An Unsupervised Learning Model to Perform Side Channel Attack

Jung-Wei Chou[1], Min-Huang Chu[1], Yi-Lin Tsai[1], Yun Jin[2], Chen-Mou Cheng[2], and Shou-De Lin[1]

[1] Department of Computer Science National Taiwan University
{r99922018,r96943077,r99922099,sdlin}@csie.ntu.edu.tw
[2] Department of Electric Engineering National Taiwan University
razzzer@gmail.com, ccheng@cc.ee.ntu.edu.tw

Abstract. This paper proposes a novel unsupervised learning approach for Power Analysis – a form of side channel attack in Cryptanalysis. Different from existing works that exploit supervised learning framework to solve this problem, our method does not require any labeled pairs, which contains information of the form {X,Y}={key, power-trace}, but is still capable of deciphering the secret key accurately. Besides proposing a regression-based, unsupervised approach for this purpose, we further propose an enhanced model through exploiting the dependency of key bits between different sub-processes during the encryption process to obtain accurate results in a more efficient way. Our experiment shows that the proposed method outperforms the state-of-the-art non-learning based decipherment methods significantly.

Keywords: Power analysis, side channel attack, unsupervised learning.

1 Introduction

In cryptography, side channel attack is a kind of attacking strategy taking advantage of the information gained from physical implementation of a cryptosystem to obtain the cryptographic keys of the device. One major advantage of side channel attack lies in its non-intrusive characteristic that allows the attacker to obtain side information that facilitates the deciphering of the key. It also enjoys a much lower computational complexity than cryptanalytic-theoretical attack, most of which is of super-polynomial complexity. For well-designed ciphers, side channel attack might be the only feasible way to recover the key to the device in practice. In this work we would like to introduce a machine-learning based attacking strategy for side channel attack.

We focus on a specific type of side channel attack called power analysis, but in general the proposed technique can be applied to several other kinds of side channel attacks such as electromagnetic attacks and acoustic cryptanalysis. Power analysis is a form of side channel attack, with which the power consumption of a cryptographic device (e.g. a smart card) is analyzed to obtain the cryptographic key of the device. As some cryptographic devices are implemented using semiconductor logic gates, and current flows across the silicon substrate when change is applied to or removed from

J. Pei et al. (Eds.): PAKDD 2013, Part I, LNAI 7818, pp. 414–425, 2013.

the gate, it is not hard to imagine that through examining the power consumption of the device externally it is possible to determine what kind of operations (i.e. macro-characteristics) are executed on the device chip. One can then use such information to guess the secret key that corresponds to the hypothesized operations.

Le (2006) classified these techniques into two categories: attacks without reference devices and attacks using reference devices. With a reference device, it is possible to arrange different keys and plaintexts to feed into the device and record the output ciphertexts and power traces for further analysis. Without the reference device, while the outputs can still be measured, we have no idea what the inputs (i.e. keys) are. Hence, attacks using reference devices is like the supervised machine learning scenario, where the training data are labeled with known keys, and no (input, output) relation is provided for the other case. Therefore, attacking without reference devices is considered a much harder unsupervised learning problem.

In this paper, we propose an efficient, extensible unsupervised framework of power analysis based on machine learning techniques. We model the decipherment process as identifying a key that minimizes the training error of a given time stamp, which can be done unsupervised without using any labeled training data. Besides, the approach can be viewed as parameter estimation in abstraction, where the parameter domain contains all possible key candidates. To tackle sparse-training situation, we further propose a technique to exploit the dependency of multiple round functions in the encryption process. Finally we perform experiments on datasets obtained from the DPA Contest to show that the proposed method outperforms the competitors significantly.

The contributions of this paper are as follows. First, to the best of our knowledge, there has not yet been any work aiming at exploiting machine learning approaches to perform unsupervised side channel attack. Here we show that with careful design, simple machine learning techniques such as regression models can be exploited to tackle a cryptography problem. In this work, we hope to send an encouraging message to ML researchers on how the bridge between machine learning and cryptography can be established by demonstrating how the side-channel attack problem can be conducted from learning point of view.

