Deep Learning-Based Side Channel Analysis

Pantea Kiaei [1] **Richa Singh** [1] **M. Caner Tol** [1]

Abstract

We investigate the use of neural network architectures originally proposed for time series and Natural Language Processing (NLP) for Side-Channel Analysis (SCA). We explore two main models; one based on Long Short Term Memory (LSTM) and the other based on Transformers. We show that these models can help an adversary to disclose the secret key of an Advanced Encryption Standard (AES) implementation and can perform better than a fully-connected network architecture. Furthermore, we observe that the loss function used during training for such attacks does not directly translate to better attack results.[1]

1. Introduction

Encryption algorithms are used to provide the confidentiality and integrity of data. Symmetric encryption algorithms, such as Data Encryption Standard (DES) (1) and Advanced Encryption Standard (AES) (2), are one type of such algorithms. In symmetric algorithms, the same key (i.e. secret key) is used by all parties to encrypt or decrypt the data. Since these algorithms are known, the only unknown variable that provides the security of encrypted data is the underlying secret key.

It has been shown that even when the encryption algorithm is mathematically secure against crypt-analysis, the byproducts of the physical implementations of such algorithms can reveal information about the secret key that can help an attacker who has physical access to the device to disclose the secret key. The physical byproduct of running an encryption algorithm is called side-channel leakage and can be in the form of power consumption of the device or the electromagnetic emanation from the device among many others.

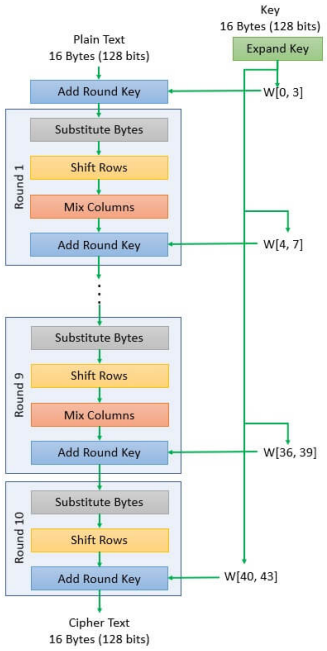


Figure 1: AES-128 encryption process

Traditionally, side-channel analysis was performed using statistical methods such as Correlation Power Analysis (CPA) (3) and Template Attacks (4). More recently researchers have started to look into how machine learning and Deep Learning (DL) methods can help in this direction. In this project, we aim to apply new DL models based on LSTMs and transformers to the existing datasets for side channel analysis and compare our results with the existing work.

2. Background

2.1. AES Cipher

AES is one of the most widely used symmetric cipher today. It is based on substitution-permutation network. AES standard that we consider here is AES-128 which has a block size of 128 bits and uses 128-bit key to encrypt 16 bytes of plaintext or decrypt 16 bytes of ciphertext. AES with 128-bit key has 10 rounds and each round consists of

[1]Worcester Polytechnic Institute. Correspondence to: Pantea Kiaei <pkiaei@wpi.edu>.

CS/DS 541 Deep Learning Final Project, Worcester Polytechnic Institute.

[1]The presentation is given on https://youtu.be/ZCRRQj4k9Ag.

four different types of layers, namely, Key addition layer, Byte substitution (S-Box) layer, Shift rows layer and Mix column layer. Further, last layer does not contain the Mix column layer. AES transformations for encryption process is shown in the Figure 1. Detailed explanation about AES cipher can be found here (5).

3. Evaluation Metrics

We will evaluate the performance of our models with two different metrics, which are Mean Rank and Minimum Traces to Key Disclosure.

