Convolutional neural network based side-channel attacks

Tran Ngoc Quy, Nguyen Thanh Tung, Do Quang Trung, Dang Hung Viet

***Abstract—The* profiled attack is considered one of the most effective side-channel attacks (SCA) methods used to reveal the secret key and evaluate the security of the cryptographic devices. By considering a classification problem, profiled SCA can be successfully conducted by machine learning techniques, as shown by recent works. However, these studies only provide general principles of the attack. Therefore, this paper presents technical aspects and specific instructions for an attacker when performing a profiled attack on a specific cryptographic device using a popular deep learning technique called convolution neural network. The experimental process and the results of the attack on AES-128 are presented to prove the effectiveness of the attack procedure.**

***Tóm tắt*—Trong các phương pháp tấn công kênh kề, tấn công mẫu được xem là một trong các phương pháp hiệu quả được sử dụng để tìm khóa bí mật và đánh giá độ an toàn của thiết bị mật mã. Bài toán tấn công mẫu có điểm tương đồng với bài toán phân lớp sử dụng các kỹ thuật học máy, học sâu. Các nghiên cứu về tấn công mẫu gần đây chỉ ra rằng có thể áp dụng thành công kỹ thuật học sâu khác nhau vào quy trình của cuộc tấn công mẫu. Tuy nhiên các nghiên cứu này chỉ đưa ra nguyên lý chung của tấn công. Do đó, bài báo này đề xuất một quy trình tấn công cụ thể bao gồm các khía cạnh kỹ thuật, các chỉ dẫn cụ thể cho người tấn công khi thực hiện cuộc tấn công mẫu trên thiết bị mật mã cụ thể sử dụng một kỹ thuật học sâu phổ biến là mạng nơ-ron tích chập. Quá trình thực nghiệm và kết quả tấn công trên AES-128 cũng được trình bày để chứng minh tính hiệu dụng của quy trình tấn công đề xuất.**

Keywords—Side-channel attack, Profiled attack, machine learning.

Từ khóa—tấn công kênh kề, các cuộc tấn công phân tích, học sâu.

I. INTRODUCTION

Side-channel attacks (SCA) is a powerful cryptanalytic technique that exploits the information leaked from the physical implementations of cryptographic algorithms to break the secret key [1]. SCA can be classified into two main types: non-profiled attacks such as Differential Power Analysis (DPA) [1], Correlation Power Analysis (CPA) [2] and profiled side-channel attack. Profiled attacks play an important role in the security evaluation of cryptographic implementations [3]. Indeed, they provide a security assessment assuming the worst-case scenario. The profiled SCA attacks based on supervised learning techniques have recently received significant attention in the SCA community. Researchers in the security field explore different machine learning techniques to assess their effectiveness in the SCA context. As a consequence, there are several papers on the intersection of machine learning techniques and profiled SCA attacks [4] [5]. While different scenarios usually require different machine learning techniques, almost all work demonstrates that Support Vector Machines (SVM) and Random Forests (RF) are good baseline algorithms for profiled SCA attacks.

Although machine learning-based profiled attacks relax the need for probability distributions of side-channel leakage samples, they still require specific extraction techniques to identify points of interest (POIs) on the trace. For unprotected devices, finding POIs is quite easy based on methods such as signal-to-noise ratios (SNR), the sum of squared differences (SOSD), and correlation power analysis (CPA) [4] [3] [5]. However, for protected devices, determining POIs is a challenge for SCAs [6] [7]. So far, no effective method has been proposed for selecting POIs for such devices. Fortunately, the deep learning method can solve the problem of modelling without extracting specific features in the pre-processing phase of traces [8] [6] [7]. Therefore, in recent years, deep learning has begun to demonstrate its powerful efficiency in profiled SCA attacks because it almost perfectly approximates arbitrary functions.

Several studies have already investigated the performance of deep neural networks in profiled

SCA attacks. Maghrebi et al [8] first compared the SCA-efficiency of deep learning and machine learning in terms of the number of side-channel traces. The work by Cagli et al. [9] evaluates the performance of convolutional neural networks (CNNs) in scenarios where power consumption traces are misaligned due to countermeasures or hardware-related effects. Their research shows that CNNs combined with data augmentation techniques can effectively suppress those misalignment effects. Prouff et al. [6] give an empirical solution to the problem of choosing hyper-parameters for CNNs and multi-layer perceptrons (MLP), and further established the power of applying deep learning to profiled SCA attacks. The other important contribution is the release of the public ASCAD dataset, which provides side-channel traces of a masked 128-bit AES implementation. The ASCAD dataset makes it easy for researchers to improve existing models or compare new deep neural network architectures. Zaid et al. [7] highlight the importance of configuring the hyperparameters and architecture; without proper configuration, the models do not perform well. They state that when we do not comprehend the influence of a hyperparameter we cannot realize the maximum potential of deep learning architectures.

However, the above researches only describe the theoretical aspects of the attack without providing the details of the attack process from training CNN to finding the secret key of the device under attack. Therefore, in this article, we show the comprehensive aspects of attacks, specific instructions to execute the profiled attack using CNN for an attacker when conducting the attack and evaluate the security of specific devices.

The paper is structured as follows: Part 2 introduces the basics of profiled attacks and deep learning. Part 3 presents the method of profiled attacks using a convolutional neural network. Experiments and experimental results are presented in Part 4. The conclusions of the paper are presented in Part 5.