2 Related Work

The concept of side channel cryptanalysis was first proposed by Kelsey (1998), which describes the use of side channel information such as current consumption leaked from imperfect implementation to facilitate breaking the cipher system. It is conceptually different from traditional cryptanalysis. That is, side channel cryptanalysis uses the correlation between plaintext and ciphertext to guess what happens inside a cryptosystem and further infer the key of the system. Side channel attack exploits the fact that most implementations of a cryptography system are not perfect and hence could inevitably leak some side channel information. The side channel information can be in the form of, for example, the electromagnetic (EM) gauged from CMOS device (Agrawal et al., 2002), or the electric current in standard block ciphers (Kocher et al., 1999) such as what has been used to attack many implementations of Data Encryption

Standard (DES) or Advanced Encryption Standard (AES). There are some other forms of power analysis, such as timing attack (Kocher, 1996), template attack (Chari et al., 2002), and acoustic attack (Backes et al., 2010).

Analyzing the snooped data is a non-intrusive attack of a cryptographic implementation, and power analysis is one of the most successful forms of such attack. The key reason to the success of power analysis lies in that the power consumption of a device generally possesses some correlation to the intermediate values that can be produced based on the cipher algorithm. In other words, maximum correlation can be obtained given correct hypothesis on the key. Below we will discuss some popular approaches based on this concept including DPA (Kocher et al., 1999), CPA (Brier et al., 2004), and BS-CPA (Komano et al., 2010).

Differential Power Analysis (DPA) is a type of attack that examines the power consumption signals through exploiting statistic measures to retrieve the correct key that has the maximum likelihood of producing the observed power consumptions. Similar to DPA, Correlation Power Analysis (CPA) is based on the linear relation between the real power consumption of the device and the intermediate values from the encipher model; it can be regarded as a form of multi-bit DPA. Messerges et al. (2002) demonstrate that CPA is just another form under DPA divided by a normalization factor. Built-In Determined Sub-Key Correlation Power Analysis (BS-CPA) is an enhancement of CPA that results in efficient trace usage. Whenever a sub-key is determined in each S-box, the BS-CPA can pass such information to assist other S-boxes to decrease the signal-to-noise ratio. In DPA Contest 2008, BS-CPA has been shown to be the most effective method. We will later compare our method with it.

In recent years, machine learning techniques have been playing an increasingly important role in attacking a cryptosystem. Hospodar (2011) proposed a supervised learning architecture to attack an AES system by side channel information. It regards the power consumption signal as an instance, divides the key bits into several binary labels and treats the problem as several binary classification tasks with Least Squares Support Vector Machine (LS-SVM) as the learner. The experiments show that LS-SVM is suitable for such purpose (Chari et al., 2002). A similar supervised approach is proposed by Lerman (2011). In practice, however, such labeled training examples are not available in most situations because it requires knowing the hidden key information in advance. Acknowledging such fact, we design an unsupervised learning approach that follows different assumptions than the previous work. In our case, only one single encrypted device with unknown key is needed..

3 Methodology

We start by interpreting the encoding process using Shannon's Noisy Channel Model. As shown in Figure 1, the inputs X to the channel contain a set of plaintext or known ciphertext (denoted as $C=\{c_1...c_n\}$) and an unknown secret key (denoted as key), while the interaction of the inputs produces the observed outcome $Y=\{y_1...y_n\}$ that reflects a sequence of measured power consumption. Note that it is possible (and generally required for side channel attack) to use a variety of cipher-texts interacting

with the same key to produce a set of observations. The noisy channel P(X|Y) can be considered as a black box that produces an output given an input. Given a fully observed Y and a partially observed X with P(X|Y) unobserved, the goal then becomes to recover the missing part of X (i.e. the secret key) using Y and C. The problem can then be mathematically formulated as argmax_x P(X|Y). We can first use Baye's rule to decompose argmax_xP(X|Y) into argmax_x[P(Y|X)*P(X)]. This essentially tells us that a proper X should not only possess a higher chance to produce the observation Y, but also has a higher chance to occur among other *X*'s. Here we assume no prior knowledge about X in a cryptography system, and consequently *P(X)* is uniformly distributed. In this problem, we are given a set of n instances as inputs X={x_k=(c_k,key), *where k=1...n*}, where c_k is a known cipher-text with the corresponding observation y_k, but the secret *key* is unknown. Assuming the deciphering processes are independent for each cipher-text, the problem argmax_xP(Y|X) can be transformed to

$$\text{argmax}_{key}[\text{P}(y_1|c_1, key)\ \text{P}(y_2|c_2, key) \ldots \text{P}(y_n|c_n, key)] \quad (1)$$

