Side-Channel Attacks and Machine Learning Approach

Alla Levina, Daria Sleptsova, Oleg Zaitsev
ITMO University
Saint Petersburg, Russia
levina@cit.ifmo.ru, dsleptsova@corp.ifmo.ru, zaitsev@voltus.ru

***Abstract*—Most modern devices and cryptoalgorithms are vulnerable to a new class of attack called side-channel attack. It analyses physical parameters of the system in order to get secret key. Most spread techniques are simple and differential power attacks with combination of statistical tools. Few studies cover using machine learning methods for pre-processing and key classification tasks. In this paper, we investigate applicability of machine learning methods and their characteristic. Following theoretical results, we examine power traces of AES encryption with Support Vector Machines algorithm and decision trees and provide roadmap for further research.**

I. Introduction

Cryptanalysis is a study that investigates ciphers and cryptosystems in order to find vulnerabilities and exploits them to get secret information used in such systems and ciphers or in some other way retrieve the plaintext from the ciphertext. Commonly cryptographic algorithms are studied from the viewpoint of the mathematics, because cryptography in digital application is a mathematical study. However a new direction developed since 1996, that considers algorithm altogether with its physical implementation[1]. Side-channel attacks is a field in cryptanalysis that exploits information gained from the physical implementation of a cryptosystem, rather than theoretical weaknesses in the algorithms. Real-world cryptoalgorithms are certainly implemented on some device, like PC or a dedicated hardware module, so the device's properties can be used to break the algorithm. Side channels include, but not bounded to, power consumption, electromagnetic emanations and acoustic emissions. Another side-channel that stands apart is fault analysis. Fault analysis is always an active attack, which means direct intervention to the operation of a device, and it is performed by inducing faults in the device (e.g. using power glitches) and basing on the output recovers the plaintext. [2]

The majority of works in this field has power analysis is under examination as the most fundamental and efficient side-channel. Timing analysis is less of concern nowadays, electromagnetic emanations are commonly a derivative from power fluctuations and acoustic emission covers narrow range of device, being on the border with electromagnetic emanations. Moreover, several steps were made towards standardization of power leakage measurement including creation of a hardware platform [3], and several contest aimed on evaluation of leakage of AES and DES [4]. Power traces used in this paper include those provided for DPA contest v2 [4] and Tescase AES traces [5]. In our work we analyse power consumption traces of AES algorithm, performed on the SASEBO-GII board with application of machine learning to key recovery and traces pre-processing as well as work out background for the attack and characteristics effect on the performance.

Our paper is organized as follows: second section presents an introduction to side-channel attacks with a stress on phases, where machine learning can be applied. In section III current state of art is presented with practical examples of attacks. Sections IV and V disclose in details two selected for research algorithms. In section VI practical results are presented. Conclusions and further work can be found in the last section.

II. Side-channel attack

There are two main types of side-channel attacks. Non-profiled attack makes use of a leakage model, which consist of the output value of encrypting function for each target value, and then compares real leakage to predicted model. More stronger assumption is that an adversary can perform a profiling phase, which measures probability of the subkey hypotheses, and is called profiled attack.

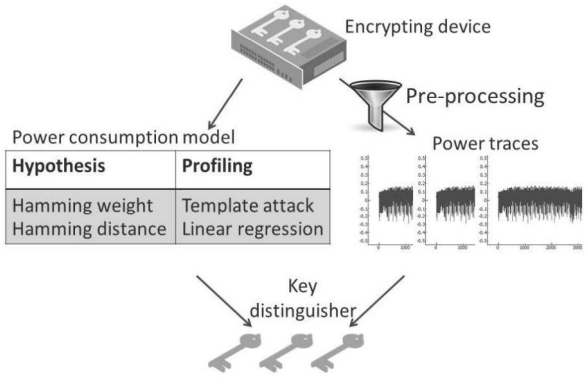

Fig. 1. Typical scenario for a side-channel attack

On the Fig. 1 the typical attack scenario is shown. An attack can be divided into two phases:

1) *Training phase* (left side of the Fig. 1). In this phase an attacker builds a model based on some hypothesis, for example, Hamming weight or Hamming distance between two processed values as in differential attack. He or she can also perform more stronger technique called profiling.