II. BACKGROUND

A. Profiled side-channel attacks

For profiled SCA attacks, the adversary is assumed to have a pair of identical devices: a profiling device and a target device. In the attack scenario of our paper, the target device runs a symmetric cryptographic algorithm with a fixed secret key. The attacker has access to control the input and the key of the profiling device, so he can characterize the leaked information very precisely by applying statistical techniques. The profiled SCA attacks are performed in two phases: the profiling phase and the attack phase.

In the profiling phase, a dataset of N_p profiling traces is acquired on the profiled device. It will be seen as a realization of the random variable $S_p \triangleq \left\{(x_1, z_1), \ldots, \left(x_{N_p}, z_{N_p}\right)\right\} \backsim Pr[\boldsymbol{X}|Z]^{N_p}$, where all the x_i are traces corresponding to the intermediate value $z_i = \varphi(P, K)$ processing by device. Based on S_p, a model is built to characterize the side-channel leakage of the cryptographic device for each hypothetical value z_i. This can be modelled as $F(X|Z) : \mathcal{X} \rightarrow P(\mathcal{Z})$.

In the attack phase, a dataset of N_a attack traces is acquired on the target device. It will be seen as a realization of $S_a \triangleq \left(k^*, \left\{(x_1, p_1), \ldots, \left(x_{N_a}, p_{N_a}\right)\right\}\right)$ such that $k^* \in \mathcal{K}$, and for all $i \in [1, N_a], p_i \sim Pr[P]$ and $x_i \sim Pr$ $[\boldsymbol{X}|Z = \varphi(p_i, k^*)]$. After that, a prediction vector is computed for each attack trace, based on a previously built model: $y_i = F(x_i), \forall i \in [1, N_a]$. A score, for example, the probability, is assigned to each trace for each intermediate value hypothesis $z_j, with\ j \in [1, |Z|]$. The j-value of y_i describes the probability of z_j according to the model when the attack trace is x_i. These scores are combined over all the attack traces to output a likelihood for each key hypothesis and the candidate with the highest likelihood is predicted to be the right key. The maximum likelihood score can be used for predicting. For every key hypothesis $k \in \mathcal{K}$, this likelihood score is defined by equation (1) and the key with the highest score is the most likely prediction.

$$d_{S_a}[k] \triangleq \prod_{i=1}^{N_a} y_i[z_i] \;\; where\; z_i = \varphi(p_i, k) \qquad (1)$$

Template attack (TA) is a typical profiled attack that assumes that $F(X|Z)$ follows a Gaussian distribution for each target value Z_i:

$$(2\pi)^{-\frac{N}{2}}|\boldsymbol{C}|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{C}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right)$$

where $\boldsymbol{x}$ represents a N-dimensional vector, $\boldsymbol{m}$ is the mean vector, $\boldsymbol{C}$ is the covariance matrix which is called templates. For TA, the attacker builds different templates for different classes, which corresponds to different intermediate values of Z in the learning phase. In the attacking phase, the attacker uses the maximum likelihood estimation (1) for the key recovery process.

B. Deep Learning

Deep learning is a branch of machine learning that has been applied to image classification, speech recognition, and other fields [14]. Machine learning usually requires manual feature engineering while CNNs learn the automatic features directly from raw data. Furthermore, the features extracted by convolutional layers are independent of their position in the data, and dense layers can identify the features related to the labeled traces. Therefore, convolutional neural networks should be robust to jitter effects from unstable clock domains or even desynchronization [9]. The common architecture of CNNs consists of two parts, namely, feature extraction and classification. The main block of a CNN is a convolution layer (CONV) directly followed by an activation layer (ACT). The former locally extracts information from the input thanks to filters and the latter increases the complexity of the learned classification function through its non-linearity. After the activation, batch normalization (BN) is used to train deep neural networks to be faster and more stable. After some (CONV ◦ ACT◦ BN) blocks, a pooling layer (POOL) is usually added to reduce the number of neurons. This block is repeated in the neural network until an output of a reasonable size is achieved. Then, some fully connected (FC) layers are introduced to obtain a global result that depends on the entire input. The last layer of CNN is the output layer with the number of neurons equal to the number of classes to be distinguished and the activation function is *softmax*. To sum up, a common convolutional network can be characterized by the following formula:

$$IN\ o\ [CONV\ o\ ACT\ o\ BN\ o\ POOL]^{n_1}\ o\ [FC\ o\ ACT]^{n_2}\ o\ FC\ o\ Softmax$$

where n_1 and n_2 are the number of convolution and fully connected layers.

III. PROFILED ATTACKS BASED CNN

A. Attack procedure

The application of deep learning requires carefully analyzing the problem and configuring the neural network. The network for performing SCA attacks on cryptographic devices requires at least one section for performing the function of detecting and learning the features of traces and one section for performing the classification. Of the deep learning network architectures, the convolutional neural network CNN satisfies these purposes effectively. In CNN networks, the convolution layers are responsible for detecting the features of traces and the hidden neurons in the MLP network structure are responsible for classifying. Therefore, the proposed deep learning network architecture for use in profiled attacks is CNN and the general procedure of attack is shown in Figure 1.

