` package. ANSSI developed this database with the intention of it becoming a MNIST-like library for side-channel attacks. Support for Keras, Tensorflow, and GPU acceleration is provided.

4.2 Dimensionality Reduction

According to the Nyquist-Shannon sampling theorom, the measurement frequency must be higher than the measured device under attack frequency (i.e., the clock rate). Often times, measurement frequency may be in the GHz range for microcontrollers in the MHz. This is often due to the resolution required for a certain attack and the low power emitted from such devices that make useful measurements sparse, and most measurements noisy. [] lists various dimensionality reduction techniques to hone in onto Points of Interest. These include Difference of Means based methods (DOM), Sum of Sqaured Differences (SOSD), Correlation Power Analysis based methods (CPA), Sum of Squared pairwise T-differences (SOST), Signal-to-Noise ratio (SNR), Variance based methods (VAR), Mutual Information Analysis (MIA), and Kolmogorov-Smirnov Analysis (KSA), and Principal Component Analysis (PCA).

The SNR method was chosen as the ASCAD authors also used this method. The SNR is generally defined as,

$$SNR = \frac{Signal}{Noise} \tag{1}$$

In this paper, it is specifically defined as

$$SNR = \frac{\sigma_{\bar{x}} - \mu_{\bar{x}}}{\mu_{\sigma^2}} \tag{2}$$

Intuitively, the numerator represents the difference in the mean of different class means and the variance of those average means, while the denominator represents the mean of the variances. The graph show is the result, and is a replication of what ASCAD similarily produced. The range of [45400, 46100] was chosen because leakage model functions *snr4* and *snr5* were too simple. Their representations are easy to spot. The function *snr1* was not able to be seen, as this was considered a cryptographically secure implementation with no first-order leakage (i.e., simple linear attacks will not be sufficient for predicting key byte values).

5 Methods

A Correlation Power Analysis and Linear Regression Analysis were used to baseline the model. This was to help compare the MLP and CNN models to older and more traditional models to give perspective on their effacacy.

- **CPA:** The CPA is a collary of the Pearson correlation coefficient:

$$\begin{aligned} \rho_{X,Y} &= \frac{cov(X,Y)}{\sigma_X \sigma Y} && (3) \\ &= \frac{N\sum Tr_i H_i - \sum Tr_i \sum H_i}{\sqrt{N\sum Tr_i^2 - (\sum Tr_i)^2}\sqrt{N\sum H_i - (\sum H_i)^2}} && (4) \end{aligned}$$

  where Tr_i is our trace at sample index i, s.t. $i \in \{0, ..., N_{traces}\}$ and H_i is shorthand for $HammingWeight(sbox(P_i \oplus K_i))$ Intuitively, we are trying to map the raw power intensity to the Hamming Weight of the bits produced by the `add_round_key` and `substitute_bytes` functions.

- **LRA:** An LRA is conceptually similar to any other linear regression. We are trying to fit our key-byte classes to a spline, instead of a fixed dimensional polynomial. In this case, we create basis functions in the matrix M. The mathematics behind these functions delve more into crytography and finite fields, so we won't delve into that rabbit hole. The big take away is that we create a coefficient matrix that we can apply to our trace data. These splines represent a fit approximation to each of the 256 classes that 8-bit target key byte can represent.

$$M = \begin{bmatrix} sbox(P_i \oplus K_i)_1^{b_1} & \dots & sbox(P_i \oplus K_i)_1^{b_t} \\ \vdots & \ddots & \\ sbox(P_i \oplus K_i)_N^{b_1} & & sbox(P_i \oplus K_i)_N^{b_t} \end{bmatrix} \tag{5}$$

$$\beta = (M^T M)^{-1} M^T Tr \tag{6}$$

The goodness of fit measure is a scalar representation from $\{0, ..., 1\}$ of how close a measurement value from $\mathbf{Tr}$ is to our model $\mathbf{M} \cdot \boldsymbol{\beta}$

$$R^2 = 1 - \frac{\sum (Tr - M\beta)^2}{\sum (Tr_i - \bar{T}r)^2} \tag{7}$$

$$= 1 - \frac{||\mathbf{Tr} - \mathbf{M} \cdot \boldsymbol{\beta}||}{\sigma_{Tr}} \tag{8}$$

- **MLP:** The MLP used has a total of 6 layers used. The first input layer consists of 700 notes–this is due to the Point of Interest interval of [45400, 46100]. This input layer represents the power intensities for that given time range. The next four layers are hidden layers of 200 nodes each. The final output layer is 256 node layer, representing the MLE prediction of the byte. (i.e., it's predicting each individual bit) A categorical cross entropy loss function was used. This was because the categorical cross entropy loss function weights the true "class" (our byte) with a value of 1.0, and all other classes of 0.0. This is useful because crytography has to have a deterministic and exact key-byte value; we would like to weight non-true answers as low as possible. The activation function for the four hidden layers was ReLu, with a softmax activation for the final layer. The optimizer was RMSProp–no specific reason why this was chosen. The number of nodes and hidden layers seem to have a more disceranable impact on performance for these side-channel attacks, as discussed in. A great deal of time was instead devoted towards comparing MLP to other older/traditional methods of inference. The results are shown.