At profiling an attacker controls a copy of the device identical to the target and can perform any operation he/she wants. Cryptographic key at the profiling phase is at attacker's disposal as well, following the Kerkchoff's principle, so the attacker can change key, choose any plaintext as the input to the device. The device can execute sufficient times of encryption operations for power trace acquisitions.
Output of this phase is key hypotheses modeled with selected metric and selection function.

2) *Attack phase* (right side of the Fig. 1). In this leakage on the attacked device is measured, power traces are pre-processed to diminish dimensionality. Ready traces are compared with predicted leakage using distinguisher algorithm. The most common technique in this case is correlation computation. Pearson correlation coefficient ρ for information component t of all measured traces between predicted leakage L_p and measured leakage $L_m(t)$ is defined as follows:

$$\rho(t) = \frac{Cov(L_p, L_m(t))}{\sqrt{Var(L_p) \cdot Var(L_m(t))}} \quad (1)$$

where Cov is covariance and Var defines variance.

Two main objectives for researchers are pre-processing or feature selection, that filters out traces with low informational component and extracts most useful points for the following steps, and classification, that allows to distinguish secret key basing on the device model and real acquired traces.

Pre-processing is as well needed in order to diminish the set of points in trace, as high-order signal takes more time to process in the attack. Common side-channel metrics on the preprocessing step are Pearson correlation coefficient (aligns traces) and signal-to-noise ratio, which is the ratio between the power consumption and the standard deviation of power leakage. Another improvement of an attack and reduction of the traces step can be made using sum of squared pairwise differences (SOSD) and SOST, that utilises a Student T-test instead of a difference as in SOSD. The latter was proved to be more optimal [6]. In [7] new statistical pre-processing technique is proposed. Normalized Inter-Class Variance does not require neither a clone device nor the knowledge of secret parameters and can also be used for testing leakage models and countermeasure implementation.

A. Template attack

Template attacks are the strongest kind of side-channel attack. It has similar premises to profiling phase, but more precisely refers to modeling the target device's algorithm using all possible keys. Captured and processed data for each key is called template. It might include noise characteristic as well, along with information component (points of interest, leaking knowledge about secret data). As in profiling phase big amount of data is collected, pre-processing techniques to overcome computational complexity are also applied. The basic idea of the template attack is to model the power consumption as a high dimensional Gaussian distribution dependent on few key bits.

On the attack phase power trace for some secret key is captured. Using existing templates for each key, the task is to classify the trace to one or more templates. Examples of successful attack can be found in [8] and with combination of machine learning in [9].

III. Machine learning in side-channel attacks

First ever mention of using machine learning in cryptography belongs to Ronald L. Rivest in his work "Cryptography and Machine Learning" [2]. Noticing the similarity between the said fields, between notions, e.g. cryptographic "secret key" and "target function" in machine learning, he poses the question of applicability and provides some initial examples. As cryptanalysis develops, arises new field – side-channel attacks, that yields even more objectives than classical cryptography. A lot of them, e.g. preprocessing, adapt machine learning for problem solving.

Application of machine learning particularly in side-channel attack was in the work concerning printer acoustic emanation [10]. Combination of different techniques including neural networks allowed authors to recover up to 95% text by listening the sound of a printer in work.

However, generally, side-channel attacks aim on cryptography and try to recover secret data, such as keys, from side signal. AES is studied in [11] with the focus on Least Squares Support Vector Machines. No key recovery was made in said research, as emphasis was rather made on binary classification application in side-channel. Similar approach is used in [12], where Least Square Support Vector Machines were studied to perform an unsupervised attack on AES.

Successful attack on AES using SVM is presented in [13]. It should especially be noted that attacked version of AES is masked, so this side-channel countermeasure was completely overturned by well-trained SVM classifiers.