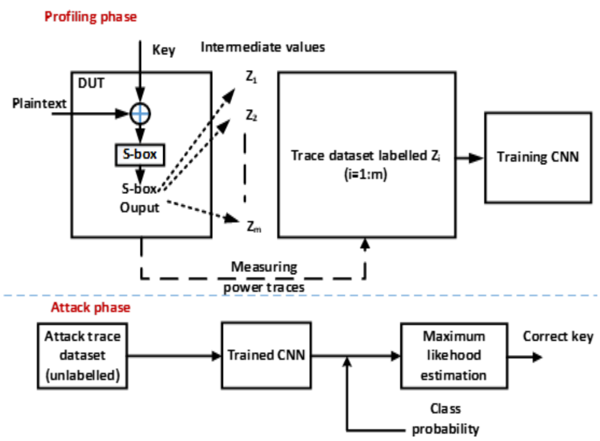

Figure 1. Profiled attacks by CNN

The profiled attack using CNN in Figure 1 proceeds through two phases: a profiling phase and an attack phase. In the profiling phase, traces collected during the operation of the cryptographic algorithm are performed on the profiling device to form a trace set. This trace set is labeled according to the intermediate value of the algorithm that needs to be profiled $Z_1, \dots, Z_m$. Usually, these intermediate values are taken at the output of the S-box. This labeled set of traces is used to train a CNN to obtain a CNN network model describing the dependency characteristic of the intermediate value Z_i on device power consumption. Specifically, in the profiling phase, the attacker does the following:

B1: First, the attacker procures a similar or identical device to the target device.

B2: He selects an intermediate attack point of the target cryptographic algorithm. For example, AES is Sbox ouput.

B3: He records several traces of the targeted operation and labels them according to $z_i = Sbox(pt_i \oplus k)$, where pt_i is the plaintext of i^{th} trace and k is the secret key of the profiling device.

B4: He selects a CNN and trains it based on the training traces set obtained in step B3. During the training, the dataset is divided into two unequally sized groups; approximately 10% of the data set is randomly selected and used as validation while the other 90% is used as a training set. Once the accuracy of the neural network is high enough, the training ends. As a result, the attacker has a trained CNN model describing the power consumption characteristics of the device at the attack point that is determined in step B2.

In the attack phase, the attacker tries to reveal the secret key from the device under attack. A N_a unlabeled traces collected from the target is classified by the trained CNN model to determine the probabilities of the traces for classes $z_1, \dots, z_m$. These class probabilities are then associated with a key byte hypothesis in order to extract the likelihood (equation (2)) for each key byte candidate and the key k_c with the highest score is the most likely prediction.

$$\log L_k = log \prod_{i=1}^{N_a} P_{CNN}(x_i|z_j) = \sum_{i=1}^{N_a} \log P_{CNN}(t_i|z_j) \tag{2}$$

where $z_j = Sbox(p_i, k); j = 1, \dots, m$ và p_i is the plaintext of trace t_i. Specifically, during the attack phase, the attacker does the following:

A1: The attacker finds a target device that is the same or similar to the profiled device.

A2: He identifies the attack point; the same attack point as in the characterization phase must be used, such as the S-Box operation.

A3: He creates an attack set by recording multiple traces of the identified operation.

A4: He applies the measured traces to the trained CNN which predicts the probability of each class. The results of this step can be found in Equation 3, where $CNN(t_i)_j$ represents the output of the CNN for trace t_i for class j and N_a the number of recorded traces, and the number of classes in the CNN is 256 (from 0 to 255).

$$\boldsymbol{D} = \begin{pmatrix} CNN(t_1)_0 & CNN(t_1)_1 & CNN(t_1)_2 & \cdots & CNN(t_1)_{255} \\ CNN(t_2)_0 & CNN(t_2)_1 & CNN(t_2)_2 & \dots & CNN(t_2)_{255} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CNN(t_{N_a})_0 & CNN(t_{N_a})_1 & CNN(t_{N_a})_2 & \cdots & CNN(t_{N_a})_{255} \end{pmatrix} \tag{3}$$

A5: Finally, he recovers the key value, one sub-key at a time, using the log-likelihood function. We explain this recovery in three steps:

- Suppose the attacker wants to recover the sub-key k while attacking an encryption algorithm. He first computes the *SBox* value of the XOR of all possible combinations of pt_i and the key value k. Let's assume that all the results are stored in a matrix P of size N_a by 256. For example, the element $p_{i,j}$, where i and j represents the row and column indices, presents the value $S(pt_i \oplus k_j)$, i.e., the *Sbox* value S of the XOR operation of the sth byte of trace i and the sub-key value $k_i = j$.

$$P = \begin{pmatrix} S(pt_1 \oplus k_0) & S(pt_1 \oplus k_1) & S(pt_1 \oplus k_2) & \cdots & S(pt_1 \oplus k_{255}) \\ S(pt_2 \oplus k_0) & S(pt_2 \oplus k_1) & S(pt_2 \oplus k_2) & \dots & S(pt_2 \oplus k_{255}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ S(pt_{N_a} \oplus k_0) & S(pt_{N_a} \oplus k_1) & S(pt_{N_a} \oplus k_2) & \cdots & S(pt_{N_a} \oplus k_{255}) \end{pmatrix} \quad (4)$$

- He replaces each value in the matrix $\boldsymbol{P}$ by $CNN(t_i)_{p_{i,j}}$ in the matrix $\boldsymbol{D}$ to get the matrix $\boldsymbol{S}$ (equation 5), where for each element the probability of a trace t_i is encrypted by the key $k_i = j$.

$$S = \begin{pmatrix} CNN(t_1)_{p_{1,0}} & CNN(t_1)_{p_{1,1}} & CNN(t_1)_{p_{1,2}} & \cdots & CNN(t_1)_{p_{1,255}} \\ CNN(t_2)_{p_{2,0}} & CNN(t_2)_{p_{2,1}} & CNN(t_2)_{p_{2,2}} & \dots & CNN(t_2)_{p_{2,255}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CNN(t_{N_a})_{p_{N_a,0}} & CNN(t_{N_a})_{p_{N_a,1}} & CNN(t_{N_a})_{p_{N_a,2}} & \cdots & CNN(t_{N_a})_{p_{N_a,255}} \end{pmatrix} \quad (5)$$

- Finally, he takes the logarithmic sum of each column of matrix S with the objective to identify the most probably key value; the index of the column with the largest sum represents the value of the sub-key.