3.1. Mean Rank

Mean Rank is a commonly used metric in SCA for assessing the performance of an attack. First, Maximum Likelihood for every key candidate denoted by $\vec{d}_{N_a}[k]$ is calculated using equation 1, where, N_a is the number of attack/test traces, k is every possible value of key $k \in K$, $\vec{l_i}$ is the i^{th} trace in the attack set acquired under known plaintext p_i and unknown key k^*. $\vec{d}_{N_a}$ is called scores vector and its k^{th} co-ordinate is the score corresponding to key candidate k.

$$\vec{d}_{N_a}[k] = \prod_{i=1}^{N_a} \mathrm{P}[(P, K) = (p_i, k) | \vec{L} = \vec{l_i}] \tag{1}$$

Next, using equation 2, if k^* has the highest probability or score $\vec{d}_{N_a}[k^*]$ then its rank is 0 and if it has the lowest score then its rank is $|K| - 1$. $\vec{d}_{N_a}[k^*]$ is calculated over that number of minimum attack traces N_a so that the mean rank of correct key k^* becomes zero where mean rank can be obtained by t-fold cross validation given by equation 3. In 2 and 3, $\hat{g}$ is the model function trained on dataset D_{train} and tested on dataset D_{attack}. Hence, we compute the evolution of the mean rank of correct key when the model is tested with an increasing number n of traces in D_{attack}, where, $n < N_a$.

$$\text{rank}(\hat{g}, D_{train}, D_{attack}, N_a) = |k \in K | \vec{d}_{N_a}[k] > \vec{d}_{N_a}[k^*]| \tag{2}$$

$$\text{RANK}_{N_{train}, N_a}(\hat{g}) = \mathrm{E}[\text{rank}(\hat{g}, D_{train}, D_{attack}, N_a)] \tag{3}$$

3.2. Minimum Traces to Key Disclosure (MTD)

It is a measure of attack efficiency and is defined as the minimum size N_a of the attack set which is needed to ensure that the attack succeeds in ranking the correct key as zero hence leading to its disclosure.

Table 1: Related work using deep-learning models for side-channel attack.

Paper	Dataset	DL model
(6)	ASCAD	CNN MLP LSTM
(7)	ASCAD	MLP SNN VGG16 ResNet-50 InceptionV3
(8)	AES-RD AES-HD ASCAD	VGG16 MLP
(9)	DPA Contest v4	CNN
(10)	DPA Contest v2	MLP AE CNN LSTM

4. State-of-the-art models for DL-based SCA

Table 1 enumerates the state of the art showing their DL models and employed datasets. We implement LSTM with embedding layer on the ASCAD dataset and further extend the state of the art by applying transformer-based architecture to the ASCAD dataset.

5. ASCAD Dataset

ASCAD dataset[2] includes the electromagnetic emanation measurements taken from an ATMega8515 based WB Electronics 64 Kbit ATMega chipcard while a masked AES cipher is running on it. There are two versions of ASCAD dataset, namely fixed key version and variable key version. In this project, we focus on the fixed key version. The fixed key version contains 50,000 traces for profiling and training, and 10,000 traces for the attack and testing to evaluate the performance of trained models (7). Each trace is labeled with the $SBox$ (shown as Substitute Bytes in Figure 1) output of the masked AES cipher,

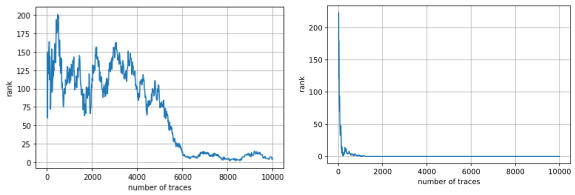
$$Y(k^*) = SBox[P_i \oplus k^*], \tag{4}$$

where P_i is the i^{th} byte of the plaintext P, and k^* is the fixed secret cryptographic key byte. Since Y is an 8-bit number, there are 256 classes in total.

6. Experiment Setup

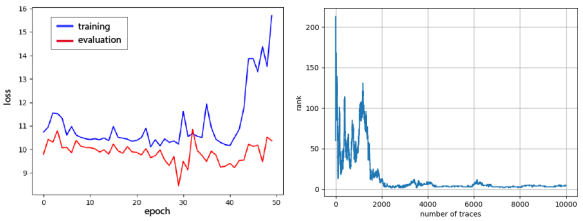
The experiment setup we use includes two NVIDIA GeForce GTX 1080 Ti and one NVIDIA GeForce Titan Xp GPUs on a local server with required libraries and drivers installed, such as, Tensorflow, PyTorch and Cuda.