Another attack that made use of SVM is [14] with target smartcard running DES. Authors proved that SVM outperforms traditional DPA, because of impossibility of varying keys, and SPA, because of signal complexity, as well as template attacks, which are efficient, but more complicated to apply.

Random Forests and Self Organizing Maps have also proved to be useful in key distinguishing tasks. AES was attacked using Random Forest in [15]. In [16] 3DES was attacked using Random Forest and Self Organizing Map and their combination as a pre-processing and classification methods. Same authors further study Random Forest in the attack on AES along with SVM, template attacks and multivariate regression analysis in [17].

Overall, it was shown, that if traces follow a parametric Gaussian distribution, machine learning methods perform worse than template attacks as template attacks are based on Gaussian model [18].

Pre-processing step has widely adopted a method called principal component analysis (PCA). It is a orthogonal, non-parametric transformation, which is aimed on computing a new basis for the traces, so that it discloses its principal structure. Shown on the Fig. 2 Principal Components Analysis chooses the first PCA axis as that line that goes through the centroid,

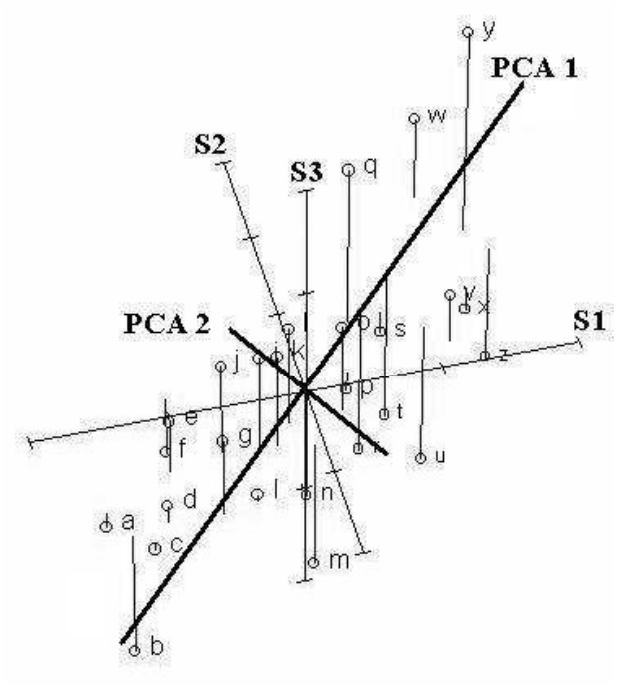

Fig. 2. Principal Component Analysis method

but also minimizes the square of the distance of each point to that line. Thus, in some sense, the line is as close to all of the data as possible. Equivalently, the line goes through the maximum variation in the data. Addresing the question of components quantity there are two main approaches: threshold can be defined as a percentage of total variation or by a scree test. The latter means that desired number of components is derived empirically from the plot of eigenvalues(intermediate values used in PCA).

However, PCA can be used as a distinguisher as well, as in work [19] attacking masked DES. Reference [20] covers PCA applicability to side-channel attacks in more details.

Summing up, principal component analysis in side-channel analysis is:

- Useful for template attacks i.e. interesting points selection
- Used for new distinguishers (variance dependency)
- Reducing the dimensionality of data
- Learning about leakage model

IV. Support Vector Machines

Support vector machines (SVM) are a group of supervised learning methods, a kernel-based technique that can be applied to classification or regression. The aim is to perform binary classification with high generalization ability and it does it by separating space with a hyperplane. If such hyperplane doesn't exist it maps data to higher dimension, where constructs another hyperplane.

A. Binary classification

There are two ways of hyperplane separation. First of all, it can be based on maximum margin. Margin is a distance in-between two investigated groups of two different classes. Classes are separated using support vectors(Fig. 3) and the goal is to find such vectors that separate these classes in a way that will maximize the margin.