B. CNN architectures selection for the attacks

The basic architecture of CNN consists of convolutional layers used to detect features of power consumption traces and hidden neuron layers to classify power consumption traces. The ability of CNN to classify power consumption traces is greatly influenced by main parameters such as the number of convolutional layers, the kernel size of the convolutional layer, the number of hidden layers, and their number of neurons. For each cryptographic device, these parameters should be selected appropriately to ensure that the CNN reaches the maximum classification accuracy.

For unprotected devices, according to [7], the more convolutional layers of CNN, the less confident it is in feature detection because the information on the trace is lost when it passes through the Pooling layer and the smaller of kernel size, the ability to focus on detecting the features of a trace is better. Therefore, the CNN architecture is recommended for unprotected devices consisting of one convolutional layer with 4 filters of kernel size 3, one pooling layer with the pooling size and stride is 2, one hidden layer of 10 neurons and the output layer of 256 neurons with a *softmax* activation function.

For protected devices by random delay insertion [10]*:* The protection uses random delay countermeasure as described by Coron and Kizhvatov [10]. Adding random delays to the normal operation of a cryptographic algorithm has an effect on the misalignment of important features, which in turns makes the attack more difficult to conduct. According to [7], the CNN architecture that is used to attack this kind of devices consists of 3 convolutional layers: the first layer with a small kernel size is used to detect the feature of power traces, the second layer tries to detect the value of the desynchronization due to the delay in power traces, the third block aims at reducing the dimensionality of each trace in order to focus the network on the relevant points and to remove any irrelevant ones. The details of CNN architecture are as follows: first convolution layer: number of filters 4, filter size 3, second convolution layer: number of filters 8, filter size 50, third convolution layer: number of filters 8, filter size 3, followed by 02 hidden layers with 20 neurons, and finally the output layer with 256 neurons using the *softmax* activation function.

For masking protected devices [6]: Attacks against the masking-protected devices are known as higher-order side-channel attacks, where an attacker need to combine independent feature by the operations that relate to the mask values and masked values. In order to conduct successfully profiled attacks based on CNN, the CNN network must be able to detect the features of power traces and the combination between them. According to [11], the CNN architecture that is used to attack this kind of devices consists of 2 convolutional layers: the first layer with a small kernel size is used to detect the feature of power traces and the second layer tries to generate the combination between features. The details of CNN architecture are as follows: first convolution layer: number of filters 4, filter size 3, second convolution layer: number of filters 8, filter size 51, followed by 02 hidden layers with

10 neurons, and finally the output layer with 256 neurons using the *softmax* activation function.

IV. EXPERIMENTS

In this section, we present the experimental results of implementing profiled attacks based on the CNN architectures and TA attacks for different devices. The parameters used to evaluate effectiveness are as follows:

- The ability to reveal the correct key: To confirm that our profiled attacks can reveal the correct key used by AES-128, we figure out the probability of the correct key over all keys. The key with the highest probability is the best one.

- The guessing entropy (GE) [12]: This is widely used to evaluate the effects of attacks in multi-trace experimental scenarios. When using maximum likelihood estimation to recover the secret key, we pay more attention to the final probability output of each side-channel trace. The output probability of each key candidate is ranked in descending order. The guessing entropy is then defined as the index or real key's rank within the sorted probabilities. We care about the number of traces that are required to achieve a guessing entropy of zero, that is, the number of traces required to recover the key. We estimate such a guessing entropy after 10 independent attacks.

A. Results with an unprotected device

To conduct the attack for this type of devices, we use the DPA contest v4 trace data set. The set consists of 100000 traces, each consisting of 4000 features, of a masked AES implementation. However, the traces leak first-order data and this dataset is only used as an unprotected dataset after unmasking the S-box output. The targeted sensitive variable is the output of S-box, $Sbox(p + k^*) \oplus m$, where M is the known mask. This dataset is publicly available at: http://www.dpacontest.org/v4. In the attack phase, the estimated probability of the hypothetical keys is determined by the maximum likelihood estimation. The correct key is defined as the key with the highest probability. Figure 2 shows the correct reveal key (130) having the largest probability value. The GE values obtained by the attack based CNN and TA are shown in Figure 3. The attack based CNN architecture is more effective in terms of the number of traces required for GE to reach 0. It requires only 2 traces to reach 0 while TA requires more than 7 traces. This result demonstrates that CNN can profile the characteristic of power traces more precisely than the template attack.

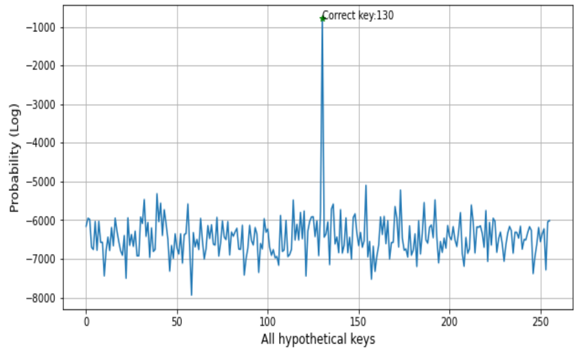

Figure 2. Estimation probability of all hypothetical keys for unprotected devices.