Then it becomes obvious that with a faithful estimation of P(y_k|c_k.key), one can eventually solve (1) by enumerating all possible keys. Here, a faithful estimation of P(y_k|c_k.key) implies that among all possible keys (key_1...key_m), only the correct key (denoted as key_c) shall obtain high P(y|c,key) value. Mathematically, a faithful estimation of P(y_k|c_k.key) should possess the following property:

$$\max_z[\text{P}(y_1|c_1, key_z)\ \text{P}(y_2|c_2, key_z)\ \ldots \text{P}(y_n|c_n, key_z), z \in [1, m], z \neq c] \ll \text{P}(y_1|c_1, key_c)\ \text{P}(y_2|c_2, key_c)\ \ldots \text{P}(y_n|c_n, key_c)$$

In other words, incorrect keys should possess much lower probability of producing y than the correct one. After taking log on both sides, we can obtain

$$\max_z[\log\text{P}(y_1|c_1, key_z) + \log\text{P}(y_2|c_2, key_z) + \ldots + \log\text{P}(y_n|c_n, key_z), z \in [1, m], z \neq c] \ll \log\text{P}(y_1|c_1, key_c) + \log\text{P}(y_2|c_2, key_c) + \log\text{P}(y_n|c_n, key_c)$$

The above equation is reasonable as the power signature reflects only the interaction between the correct key and the ciphertext. Therefore, we propose a three-stage framework to solve this problem:

1. Build m different learners ML_1... ML_m, each contains n instances (I_1,...I_n) that correspond to one single key:

$$\text{ML}_\text{r} = \{\text{I}_1 = (\text{c}_1, key_r, \text{y}_1), \text{I}_1 = (\text{c}_2, key_r, \text{y}_2), \ldots, \text{I}_\text{n} = (\text{c}_\text{n}, key_r, \text{y}_\text{n})\}, \text{r} \in [1, \text{m}]$$

For each input instance *(c, key)*, we generate a set of features *f(c, key)* to train the learners ML_r. The generation of such features depends heavily on the backend cryptography algorithm. An example will be demonstrated in the experiment section.

2. We propose to model log P(y_k|c_k,key) using the inverse of training errors of the learners.

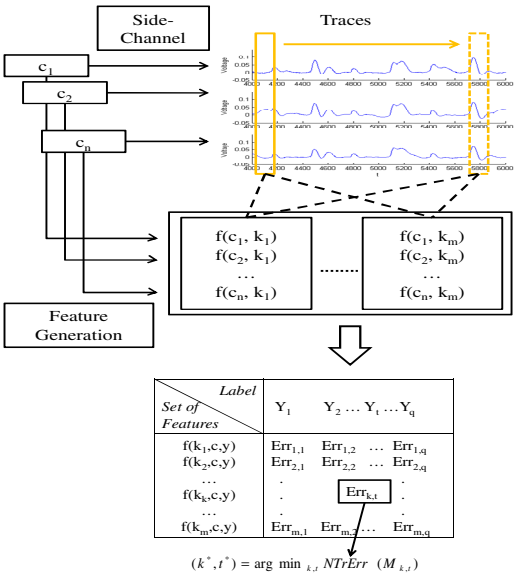

Fig. 1. The Framework

Acknowledging the fact that $P(y_k|c_k,key)$ represents how likely y_k results from an interaction between c_k and key, here we assume that the relationship between the input {c, key} and output *y* are learnable (i.e. low prediction errors) for correct key, and not learnable (high prediction errors) for incorrect keys. Therefore, the predictability of a machine learner has been used here to estimate the quality of a noisy channel. The channel corresponds to the right key should contain less noise and consequently be more learnable. Figure 1 illustrates the process. Next we assume the relation between trace sample *y* and features *x* generated from each pair of cipher-text and key candidate can be modeled simply as

$$y = w^T + n$$

where n can be assumed as Gaussian noise as in (Prouff et al., 2009), and its density function can be written as

$$P(n) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(y - w^T x)^2}{2\sigma^2}\right)$$

where σ is the standard deviation of the noise. As a result, the weight vector *w* for each timestamp and each possible key candidate can be derived by ordinary least square (OLS). Furthermore, coefficients of determination are used to normalize different scales at different time stamps.