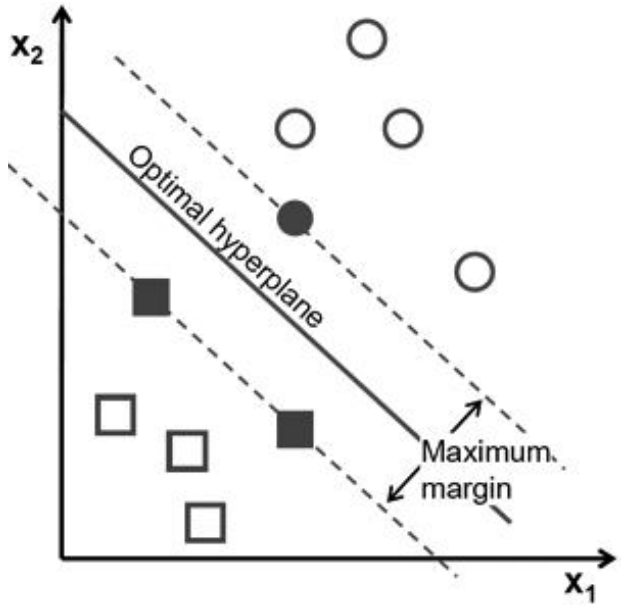

Fig. 3. Support Vector Machine concept [21]

Namely, support vector machines are trying to solve quadratic optimization problem. The reason of such maximization is that if there exists infinite number of hyperplanes that allow separation, then the one that maximizes the margin, will minimize generalization error.

In case of linearly separable data the previous way works fine, but when it comes to non-linear data another technique is needed. Non-linearity is overcome by mapping data to higher dimensions, where data is linearly separable and hyperplane can be built.

Further, it should be noted that using PCA as a preprocessing step must be performed cautiously as it can imperil the performance of SVM. Another serious problem of SVM is overfitting. It can be identified by comparing results of classification of training mode and the classification itself, which should not differ greatly. Several tricks are used to reduce overfitting, such as margin maximizing and careful choice of kernel function.

B. Multi-class classification

Essentially Support Vector Machine is a binary classification method and research on how to effectively extend it to multi-class is still going on. Nevertheless, it can be used for multi-class classification via combination of several binary SVMs. Despite the fact there are other original approaches created, e.g. [9], two general solutions to this problem can be highlighted:

- In *one-versus-all* approach k-class classifiers are used and set of binary classifiers $f^1, f^2, ..., f^k$ is constructed, where each trained to separate one class from rest. This is the earliest used implementation SVM multi-class classification, but according to [22] is less competitive than the following.
- *One-versus-one* approach needs only $\frac{k(k-1)}{2}$ in case of k-class classification. Used binary SVMs are trained to distinguish between each pair of classes, so each SVM votes for one class. Result is based on the sum of the votes, so the class with maximum number of votes wins.

1) Kernels: SVMs, described in previous section, are useful to classify linearly separable data. However, in some scenarios the data might not be linearly separable. In order to defeat this problem SVM are combined with kernel functions. As it was said, non-linear data can be mapped into a higher dimension, where feature vectors will be linearly separated. Thus, the computation of the inner product can be extended by a non-linear mapping function $\phi(\cdot)$ through $(x_i, x_j) \rightarrow (\phi(x_i), \phi(x_j))$. The exact mapping $\phi(\cdot)$ does not need to be known, which is denoted as the kernel trick, since it is implicitly defined by the kernel function.

Example of such kernel can be Radial Basis Function, which computes dot-product between points of feature vectors. If a Radial Basis Function (RBF) kernel is used, for example, and the scale factor (kernel parameter) is set to a very small value, the SVM will tend towards a linear classifier. In case of a high value, the output of the classifier will be very sensitive to small changes in the input, which means that even with margin maximisation, over-fitting is likely to be present.