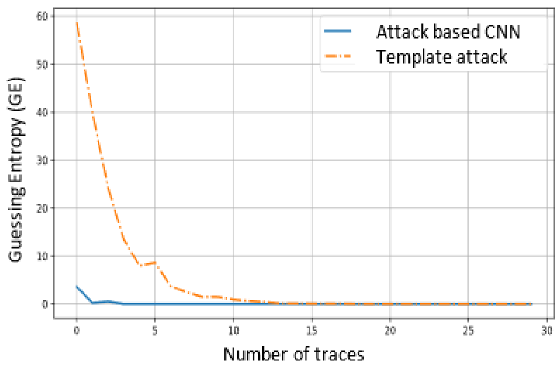

Figure 3. Guessing entropy results for unprotected devices

B. Results with an unprotected device

For random delay insertion countermeasure devices, AES-RD [10] traces data set is used. AES-RD is obtained from an 8-bit AVR microcontroller where a random delay desynchronization is implemented. For masking-protected devices, ASCAD traces data set presented in [13] is used. This data set is set up like the MNIST dataset and has 50000 profiling traces and 10000 attack traces. The traces are recorded from an 8-bit AVR microcontroller from a masked implementation of AES-128.

The attack results in Figure 4 and Figure 6 show that the profiled attack using CNN is able to recover the correct key of the protected devices. The correct keys found are 43 and 224, which have the highest decision scores among all hypothetical keys. In Figure 5 and Figure 7, comparing the attack efficiency, the template attack needs more than 500 traces, while for a profiled attack using CNN, AES-RD needs 5 traces and ASCAD need about 190 traces to rank the correct key first. The efficiency of profiled attacks using CNN is much better because the CNN network can automatically learn the hidden features in the power consumption traces, thereby classifying the traces with high accuracy. As for the template attack, the selection of trace features needs to be done manually before the attack, which makes the attack efficiency low.

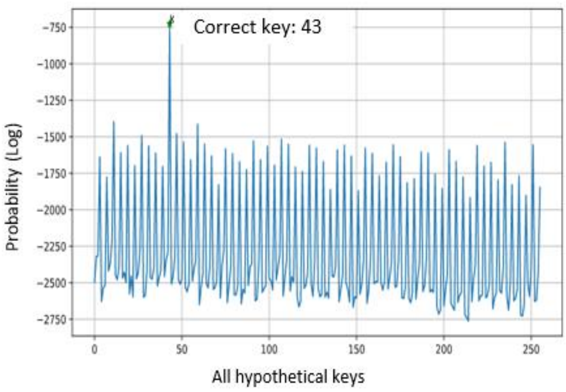

Figure 4. Estimation probability of all hypothetical keys for delay insertion-protected devices.

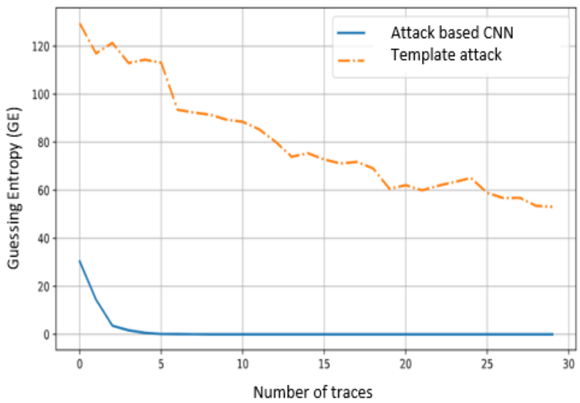

Figure 5. Guessing entropy results for random delay insertion-protected devices.

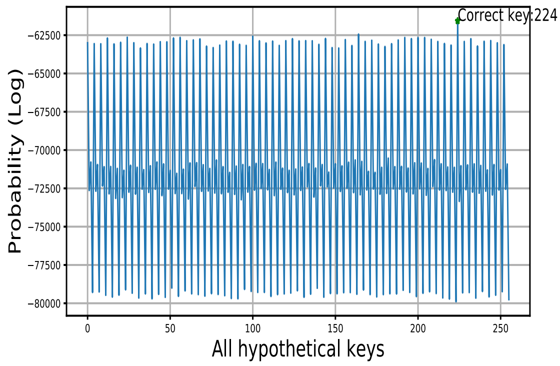

Figure 6. Estimation probability of all hypothetical keys for masking-protected devices.

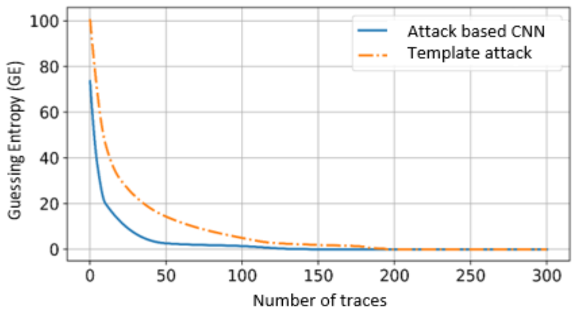

Figure 7. Guessing entropy results for masking-protected devices

V. CONCLUSION

The article presents in detail, technical aspects to conduct the profiled attacks using CNN deep learning technique. By using CNN, the attack can succeed on different cryptographic devices with better efficiency than the template attack. However, when performing attacks on different devices, the CNN architecture needs to be configured in accordance with the characteristics of the traces of each device.

REFERENCES

[1] Kocher P, Jaffe J, Jun B, "Differential Power Analysis," *CRYPTO 1999, LNCS 1666. Springer: Heidelberg*, p. 388–397, 1999.

[2] BRIER, E., CLAVIER, C., OLIVIER, F, "Correlation power analysis with a leakage model," in *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems*, Cambridge (USA), 2004.

[3] Chari S, Rao JR, Rohatgi P, "Template Attacks," *CHES 2002, LNCS 2523. Springer: Heidelberg*, pp. 13-28, 2002.