Above we have explained how a supervised learning model can be exploited to solve such an unsupervised decipherment problem. However, for power analysis, there is still one more issue to be addressed. Although the outputs y for power

analysis can be regarded as a sampled time sequence $\{y_1...y_q\}$, truth is that for most of the features generated, only very few signal in certain particular time stamps of y reveals apparent relationships with respect to (c, key). That is, usually only the right combination of key (denoted as key_c) and time stamp (denoted as t_c) possess higher learnability than other time stamp. Therefore, instead of building m different learners, we propose to create m*q different learners (q represents the number of samples for each time sequence) and argue that the one with the best predictability does reflect the correct key and time stamp.

3.1 Sub-Key Breaker

One major concern for such key enumeration approach lies in the fact that there are exponentially many keys to try. To conquer this problem, we follow a commonly used strategy of CPA (Brier et al., 2004) to divide the key into several sub-keys according to the permutation of inputs of substitution boxes (S-boxes). S-boxes are important components in block cipher design, which perform substitution and provide nonlinearity between ciphertexts and plaintexts. For instance, the length of key is 56 bits for DES; based on such a 56-bit key, each round a 48-bit round key is derived and distributed to 8 S-boxes (see section 4.1 for details). Generally, each sub-key has certain physical meaning, and we can extract features from it given some domain knowledge. Then we can apply the method proposed in 3.1 on each sub-key independently for better efficiency.

3.2 Dual-Round Approach for Multi-round Ciphers

So far we have introduced our approach to obtain the secret key from power traces. The quality of the results depends significantly on whether there are sufficient examples (or traces) to learn from. Without sufficient training examples, by chance some incorrect key might possess high P(y|c,key) and create false positives under our framework. One practical method to determine whether there is sufficient trace is to draw the "learning curve" that indicates whether the deciphered key becomes stable. In our experiments, we consider the deciphering process as completed if the results do not change after 100 additional traces are added.

The method mentioned in 3.1 might not be as effective if the number of traces is not enough to reveal the correct key (or to eliminate the false positives). Experiments show that in multiple-round ciphers, the encryption process that affects the power signal in each round does correspond to one particular time interval. Therefore it is possible to generate multiple training instances based on information from different rounds. Remember in Section 3.1 we have described how to break a longer key into sub-keys for analysis. In general these sub-keys are organized differently in different rounds. For examples, some bits might be grouped into the same sub-key in one round and broken into several different sub-keys in another round. For each round, some sub-keys can be deciphered easily (i.e. requiring fewer traces to converge to a steady result), while others require more training examples. Here we would like to further describe how one can fully exploit the side channel information from multiple-round

ciphers given a limited number of traces. For Feistel ciphers with multiple rounds such as DES, if we are given some information of dependency between sub-keys in different rounds, a modified version of our method can exploit the relations between these sub-keys to improve the deciphering performance. The intuition behind our idea is as follows: if a correct sub-key in one round is identified (i.e. possesses lower training error than most of other candidates), then we can propagate such information to other learners and group the overlapped bit using the learned values. By doing such, the search space for other harder sub-keys is reduced, which alleviates the high demand for training traces and reduces the false positives.

An example is shown in Figure 2. It is known that some of the key bits such as b_0 and b_4 are used in both rounds, indicated by the connecting edges. Therefore, we can search for the key bits in these two rounds simultaneously. The difficulty of finding the correct key of the harder round (need more traces to break) can be reduced with the help of the easier one because of the increasing of signal-to-noise ratio when considering all cases at the same time. In dual-round approach, during training we first identify a set of S-boxes, called S-boxes set, whose bits overlapped with each other to some extent. Then within each S-boxes set, we train each of the S-boxes independently, but weighted-sum the errors of each to represent the quality of a particular assignment of bits. The weight is determined by the inverse of the "number of S-boxes" in each round containing in this particular S-boxes set. That is, if an S-box set contains one S-box in round 1 and two S-boxes in round 2, then the weight for the round 1 S-box is twice as much as that of the ones in round 2.

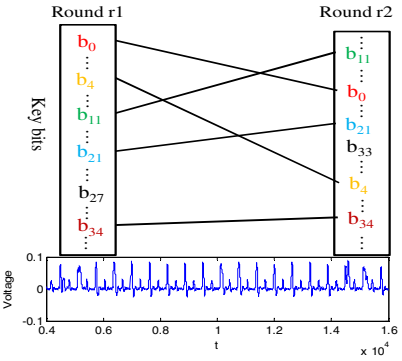

Fig. 2. Example of key dependency. The endpoints of each edge have the same key bit. In other words, the key bits used in both rounds are overlapped.