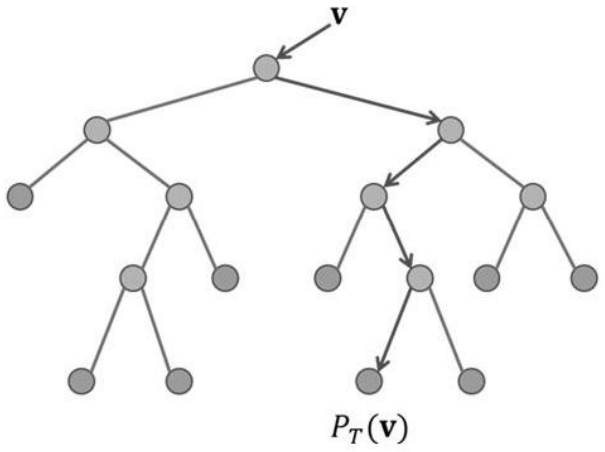

Fig. 4. Binary decision tree

V. Random Forest

A. Decision tree

For illustration of a decision tree, also called classification tree, lets introduce an example. The task is to perform classification of a group of balls. Two sets of features are colour and ball size, and it's defined that this features are interdependent. Let suppose we know this dependency: black balls are 1 cm in diameter, white are 5 cm in diameter and white ones are 10 cm. Then, a system with 3 holes is made. It represents selection function. In the first hole can fall all the balls, whose size is is exactly 1. Size of the second is enough to let balls less then 5 cm pass. The last hole suits for all sizes. By putting a ball in a hole, we can deduce its color. For that we are going to construct a decision tree.

We have rules of a type "If an object this feature, then we categorize it as this group". So if a ball doesn't fit in the first hole, we categorize it as "colour not black", then doesn't fit in the second hole and colour is not white an we get to the leaf with the answer "the ball colour is red" Arrangement of a group of such rules we get an hierarchy Fig. 4

This hierarchy forms a decision tree, where each leaf determines one category. Construction is iterative, starts with the set of observation on the target device and number of classes every leaf should have. First of all, at the root we divide all the data in homogeneous sets (homogeneity is assumed for simplicity). At each node the set is divided again on the basis of some feature. For more details on the best splitting and splitting principles please refer to [23].

On classification input starts from the top and proceed down the tree until a leaf is reached. Leafs represent answers to classification, for example, a class label corresponding to the target.

B. Random forest

Decision tree is a "weak learner", similar to SVM it is prone to overfitting, but it can be improved using ensembles. It is a divide-and-conquer approach to combine a group of "weak learners" to form together a "strong learner".

Algorithm works as follows. At first step, set is randomly divided at N cases, which are going to be separate decision tree. At each node m features are selected at random and the one that provides the best split (according to some objective function) is chosen. And the process continues iteratively for each node. It worth mentioning basis principle of a random forest that makes them resistant to overfitting:

- *bagging* improves unstable procedures by combining classifications of randomly generated training sets, can be paraphrased as averaging. On the Fig. 5 dark line represents "strong learner", whereas pale lines are separated "weak learners"

The main parameter of Random Forest is number of trees, other important parameter is number of features. While tree depth is dependent on the selected features, number of trees is more interesting object of study and needs more research as after certain point of forest growth results stop getting significantly better. For example, in case of side-channel attacks existing works used size of 500 on the basis of sufficiency for experiment's goals [25]. Empirically, optimal size of forest equals to $sqrt(features)$ for classification tasks [26].

VI. Practical results

In this section intermediary results of machine learning methods applicability are shown. As it was stated, power traces were obtained from DPA contest website [4] and TeSCASE

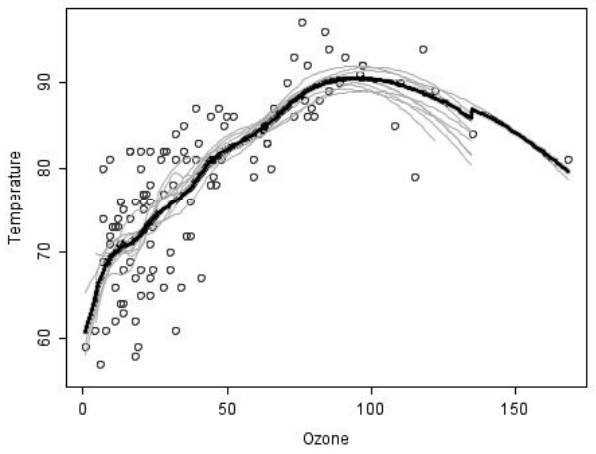

Fig. 5. Graphic example of bagging [24]