[4] A. Heuser and M. Zohner, "Intelligent Machine Homicide Breaking Cryptographic Devices Using Support Vector," in *COSADE 2012*, Heidelberg, 2012.

[5] Bartkewitz, T., Lemke-Rust, K, "Efficient template attacks based on probabilistic multi-class support vector machines," in *Mangard, S. (ed.) Smart Card Research and Advanced Applications:11th International Conference, CARDIS 2012*, Graz, Austria, 2012.

[6] Emmanuel Prouff, Remi Strullu, Ryad Benadjila, Eleonora Cagli, and Cecile Dumas, "Study of deep learning techniques for side-channel analysis and introduction to ascad database," Cryptology ePrint Archive, Report 2018/053, 2018. https://eprint.iacr.org/2018/053, 2018.

[7] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli, "Methodology for efficient cnn architectures in profiling attacks," Cryptology ePrint Archive, 2019.

[8] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff, "Breaking cryptographic implementations using deep learning techniques," in *In Claude Carlet, M. Anwar Hasan, and Vishal Saraswat, editors, Security, Privacy, and Applied Cryptography Engineering*, Springer International Publishing. ISBN 978-3-319-49445-6, 2016, pp. 3-26.

[9] Cagli E., Dumas C., Prouff E, "onvolutional Neural Networks with Data Augmentation Against Jitter-Based Countermeasures," in *Fischer W., Homma N. (eds) Cryptographic Hardware and Embedded Systems – CHES 2017*, Lecture Notes in Computer Science, vol 10529. Springer, Cham, 2017.

[10] Coron, J. ,Kizhvatov, I., "An Efficient Method for Random Delay Generation in Embedded Software," in *CHES 2009*, 2009.

[11] Tran, N.Q., Nguyen, H.Q., "Efficient cnn-based profiled side-channel attacks," *Journal of Computer Science and Cybernetics*, vol. 37, no. 1, pp. 1-22, 2021.

[12] Standaert FX., Malkin T.G., Yung M., "A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks," in *In: Joux A. (eds) Advances in Cryptology - EUROCRYPT 2009. EUROCRYPT 2009. Lecture Notes in Computer Science, vol 5479*, Springer, Berlin, Heidelberg, 2009.

[13] Benadjila, R., Prouff, E., Strullu, R., Cagli, E., Dumas, C, "Deep learning for side-channel analysis and introduction to ASCAD database," *J. Cryptographic Engineering*, vol. 10, no. 2, pp. 163-188, 2020.

ABOUT THE AUTHORS

Tran Ngoc Quy

Workplace: Academy of Cryptography Techniques

Email: quytn@actvn.edu.vn

Education: Master's degree in Electronic and Communication Techniques

Recent research direction: hardware attack, side-channel attack, IoT security

Nguyen Thanh Tung

Workplace: Academy of Cryptography Techniques

Email: tungnt@actvn.edu.vn

Education: Master's degree in Cryptographic Techniques in 2008.

Recent research direction: side-channel protection, side-channel attack.

Do Quang Trung

Workplace: Academy of Cryptography Techniques

Email: trungdq1980@actvn.edu.vn

Education: Received the BSc degree from Academy of Cryptography techniques in 2003; received the MSc degree from Academy of Cryptography techniques in 2008; received PhD. degree in Engineering Sciences in Academy of Federal Guard Service of the Russian Federation in July. 2014.

Recent research direction: Information security; Hardware security; Mathematical cryptography

Dang Hung Viet

Workplace: Academy of Cryptography Techniques

Email: vietdh@actvn.edu.vn

Education: Master's degree in Cryptographic Techniques in 2005.

Recent research direction: Hardware Design, IoT Security.