[2] https://github.com/ANSSI-FR/ASCAD



(a) MLP (b) CNN

Figure 2: Attack results of reference models



(a) Loss during training (b) Attack results

Figure 3: Results for the proposed LSTM model

7. Proposed Architectures and Results

We used the reference architectures on the ASCAD data set mentioned in Section 5 to compare with our proposed architectures. For each model, we first tune the hyper-parameters and run the training and save the best trained model. In the context of SCA, this is called the profiling phase. Next, in the attack phase (testing), we use the same testing procedure for all models in which we load the saved trained model and run the attack traces to predict the intermediate value in the AES cipher (e.g. the SBox output). We then reverse engineer the AES algorithm to get the predicted secret-key byte. In the following, we describe the reference models first and then present our proposed models based on LSTM and transformer.

MLP, CNN We run the best models (MLP and CNN) proposed in (7) to compare the performance of our proposed models with them. As Figure 2(a) shows, the key rank for the correct key gets close to zero after around 6k traces using the MLP model. The same happens for the CNN model after less than 1k traces (Figure 2(b)) and the correct key rank stays stable at zero with the CNN model.

LSTM Our proposed LSTM model consists of the following layers: 1) An embedding layer, mapping 257 entries to an embedding of size 400. 2) Two vertically stacked LSTM layers with 400 units each. 3) A dropout layer with a drop probability of 0.5. 4) Three layers of fully-connected neurons; two with 4096 neurons (ending with ReLU as their activation function) and the last with 256 neurons with softmax (to predict 256 different classes).

This final model was the result of a grid search over the hyper-parameters of the number of LSTM layers ($\{1, 2\}$), usage of an embedding layer, and different configurations on the linear layers (depth and number of neurons). Furthermore, we tried different hyper-parameters for training: number of training epochs, batch size, and learning rate.

Our final model was trained with 50 epochs and a batch size of 20. We used the Adam optimizer with an initial learning rate of 1e-3 and a learning rate scheduler to reduce the learning rate when the loss function has stopped to minimize. We use 40k traces for training and 10k traces for evaluation. We run the model on the evaluation dataset every epoch and save the model only if the evaluation loss is decreased. Therefore, the final trained model, is the one with the lowest evaluation loss. Figure 3(a) shows the evolution of the loss on training and evaluation datasets (in blue and red respectively) during training. Figure 3(b) shows the attack results of our trained LSTM model. After around 2k traces, the rank of the correct key becomes close to zero therefore makes the search space of the key smaller for an attacker. The attack (test) results for the LSTM model are therefore an improvement to the reference MLP model but are not as good as the reference CNN model.

Transformer As the attention-only models have become the state-of-the-art representation on Natural Language Processing and other time series data sets, we expect that Transformer architecture can surpass the previous architectures in side channel traces as well. In this work, we have trained a classification transformer model on ASCAD dataset from scratch. Figure 4 illustrates our proposed transformer model. This model has first layer as embedding layer which requires positive integers as input. But ASCAD dataset has some values as negative, so we type-casted ASCAD dataset from INT8 to UINT8. This embedding layer converts our 2-dimensional dataset into 3-dimensional dataset using an embedding matrix of size 256×512, where, 256 is the range of values in the dataset and 512 is the embedding dimension. Thus, each feature in dataset is mapped to a 512 dimension vector. Next, positional encoding layer is applied to account for the position of the features sampled over time in a trace. Next, stack of 4 encoder layers is added with 4 attention heads. Lastly, we applied average pooling over the number of features dimension to convert the 3-dimensional data into the 2-dimensional final output sequence with 512 hidden units. At the end, a fully-connected layer with 256 neurons (256

different classes) is added and its result is softmaxed to produce probabilities.