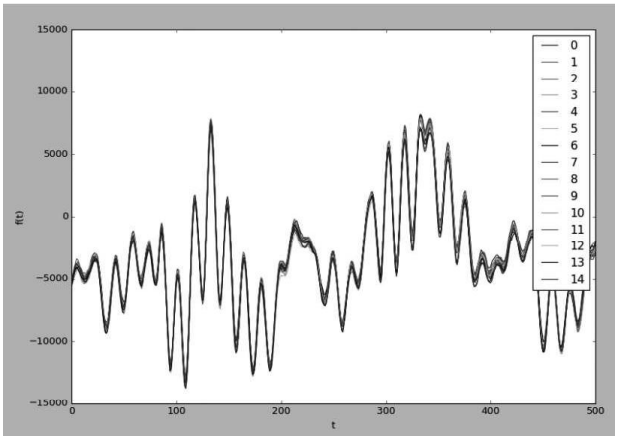

Fig. 6. Power trace of DPA contest v2

website [5]. The power traces were acquired using SASEBO GII board at a sample rate pf f_s = 5 GHz and about one round of AES is shown on the Fig. 6. One trace contains 3252 points.

The main emphasis of our work is made on classification task rather then on pre-processing. At first, no pre-processing was used, but problems connected with dimensionality and the need of selecting specific points of interest lead us to use Principal Component Analysis to filter 50 points out of traces. Another improvement used on the power traces was normalization. It was done by reducing each point to normal distribution with mean value of zero and variance equal to 1. On the Fig. 7 results are presented, for better visibility they're scaled to 500 points. It can be seen that difference between traces on the range (300-400) became more visible even with the naked eye.

In order to minimize data set and by that reduce dimensionality and computational complexity Principal Component Analysis was used and results are presented on the Fig. 8. It was as well implemented using scikit `decomposition.PCA` in Python with `n_components set to 50. Selected value is based on a scree test.`

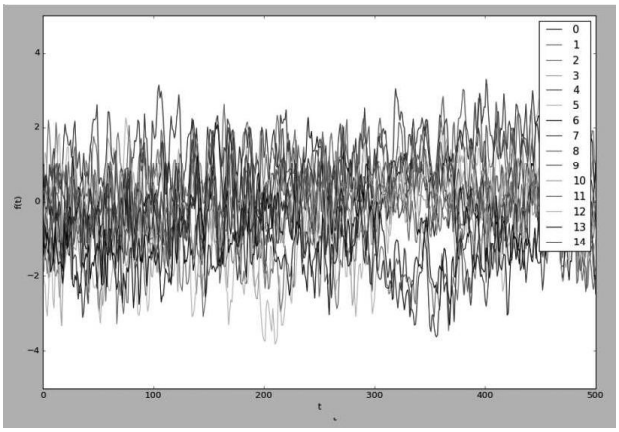

Fig. 7. Normalization of the power traces of DPA Contest v2

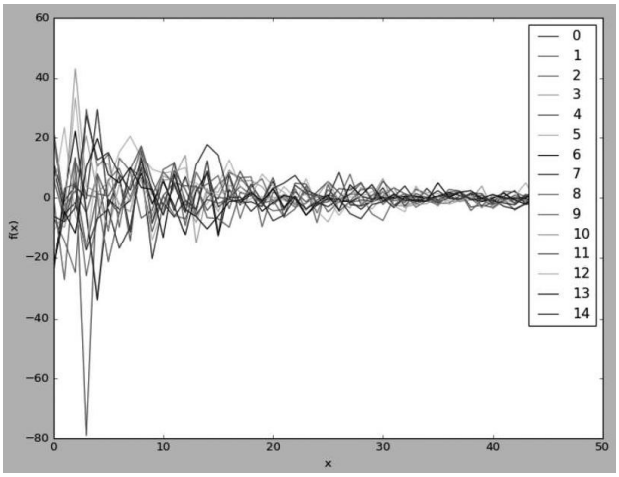

Fig. 8. Traces after applying Principal Component Analysis

Classification program was written in Python with the usage of scikit-learn library. More precisely, scikit classifiers `DecisionTreeClassifier` and `SVR` were used. General approach included bitwise and bytewise key classification. Parameter of the Decision Tree Classifier called min_samples_split determines the depth of a tree. For Support Vector Machines tuned parameter was kernel of two types: linear and based on Radial Basis Function.