VI. APPENDIX

TABLE 1: TEST RESULTS FOR SECURE TWISTED CURVE CONDITIONS OF SOME ELLIPTIC CURVES

Curve	Parameter p, $\mathrm{ord}(E)$, $\mathrm{ord}(E') = 2p + 2 - \mathrm{ord}(E) = \prod q_i$, $q' = \max(q_i)$	$q' > 2^{200}$
NIST P-224	$p = 2^{224} - 2^{96} + 1$ $\mathrm{ord}(E) =$ 26959946667150639794667015087 01962594045780771442439172168 2722368061 $\mathrm{ord}(E') =$ 26959946667150639794667015087 01963540665802480562822456533 7410229703 = 3^2*11*47*3015283*40375823*267 983539294927*1775940414881315 83478651368420021457 $q' =$ 17759404148813158347865136842 0021457 (118 bit)	FALSE
NIST P-384	$p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$ $\mathrm{ord}(E) =$ 39402006196394479212279040100 14361380507973927046544666794 69052796276593991132635693989 56308152294913554433653942643 $\mathrm{ord}(E') =$ 39402006196394479212279040100 14361380507973927046544666794 96815288637841438804770886955 75868365581090168780292281997 q' (385 bit)= 39402006196394479212279040100 14361380507973927046544666794 96815288637841438804770886955 75868365581090168780292281997	TRUE
NIST P-521	$p = 2^{521} - 1$ $\mathrm{ord}(E) =$ 68647976601306097149819007990 81393217269435300143305409394 46345918554318339765539424505 77463332171975329639963713633 21113864768612440380340372808 892707005449 $\mathrm{ord}(E') =$ 68647976601306097149819007990 81393217269435300143305409394 46345918554318339765671000006 15349896919124216286264115983 94960379207386992907284775247 689523108855 =5*7*69697531*635884237*4425 51934538305206396094021636121 23738667546934150479241676605 86115655637650866586571630996 15599322419811318028527328629 6779187895872675299 q' (461 bit)= 44255193453830520639609402163 61212373866754693415047924167 66058611565563765086658657163 09961559932241981131802852732 8629677918789587267529 9	TRUE
NIST Curve 25519	$p = 2^{255} - 19$ $\mathrm{ord}(E) =$ 57896044618658097711785492504 34395392685693087503926084801 5607506283634007912 = 8*72370055773322622139731865 6 30429942408571163593799076060 0195093828545425098 9 $\mathrm{ord}(E') =$ 57896044618658097711785492504 34395392641305379060130319144 1976501629495631988 = 4*14474011154664524427946373 1 26085988481603263447650325797 8604941254073739079 97 $q' =$ 14474011154664524427946373126 08598848160326344765032579786 0494125407373907997 (253 bit)	TRUE
NIST Edwards 25519	$p = 2^{255} - 19$ $\mathrm{ord}(E) =$ 57896044618658097711785492504 34395392685693087503926084801 5607506283634007912 = 8*72370055773322622139731865 6 30429942408571163593799076060 0195093828545425098 9 $\mathrm{ord}(E') =$ 57896044618658097711785492504 34395392641305379060130319144 1976501629495631988 = 4*14474011154664524427946373 1 26085988481603263447650325797 8604941254073739079 97 q' (253 bit)= 14474011154664524427946373126 08598848160326344765032579786 0494125407373907997	TRUE
NIST Edwards 448	$p = 2^{448} - 2^{224} - 1$ $\mathrm{ord}(E) =$ 72683872429560689054932380788 80045343536413606873180602814 90199180584015846158342864783 02116676950385324117483636664 9219095023438599116= 4*18170968107390172263733095 1 97200113358841034017182951507 03725497951460039615395857161 95755291692375963310293709091 6623047737558596497 79 $\mathrm{ord}(E') =$ 72683872429560689054932380788 80045343536413606873180602814 90199180640640487303202508009 74623058358800693659408732062 5503011972598131764=	TRUE

	4*181709681073901722637330951 97200113358841034017182951507 03725497951601601218258006270 02436557645897001734148521830 156375752993149532941 q' (447 bit)= 18170968107390172263733095197 20011335884103401718295150703 72549795160160121825800627002 43655764589700173414852183015 6375752993149532941	
Brainpo olP256t 1	p (256 bits)= 76884956397045344220809746629 00164909303795020094305520373 5601445031516197751 $\mathrm{ord}(E)$ = 76884956397045344220809746629 00164909273753178441452953875 5519063063536359079 $\mathrm{ord}(E')$ = 76884956397045344220809746629 00164909333836861747158086871 5683826999496036425= 5^2*175939*492167257*806291530 7*2590895598527*4233394996199 *401601867518226318515439169 q' (89 bit)= 401601867518226318515439169	FALSE
Brainpo olP320t 1	p (320 bits)= 17635933222391663541619098424 46019520889512772719515192772 96041528864086880214981809550 1499903527 $\mathrm{ord}(E)$ (320 bit) = 17635933222391663541619098424 46019520889512772717686063760 68612401678478484584346835568 5258203921 $\mathrm{ord}(E')$ = 17635933222391663541619098424 46019520889512772721344321785 23470656049695275845616783531 7741603135= 5*17*157*311*4799*73360318668 1370903871980400084818445113* 12070006930118583019786361350 4123076152854324319 q' (157 bit)= 12070006930118583019786361350 4123076152854324319	FALSE
	p (512 bits)= 89489622076502325516566028151 59153422162609644098354511344 59718720005701041355243991793 43041919569427654465303864273 45937963894309923928536070534 607816947 $\mathrm{ord}(E)$ = 89489622076502325516566028151 59153422162609644098354511344 59718720005701041341852837898 17306435249598574513983700292 80583094215613882043973354392 115544169 (512 bit)	TRUE
Brainpo olP512t 1	$\mathrm{ord}(E')$ = 89489622076502325516566028151 59153422162609644098354511344 59718720005701041368635145688 68777403889256734416624028254 11292833573005965813098786677 100089727= 19*41*10529*11437*575369*1314 5081*9013120177*456050322899* 30685861077967059339366909238 10614269565824643157691128435 77315432812628738900092750881 5896120180276112393723 q' (361 bit)= 30685861077967059339366909238 10614269565824643157691128435 77315432812628738900092750881 5896120180276112393723	
M-511	$p = 2^{511} - 187$ = $\mathrm{ord}(E)$ = 67039039649712985497870124991 02923063739682910296196688861 78072186088201503685928643901 42350644440700971284740679795 91479896420070205009299687445 903538392= 8*837987995621412318723376562 38786538296746036378702458610 77225902326102518796074108048 76779383055508762141059258497 44893498705250877562616246093 0737942299 $\mathrm{ord}(E')$ = 67039039649712985497870124991 02923063739682910296196688861 78072186088201503668769036286 00631024593575929033841185064 59373857462741741560646746203 102545332= 4*167597599124282463744675312 47757307659349207275740491722 15445180465220503759171922590 71501577561483939822584602962 66148434643656854353901616865 50775636333 q' (509 bit)= 16759759912428246374467531247 75730765934920727574049172215 44518046522050375917192259071 50157756148393982258460296266 14843464365685435390161686550 775636333	TRUE
	$p = 2^{521} - 1$ = $\mathrm{ord}(E)$ = 68647976601306097149819007990 81393217269435300143305409394 46345918554318339765470190350 66066546313985467746362609365 70417277131794810169271973685 174680434092= 4*171619941503265242874547519 97703483043173588250358263523 48615864796385795849413675475	