4 Experiments

We evaluate our models using the power consumption traces of the unprotected DES crypto-processor on the SecmatV1 SoC (System on Chip) in ASIC (Application-Specific Integrated Circuits), which is provided by DPA Contest 2008 and focuses on Differential Power Analysis or similar attacks. Here we select the first 1000 traces from secmatv1_2006_04_0809 for experimentation, with each trace containing 16

nominal DES runs. The raw signal of each trace looks like the one shown in Figure 3. To smooth the signal and eliminate apparent outliers, we take the average of ten original samples as one sample for every power consumption trace in our dataset. As the number of learners we need to create is proportional to the number of temporal samples, here we choose to use only 20 samples per trace for efficiency purpose. Experiments show that we can still achieve high-quality results based on only 20 learners.

This experiment tries to address two issues. First, we would like to know whether our method can accurately identify the correct key. Second, given that the correct key can be discovered, we want to compare our method with two popular non-learning based methods, CPA and BS-CPA, to see whether our method can find the correct key using fewer number of power traces (i.e. achieve the same quality using fewer data). For the second purpose, we gradually increase the training data size until the outputs become stable. Based on the rule from the DPA Contest, the attack is considered successful if the correct key appears and remains unchanged for 100 consecutive trace additions.

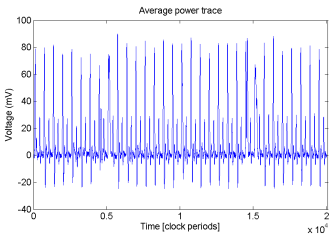

Fig. 3. A sample averaged power consumption trace, containing 20003 samples

4.1 Feature Generation

Here we first describe the encryption process of DES. In the encryption of DES, the plaintext is first permuted and divided into two 32-bit halves, L_0 and R_0. These two half-blocks are then processed by 16 identical stages called rounds. For each round, there is a 48-bit round key involved, which is a permutation of the original key and can be computed reversely. Hence, we can figure out 48 bits of the original key by revealing a round key, and then the other 8 bits can be found by exhaustive search. In every round, the right half is expanded to 48 bits and XORed with the round key. Then it is split into eight 6-bit blocks and fed to eight S-boxes, each of which has a 4-bit output and generates a 32-bit value that is then permuted again, XORed with the left half to become the new right half. For a typical implementation of the DES, the two half-blocks keep the same addresses in memory throughout the encryption. Therefore, after a round ends and before the next one begins, the register storing the right half must be replaced by a new value, which results in several bit flips and extra power consumption. By modeling the power consumption, we can derive the condition of the bit flips and eventually derive the round key. Because the eight S-boxes form a parallel structure, we can attack them one at a time. Each S-box is related to 6 bits of the round key and the four output bits are stored to some known addresses.

We take the first S-box of the first round as an example. The right half before the first round, R_0, is known and the four output bits of the target S-box can be computed by assuming a hypothetical 6-bit value as the relevant part of the round key. Therefore, we are able to find whether the bit flips for these four bits and can generate four features, each is 1 or 0, representing whether the bit flips or not.

Thus, the first feature we extract is to compute the hamming distance, which measures the corresponding bits flip between the old and new right half of register in the first or the last round of DES (Kocher et al., 1999). The second feature is to compute the hamming distance between the old and new left half of register in the first or the last round of DES (Almeida, 2008). The difference between left half and right half is that the left half inherits from previous right half directly and does not divide into 8 S-boxes. Therefore, when we attack different S-boxes in a round, each S-box has its hamming distance value. Since the output of each S-box has 4 bits, the first feature value of each S-box is between 0 and 4. On the other side, the hamming distance of left half register is always the same in a round because it inherits directly from the right half of previous round. Since there are 32 bits in the left half register, the second feature value of each round is between 0 and 32. In single-round approach, we extract those features in the last, or the 16th round, of DES. All traces and features are then normalized to zero mean and unit variance.

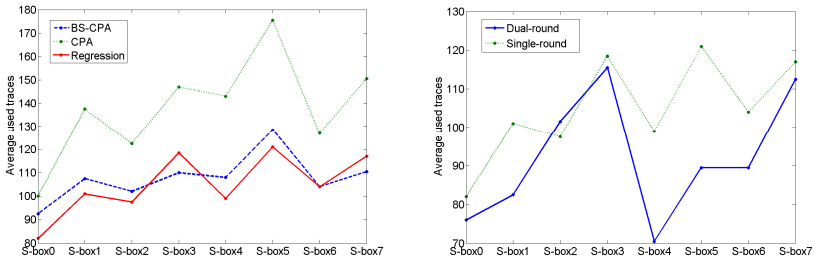

Fig. 4. On the left hand side is average traces used for each algorithm, S-box of BS-CPA, CPA, and learning-based regression methods. On the right side is the comparison of our regression-based method and the dual-round approach.

4.2 Experiment of Single-Round Approach

In order to compare the efficiency of our model, we use CPA and built-in determined sub-key CPA (BS-CPA) (Komano et al., 2010) as competitive algorithms against our model. For each competitive algorithm, we add 10 traces each time until a correct and stable key is found. As mentioned previously, we adopt the sub-key breaker to attack the sub-key (or S-box) one by one. Acknowledging the fact that the order of the traces to be added can affect the results as some training inputs are more representative, here we shuffle the order of traces each time and then average results from 20 different orders to obtain the average number of traces used for each algorithm. We depict the average traces of each method in the left hand side of Figure 4. In some cases such as S-box3 or S-box7, our method gets worse results. We believe that it is caused by the noise in the current traces or overfitting; however, our learning-based method performs better than others in most of the cases.

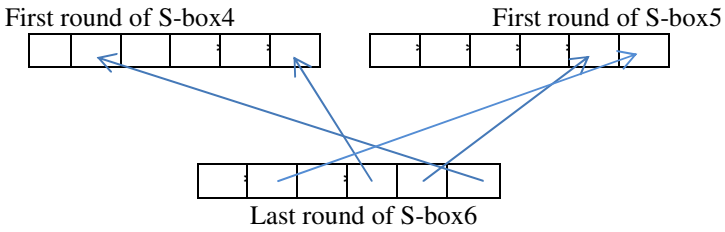

Fig. 5. Key dependency between the first round of S-box4, S-box5 and the last round of S-box6

4.3 Experiment of Dual-Round Approach

In real world scenario, sometimes there are only limited numbers of traces available. Therefore, we can resort to the dual-round approach that exploits extra multi-round information. For DES, we exploit the first and the last rounds. That is, we explore key dependency between the first and the last of nominal DES round. Once we obtain a possible key candidate in one round with high confidence, we can pass such information to the other round to reduce the search space of all possible key candidates. Before pursuing dual-round attack, we need to first observe the dependency of key bits, which can be derived from the encryption algorithm itself. For example, the bit0 and bit2 of the S-box6 in the last nominal DES round do not have corresponding key bits in the first round. The bit1, bit3, bit4 and bit5 of S-box6 in the last round have corresponding key bit positions 35, 29, 34 and 24 in the first nominal DES round respectively.

The single-round results in Figure 4 show that S-box5 is the most difficult sub-key to attack, as it requires the most traces on average. If we can use the knowledge of key dependency from another round, it is possible to reduce traces required to attack S-box5. Figure 5 shows the key dependency between the first round of S-box4, S-box5 and the last round of S-box6. We have realized that there are two overlapped keys between S-box5 in the first round and S-box6 in the last round, and another two overlapped bits between S-box6 in the last round with S-box4 in the first round. Therefore, it is possible to propagate the key bits learned in an easier S-box (e.g. S-box4) to the harder ones. We realize such idea by considering the bits in these three S-boxes altogether and use the weighted sum of the errors to evaluate the quality of certain assignment. Even though not all key bits has a corresponding mapping between the first and the last round, we still need to search all possible combinations of those non-overlapped bits. The higher the key dependency, the more likely we can use fewer traces or training examples to decipher the key.

4.4 Results of Dual Round Experiment

Here we compare the dual-round approach with the single-round approach. We focus on deciphering the S-boxes in the last nominal DES round using the dual-round attack technique because we can easily compare the results with our single-round approach. The right hand side of Fig. **4.** shows the results of regression-based single-round approach and dual-round approach. Table 1 shows the numbers of average traces

required for each S-box. Not surprisingly, the dual-round approach has better performance than single-round approach in most situations. Such results demonstrate that dual-round approach can trim the search space to avoid the interference of some potential false positives because an incorrect key needs to perform well in both rounds to be selected as false positives, which is less likely to happen comparing with single-round approach.

Table 1. The experiment results of single-round approach and dual-round approach

Method	Single-round approach	Dual-round approach
Used traces	Avg.	Avg.
S-box0	82	76
S-box1	101	82.5
S-box2	97.5	101.5
S-box3	118.5	115.5
S-box4	99	70.5
S-box5	121	89.5
S-box6	104	89.5
S-box7	117	112.5

5 Conclusion

Side channel attacks play an important role in cryptography. Despite that in theory, cryptographers can design provably-secure cryptographic algorithms, these algorithms need to be implemented and carried out by computing hardware. The implementations can subject to side channel attacks no matter how secure the algorithms are in theory. This is why many industrial and governmental standards such as FIPS (Federal Information Processing Standard), CC (Common Criteria), and EMV (Europay, MasterCard, and VISA) require that compliant security products have various levels of countermeasure against side channel attacks. It is therefore crucial to understand how efficient such attacks can be with advanced techniques from, e.g., machine learning, as well as to gain some insights into how these attacks work in order to design more effective countermeasures. In this paper, we introduced a novel unsupervised, regression-based approach to perform side-channel attack. We further extend this approach to consider information from multiple rounds with promising results. We hope this paper can serve as an encouraging example to show how machine learning approaches can be carefully crafted to solve a well-known security problem.

Acknowledgement. This work was supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC101-2911-I-002-001, NSC101-2628-E-002-028-MY2 and NTU102R7501.

References

1. Kelsey, J., Schneier, B., Wagner, D., Hall, C.: Side channel cryptanalysis of product ciphers. In: Quisquater, J.-J., Deswarte, Y., Meadows, C., Gollmann, D. (eds.) ESORICS 1998. LNCS, vol. 1485, pp. 97–110. Springer, Heidelberg (1998)
2. Agrawal, D., Archambeault, B., Rao, J.R., Rohatgi, P.: The EM side–channel(s). In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 29–45. Springer, Heidelberg (2003)
3. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
4. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
5. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining Smart-Card Security under the Threat of Power Analysis Attacks. IEEE Transactions on Computer 51(5), 541–552 (2002)
6. Bévan, R., Knudsen, E.W.: Ways to Enhance Differential Power Analysis. In: Lee, P.J., Lim, C.H. (eds.) ICISC 2002. LNCS, vol. 2587, pp. 327–342. Springer, Heidelberg (2003)
7. Le, T.-H., Clédière, J., Canovas, C., Robisson, B., Servière, C., Lacoume, J.-L.: A proposition for Correlation Power Analysis enhancement. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 174–186. Springer, Heidelberg (2006)
8. Brier, E., Clavier, C., Olivier, F.: Correlation Power Analysis with a Leakage Model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004)
9. DPA contest (2008-2009), `http://www.dpacontest.org/home/`
10. Komano, Y., Shimizu, H., Kawamura, S.: BS-CPA: Built-in Determined Sub-key Correlation Power Analysis. In: Proceedings of IEICE Transactions (2010)
11. Lerman, L., Bontempi, G., Markowitch, O.: Side-channel attack - an approach based on machine learning. In: Second International Workshop on Constructive Side Channel Analysis and Secure Design, COSAED 2011 (2011)
12. Almeida, A.: A Simple Improvement of Classical Correlation Power Analysis Attack on DES, DPA contest (2008/2009)
13. Chari, S., Rao, J.R., Rohatgi, P.: Template Attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003)
14. Backes, M., Durmuth, M., Gerling, S., Pinkal, M., Sporleder, C.: Acoustic side-channel attacks on printers. In: USENIX, p. 20. USENIX Association, USA (2010)
15. Hospodar, G., Mulder, E.D., Gierlichs, B., Verbauwhede, I., Vandewalle, J.: Least Squares Support Vector Machines for Side-Channel Analysis. In: Second International Workshop on Constructive SideChannel Analysis and Secure Design (2011)
16. Prouff, E., Rivain, M., Bevan, R.: Statistical Analysis of Second Order Differential Power Analysis. IEEE Transactions on Computers 58(6), 799–811 (2009)