TABLE I. CURRENT RESULTS

Algorithm	Key analysis	Recover results	Parameters
Decision Tree Classifier	bytewise	0.01	min_samples_split=30
Decision Tree Classifier	bytewise	0.005	min_samples_split=20
Decision Tree Classifier	bytewise	0.025	min_samples_split=60
SVM	bytewise	0.0025	kernel=rbf
SVM	bytewise	0.0025	kernel=linear
Decision Tree Classifier	bitwise	0.0042	min_samples_split=30
Decision Tree Classifier	bitwise	0.0048	min_samples_split=70
SVM	bitwise	0.0052	kernel=rbf
SVM	bitwise	0.0042	kernel=linear

Bytewise classification is regarded as first byte classification. Algorithm that were under investigation include Decision Tree Classifier and Support Vector Machine with different start characteristics. Influence of the algorithm parameters is to be investigated, as well, as impact of pre-processing, while current

results are presented in the Table I.

VII. Conclusion

Side-channels attacks are one of the most powerful kind of attack on cryptographic systems. The implementations can subject to side-channel attacks no matter how secure the algorithms are in theory. A lot of standards, created by governments and private companies, such as FIPS (Federal Information Processing Standard), CC (Common Criteria), and EMV (Europay, MasterCard, and VISA) include requirements for security compliance of products to various levels of countermeasure against side-channel attacks.

Our research concerned structure of any side-channel analysis and identified most interesting points for machine learning application and limitations and weaknesses of statistical tools. Then our focus moved on the application of machine learning in side channel data analysis and we proposed our approach based on SVM classifier and Decision Tree Classifier which is to be moreover extended to Random Forest method. For SVM classifier two most popular kernel function are investigated and compared.

For further research extension of the machine learning algorithm for classification as well as precise tuning of the parameters is left. Accurately constructed machine learning approaches can surpass existing statistical methods and provide more efficient apparatus for side-channel attacks.

References

[1] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *CRYPTO*, ser. Lecture Notes in Computer Science, vol. 1109. Springer, 1996, pp. 104–113.

[2] R. L. Rivest, "Cryptography and Machine Learning," in *Proceedings of the 1st International Conference on the Theory and Application of Cryptology*, vol. 739, 1993, pp. 427 – 439.

[3] SAKURA Hardware Security Project official website. Web: http://satoh.cs.uec.ac.jp/SAKURA/hardware.html.

[4] DPA Contest official website. Web: http://www.dpacontest.org/home/.

[5] TeSCASE – Testbed for Side Channel Analysis and Security Evaluation. Web: http://tescase.coe.neu.edu/?current_page=POWER_TRACE_LINK.

[6] B. Gierlichs, K. Lemke-Rust, and C. Paar, *Cryptographic Hardware and Embedded Systems - CHES 2006: 8th International Workshop, Yokohama, Japan, October 10-13, 2006. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, ch. Templates vs. Stochastic Methods, pp. 15–29.

[7] S. Bhasin, J.-L. Danger, S. Guilley, and Z. Najm, "Side-channel leakage and trace compression using normalized inter-class variance," in *Proceedings of the Third Workshop on Hardware and Architectural Support for Security and Privacy*, ser. HASP '14. New York, NY, USA: ACM, 2014, pp. 7:1–7:9.

[8] P.-A. Fouque, G. Leurent, D. Réal, and F. Valette, *Cryptographic Hardware and Embedded Systems - CHES 2009: 11th International Workshop Lausanne, Switzerland, September 6-9, 2009 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ch. Practical Electromagnetic Template Attack on HMAC, pp. 66–80.