6 Experiments/Results/Discussion

First-order side channel attacks are those that exploit differences in means. . A Linear Regression Analysis (LRA) and a Correlation Power Analysis (CPA) are examples of first order attacks. An attack implementation of the leakage functions *snr4, snr2, snr1* is provided. While no graphs were provided in the ASCAD paper to reference, the qualitative descriptions coincide with the results I achieved. Mainly, that in terms cryptographical secureness, *snr1* precedes *snr2*, which precedes *snr4*.

In these graphs, the grey represents an overlay of the estimated attack traces (the 256 key byte guesses). The green, purple, or blue represents the MLE key byte. The red represents the correct key byte if the MLE key was guessed incorrectly. It should not be surprising that the attacks on *snr1* do not converge. It was deemed a crytogrpahically secure implementation on the 8-bit microcontroller with no first-order leakage. You can easily see that there are no big spikes like there are in *snr2* or *snr4*, and most of the blue represents consistently within the gray attack traces, indicating its operation for correct/incorrect key values is consistent.

The MLP result is more telling. In *snr4* it quickly converges towards the solution in one iteration. Even more surprising, the previous stronger *snr1* leakage function was able to be broken and the key correctly guessed.

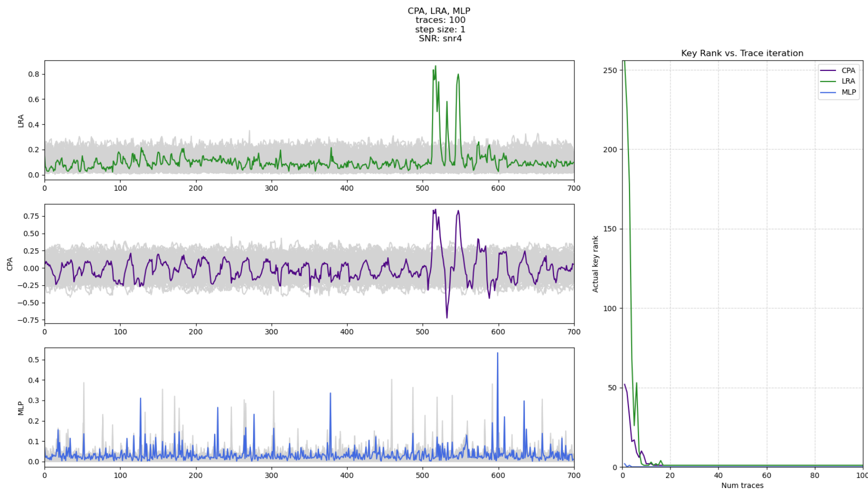

Figure 2: *snr4*

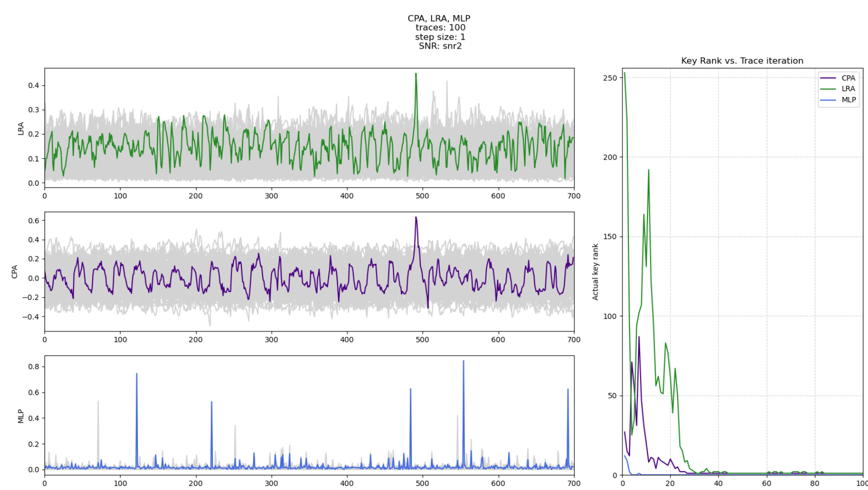

Figure 3: *snr4*

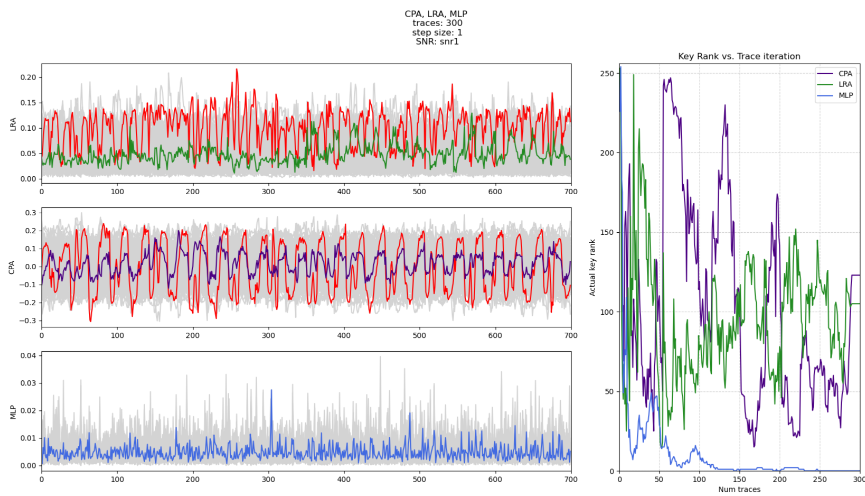

Figure 4: *snr4*

7 Conclusion/Future Work

Unfortunately running out of time to do more work on the CNN. This project was quite a bit of work, delving into more details of Crytography and Machine Learning! It was a good learning experience, but the scope of this project is far to large for a single course.

References

[1] Szefer, J. (2018) Principles of Secure Processor Architecture Design. In Martonosi, M. & Hill, M.D. (eds.) *Synthesis Lectures on Computer Architecture.* pp. 10.

[2] Lee, R.B., Kwan, P., McGregor, J.P., Dwoskin, J. & Wang, Z. (2013) Security Basics for Computer Architects. *Synthesis Lectures on Computer Architecture.* 8(4) pp. 1-111.