E-521	87665166365784963669365906523 41426043192829487025423179934 21293670108523 $\mathrm{ord}(E')$ = 68647976601306097149819007990 81393217269435300143305409394 46345918554318339765740234161 26746682777114078179865220251 45656966844204623118353174371 407549680212= 4*171619941503265242874547519 97703483043173588250358263523 48615864796385795849414350585 40316866706942785195449663050 62864142417110511557795882935 92851887420053 q' (520 bit)= 17161994150326524287454751997 70348304317358825035826352348 61586479638579584941435058540 31686670694278519544966305062 86414241711051155779588293592 851887420053	TRUE
GOST R 34.10-2001/ 2012	p (256 bits)= 57896044618658097711785492504 34395392663499233282028201972 8792003956564821041 $\mathrm{ord}(E)$ = 57896044618658097711785492504 34395392708293458372545062238 0973592137631069619 $\mathrm{ord}(E')$ = 57896044618658097711785492504 34395392618705008191511341707 6610415775498572465 =3^3*5*7*19*32245081937431410 58857448761032801666732779174 70983644762331442026040092 3 q' (241 bit)= 32245081937431410588574487610 32801666732779174709836447623 314420260400923	TRUE
id-tc26-gost-3410-12-512-paramSetA	p (512 bits)= 13407807929942597099574024998 20584612747936582059239337772 35614437217640300735469768018 74298166903427690031858186486 05085375388281194656994643364 9006083527 $\mathrm{ord}(E)$ = 13407807929942597099574024998 20584612747936582059239337772 35614437217640300734492323182 90585817636498049628612556596 89950062527990641665399387547 4742293109 $\mathrm{ord}(E')$ = 13407807929942597099574024998 20584612747936582059239337772 35614437217640300736447212854 58010516170357330435103816375 20220688248571747648589899182 3269873947	TRUE
	23*61*4447*142799*20528505501 1140558581260164490369*733078 49562017923733956150217873096 92168767081465985083259778903 06974457895382675476840570830 80453035490394257 q' (365 bit)= 73307849562017923733956150217 87309692168767081465985083259 77890306974457895382675476840 57083080453035490394257	
id-tc26-gost-3410-12-512-paramSetA	p (512 bits)= 67039039649712985497870124991 02923063739682910296196688861 78072186088201503677348840093 71490834517138450159290932430 25426876941405973284973216824 503042159 $\mathrm{ord}(E)$ = 67039039649712985497870124991 02923063739682910296196688861 78072186088201503692258541985 37481903836150629109477434055 67510148398820717100282856877 776119229 $\mathrm{ord}(E')$ = 67039039649712985497870124991 02923063739682910296196688861 78072186088201503662439138202 05499765198126271209104430804 83343605483991229469663576771 229965091 =107*457*11279*288867653*811 5660434350138997*481851783220 540204336600352026802400 77*79 39453682920433906546174291390 263043*1355284088140000993562 9935782603930805279121 q' (126 bit)= 48185178322054020433660035202 680240077	FALSE
id-tc26-gost-3410-2012-256-paramSetA	p (256 bits)= 11579208923731619542357098500 86879078532699846656405640394 5758400791312963931 9 $\mathrm{ord}(E)$ = 4*2894802230932904885589274 62 52171976963338560298092253442 512153408785530358887 $\mathrm{ord}(E')$ = 4*28948022309329048855892746 2 52171976963296432034728028577 216638595171034460773 q' (254 bit)= 28948022309329048855892746252 17197696329643203472802857721 6638595171034460773	TRUE
	p (512 bits)= 13407807929942597099574024998 20584612747936582059239337772 35614437217640300735469768018 74298166903427690031858186486 05085375388281194656994643364 9006083527	

id-tc26-gost-3410-2012-512-paramSetC	$\mathrm{ord}(E)$ = 4*33519519824856492748935062495514615318698414551480983444308903609304410075183621158683000843492212744188482058508416455147171162819093459355434 64929272813 $\mathrm{ord}(E')$ = 4*33519519824856492748935062495514615318698414551480983444308903609304410075184113725326370647342304394261677232424013799121598251240639390376733 59573768951 q' (510 bit)= 33519519824856492748935062495514615318698414551480983444308903609304410075184113725326370647342304394261677232424013799121598251240639390376733 59573768951	TRUE

TABLE 2: EDWARDS TWISTED CURVES SATISFYING SECURE TWISTED STANDARDS HAVE $p = 2^{256} - 189$

$a = 1$ d = -15342 $\mathrm{ord}(E)$ = 4*28948022309329048855892746252171976963404671 4768722470835429906443591229959 57 $\mathrm{ord}(E')$ = 4*28948022309329048855892746252171976963230320 8559480349361858013595974418239 17
$a = 1$ d =15343 $\mathrm{ord}(E)$ = 4*28948022309329048855892746252171976963230320 855948034936185801359597441823917 $\mathrm{ord}(E')$ = 4*28948022309329048855892746252171976963404671 476872247083542990644359122995957
$a = 1$ d = -50993 $\mathrm{ord}(E)$ = 4*28948022309329048855892746252171976963239690 313094613445788284697140413268167 $\mathrm{ord}(E')$ = 4*28948022309329048855892746252171976963395302 019725668573940507306816151551707
$a = 1$ d =50994 $\mathrm{ord}(E)$ = 4*28948022309329048855892746252171976963395302 019725668573940507306816151551707 $\mathrm{ord}(E')$ = 4*28948022309329048855892746252171976963239690 313094613445788284697140413268167