[9] T. Bartkewitz and K. Lemke-Rust, "Efficient template attacks based on probabilistic multi-class support vector machines," in *Proceedings of the 11th International Conference on Smart Card Research and Advanced Applications*, ser. CARDIS'12. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 263–276.

[10] M. Backes, M. Dürmuth, S. Gerling, M. Pinkal, and C. Sporleder, "Acoustic side-channel attacks on printers," in *USENIX Security Symposium*. USENIX Association, 2010, pp. 307–322.

[11] G. Hospodar, B. Gierlichs, E. De Mulder, I. Verbauwhede, and J. Vandewalle, "Machine learning in side-channel analysis: a first study," *Journal of Cryptographic Engineering*, vol. 1, no. 4, pp. 293–302, 2011.

[12] J.-W. Chou, M.-H. Chu, Y.-L. Tsai, Y. Jin, C.-M. Cheng, and S.-D. Lin, *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. An Unsupervised Learning Model to Perform Side Channel Attack, pp. 414–425.

[13] Z. Zeng, D. Gu, J. Liu, and Z. Guo, "An improved side-channel attack based on support vector machine," in *CIS*. IEEE Computer Society, 2014, pp. 676–680.

[14] H. He, J. Jaffe, and L. Zou, "Side channel cryptanalysis using machine learning," 2012.

[15] H. Patel and R. O. Baldwin, "Random forest profiling attack on advanced encryption standard," *Int. J. Appl. Cryptol.*, vol. 3, no. 2, pp. 181–194, Jun. 2014.

[16] L. Lerman, G. Bontempi, and O. Markowitch, "Side Channel Attack: an Approach Based on Machine Learning," in *Second International Workshop on Constructive SideChannel Analysis and Secure Design*. Center for Advanced Security Research Darmstadt, 2011, pp. 29–41.

[17] L. Lerman, S. F. Medeiros, G. Bontempi, and O. Markowitch, *Smart Card Research and Advanced Applications: 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*. Cham: Springer International Publishing, 2014, ch. A Machine Learning Approach Against a Masked AES, pp. 61–75.

[18] S. Chari, J. R. Rao, and P. Rohatgi, *Cryptographic Hardware and Embedded Systems - CHES 2002: 4th International Workshop Redwood Shores, CA, USA, August 13–15, 2002 Revised Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, ch. Template Attacks, pp. 13–28.

[19] Y. Souissi, M. Nassar, S. Guilley, J.-L. Danger, and F. Flament, *Information Security and Cryptology - ICISC 2010: 13th International Conference, Seoul, Korea, December 1-3, 2010, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, ch. First Principal Components Analysis: A New Side Channel Distinguisher, pp. 407–419.

[20] J. Hogenboom and L. Batina, "Principal component analysis and side-channel attacks-master thesis," Master's thesis, 2010.

[21] Introduction to Support Vector Machines.

[22] C. Hsu and C. Lin, "A comparison of methods for multiclass support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, no. 2, pp. 415–425, 2002.

[23] *Introduction to Data Mining*. Addison-Wesley Longman Publishing Co., Inc., 2006, ch. Classification: Basic Concepts, Decision Trees, and Model Evaluation.

[24] Plot of ozone data with loess smoothers from bootstrap samples. Web: https://en.wikipedia.org/wiki/File:Ozone.png.

[25] L. Lerman, R. Poussier, G. Bontempi, O. Markowitch, and F.-X. Standaert, *Constructive Side-Channel Analysis and Secure Design: 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers*. Cham: Springer International Publishing, 2015, ch. Template Attacks vs. Machine Learning Revisited (and the Curse of Dimensionality in Side-Channel Analysis), pp. 20–33.

[26] scikit-learn. Machine Learning in Python. Ensemble methods.

[27] A. Heuser, M. Kasper, W. Schindler, and M. Stöttinger, *Topics in Cryptology – CT-RSA 2012: The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 – March 2, 2012. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. A New Difference Method for Side-Channel Analysis with High-Dimensional Leakage Models, pp. 365–382.