[3] Lipp, M., Schwarz, M., Gruss, D., Prescher, T., Haas, W., Mangard, S., Kocher, P., Genkin, D., Yarom, Y. & Hamburg, M. (2018) Meltdown: Reading Kernel Memory from User Space. *27th USENIX Security Symposium, Baltimore, MD, USA, August 15-17, 2018.*

[4] Kocher, P., Horn, J., Fogh, A., Genkin, D., Gruss, D., Haas, W., Hamburg, M., Lipp, M., Mangard, S., Prescher, T., Schwarz, M. & Yarom, Y. & Hamburg, M. (2019) Spectre Attacks: Exploiting Speculative Execution. *40th IEEE Symposium on Security and Privacy (S&P'19).*

[5] Szefer, J. (2018) Principles of Secure Processor Architecture Design. In Martonosi, M. & Hill, M.D. (eds.) *Synthesis Lectures on Computer Architecture.* pp. 27-29.

[6] Department of Homeland Security, National Cyber Awareness System. (2018) *Alert (TA18-004A) Meltdown and Spectre Side-Channel Vulnerability Guidance.* `https://www.us-cert.gov/ncas/alerts/TA18-004A` Original Release: January 4, 2018. Revised May 01, 2018. Retrieved April 25, 2020.

[7] Graz University of Technology. (2018) *Meltdown and Spectre Vulnerabilities in modern computers leak passwords and sensitive data.* `https://meltdownattack.com/faq-advisory` Retrieved April 25, 2020.

[8] Intel. (2018) *Advancing Security at the Silicon Level* `https://newsroom.intel.com/editorials/advancing-security-silicon-level/#gs.4hdzye` Original Release: March 15, 2018. Retrieved April 25, 2020.

[9] Microsoft. (2018) *ADV180002 | Guidance to mitigate speculative execution side-channel vulnerabilities.* `https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/ADV180002` Original Release: January 3, 2018. Revised June 14, 2019. Retrieved April 25, 2020.

[10] The Linux Kernel. (2020) *Hardware vulnerabilities, Spectre Side Channels* `https://www.kernel.org/doc/html/latest/admin-guide/hw-vuln/spectre.html` Retrieved April 25, 2020.

[11] Bursztein, E. & Picod, J-M. (2019) A Hacker Guide to Deep Learning Based Side Channel Attacks. Defcon27 `https://elie.net/talk/a-hackerguide-to-deep-learning-based-side-channel-attacks/` Retrieved April 25, 2020.

[12] Benadjila, R., Prouff, E., Strullu, R., Cagli, E. & Dumas, C. (2019). Deep learning for side-channel analysis and introduction to ASCAD database. *Journal of Cryptographic Engineering.*

[13] Cagli, E. (2018). Feature Extraction for Side-Channel Attacks. *Cryptography and Security [cs.CR].* pp. 37-39. Sorbonne Université.