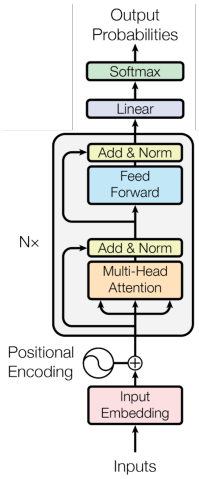


Figure 4: Proposed Transformer model architecture for sequence classification

We did grid search over the following sets for hyper parameters of the transformer model:

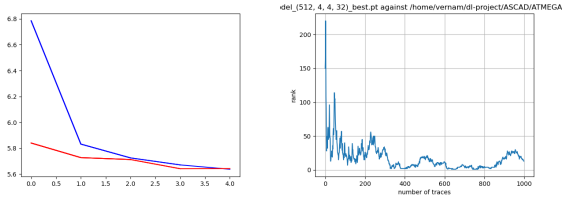
- Number of epochs: {5, 6}
- Mini-batch size: {32}
- Initial Learning rate with learning rate scheduler (ϵ): {1e-3}
- Number of encoder layers {4, 5}
- Embedding dimension {512, 640, 768, 1024}
- Number of attention heads {4, 5}

We found optimal hyper parameters on train (40k traces) + validation (10k traces) dataset as:

- Number of epochs: {5}
- Mini-batch size: {32}
- Initial Learning rate with learning rate scheduler (ϵ): {1e-3}
- Number of encoder layers {4}
- Embedding dimension {512}
- Number of attention heads {4}

Figure 5(a) shows the evolution of training and validation loss (in blue and red respectively) ran over optimal hyper parameters. It is observed that validation loss attained is less than the training loss.

Finally, we tested our trained transformer model on the test dataset or attack traces. Figure 5(b) shows correct key rank becomes close to zero after 3000 traces where rank step is 10. Our observation shows that transformer model can be leveraged in side-channel analysis.



(a) Loss during training (b) Attack results

Figure 5: Results for the proposed classification transformer model

8. Conclusion

Our work shows that state-of-the-art deep learning architectures are powerful tools to steal the secret key from an AES implementation using side channel analysis. Specifically, our proposed LSTM and Transformer models are capable of capturing the correlation between the electromagnetic emanation and SBox output of AES cipher in ASCAD dataset. Although, LSTM and Transformer models surpass the performance of the MLP model, the CNN model still achieves the best performance in ASCAD dataset. We claim that the continuous-valued nature of the electromagnetic emanation measurements is a handicap for LSTM and Transformer models since they are built upon embedding layers. Furthermore, even though the loss values during training and validation were lower for the transformer model than the LSTM implementation, the attack required fewer traces to disclose the secret key using the LSTM model. Therefore, according to our observation, lower loss function does not directly translate to better attack results.

References

[1] Vincent Rijmen and Joan Daemen. Advanced encryption standard. *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pages 19–22, 2001.

[2] Data Encryption Standard et al. Data encryption standard. *Federal Information Processing Standards Publication*, page 112, 1999.

[3] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In

International workshop on cryptographic hardware and embedded systems, pages 16–29. Springer, 2004.

[4] Suresh Chari, Josyula R Rao, and Pankaj Rohatgi. Template attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 13–28. Springer, 2002.

[5] Morris Dworkin, Elaine Barker, James Nechvatal, James Foti, Lawrence Bassham, E. Roback, and James Dray. Advanced encryption standard (aes), 2001-11-26 2001.

[6] Houssem Maghrebi. Deep learning based side channel attacks in practice. *IACR Cryptol. ePrint Arch.*, 2019:578, 2019.

[7] Ryad Benadjila, Emmanuel Prouff, Rémi Strullu, Eleonora Cagli, and Cécile Dumas. Study of deep learning techniques for side-channel analysis and introduction to ascad database. *ANSSI, France & CEA, LETI, MINATEC Campus, France. Online verfügbar unter https://eprint. iacr. org/2018/053. pdf, zuletzt geprüft am*, 22:2018, 2018.

[8] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A comprehensive study of deep learning for side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 348–375, 2020.

[9] Guilherme Perin, Baris Ege, and Jasper van Woudenberg. Lowering the bar: Deep learning for side channel analysis. 2018.

[10] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In *International Conference on Security, Privacy, and Applied Cryptography Engineering*, pages 3–26. Springer, 2016.