[14] Bartkewitz, T. & Lenke-Rust . (2013) Efficient Template Attacks Based on Probabilistic Multi-class Support Vector Machines. In Mangard, S. (eds.), *Smart Card Research and Advanced Applications CARDIS, volume 7771 of Lecture Notes in Computer Science.* pp. 263-276. Springer Berlin Heidelberg.

[15] Heuser, A. & Zohner, M. (2012) Intelligent machine homicide-breaking cryptographic devices using support vector machines. In Schindler, W. & Huss, S. A. (eds.) *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings, volume 7275 of Lecture Notes in Computer Science.* pp. 249-264. Springer.

[16] Hospodar, G., Gierlichs, B., De Mulder, E., Verbauwhede, I. & Vandewalle J. (2011) *Machine learning in side-channel analysis: a first study. J. Cryptographic Engineering.* 1(4) pp. 293–302.

[17] Lerman, L., Bontempi, G. & Markowitch, O. (2014) *Power analysis attack: an approach based on machine learning. International Journal of Advanced Computer Technology.* 32 pp. 97-115.

[18] Lerman, L., Poussier, R., Bontempi, G., Markowitch, O. & Standaert, F-X. (2015) *Template attacks vs. machine learning revisited (and the curse of dimensionality in side-channel analysis).* In Mangard, S. (eds.),

Constructive Side-Channel Analysis and Secure Design - 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers, volume 9064 of Lecture Notes in Computer Science. pp. 20-33. Springer.

[19] Martinasek, Z., Dzurenda, P. & Malina, L. (2016) Profiling power analysis attack based on MLP in DPA contest V4.2 In *39th International Conference on Telecommunications and Signal Processing, TSP 2016, Vienna, Austria, June 27-29, 2016.* pp. 223–226. IEEE, 2016.

[20] Martinasek, Z., Hajny, J., & Malina, L. (2015) Optimization of power analysis using neural network. In *Francillon and Rohatgi*, pp. 94–107.

[21] Martinasek, Z., Malina, L. & (2015) K. Trasy. Profiling Power Analysis Attack Based on Multi-layer Perceptron Network. *Computational Problems in Science and Engineering, 343.*

[22] Cagli, E., Dumas, C. & Prouff, E. Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing. In Fischer, W. & Homma, N (eds.) *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings, volume 10529 of Lecture Notes in Computer Science.* pp. 45–68. Springer.

[23] Maghrebi, H., Portigliatti, T., & Prouff, E. Breaking cryptographic implementations using deep learning techniques. In Carlet, C. M., Hasan, A. & Saraswat, V. (eds.) *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings, volume 10076 of Lecture Notes in Computer Science.* pp. 3–26. Springer.

[24] Standaert, F-X., Koeune, F. & Schindler, W. (2009) How to Compare Profiled Side-Channel Attacks? In Abdalla M., Pointcheval D., Fouque PA. & Vergnaud D. (eds.)*Applied Cryptography and Network Security. ACNS 2009. Lecture Notes in Computer Science, vol 5536* pp. 485-498. Springer, Berlin, Heidelberg.

[25] Oswald E., Mangard S., Pramstaller N., Rijmen V. (2005) A Side-Channel Analysis Resistant Description of the AES S-Box. In: Gilbert H., Handschuh H. (eds) Fast Software Encryption. FSE 2005. Lecture Notes in Computer Science, vol 3557. Springer, Berlin, Heidelberg

[26] Brier E., Clavier C., Olivier F. (2004) Correlation Power Analysis with a Leakage Model. In: Joye M., Quisquater JJ. (eds) Cryptographic Hardware and Embedded Systems - CHES 2004. CHES 2004. Lecture Notes in Computer Science, vol 3156. Springer, Berlin, Heidelberg

[27] Fan G., Zhou Y., Zhang H., Feng D. (2015) How to Choose Interesting Points for Template Attacks More Effectively?. In: Yung M., Zhu L., Yang Y. (eds) Trusted Systems. INTRUST 2014. Lecture Notes in Computer Science, vol 9473. Springer, Cham

[28] Schindler W., Lemke K., Paar C. (2005) A Stochastic Model for Differential Side Channel Cryptanalysis. In: Rao J.R., Sunar B. (eds) Cryptographic Hardware and Embedded Systems – CHES 2005. CHES 2005. Lecture Notes in Computer Science, vol 3659. Springer, Berlin, Heidelberg

[29] Kocher P., Jaffe J., Jun B. (1999) Differential Power Analysis. In: Wiener M. (eds) Advances in Cryptology — CRYPTO' 99. CRYPTO 1999. Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg