Template Attacks in Principal Subspaces

C. Archambeau, E. Peeters, F.-X. Standaert, and J.-J. Quisquater

UCL Crypto Group - Université catholique de Louvain
Place du Levant 3, B-1348 Louvain-la-Neuve, Belgium
{archambeau, peeters, standaert, jjq}@dice.ucl.ac.be

Abstract. Side-channel attacks are a serious threat to implementations of cryptographic algorithms. Secret information is recovered based on power consumption, electromagnetic emanations or any other form of physical information leakage. Template attacks are probabilistic side-channel attacks, which assume a Gaussian noise model. Using the maximum likelihood principle enables us to reveal (part of) the secret for each set of recordings (i.e., leakage trace). In practice, however, the major concerns are (i) how to select the points of interest of the traces, (ii) how to choose the minimal distance between these points, and (iii) how many points of interest are needed for attacking. So far, only heuristics were provided. In this work, we propose to perform template attacks in the principal subspace of the traces. This new type of attack addresses all practical issues in principled way and automatically. The approach is validated by attacking stream ciphers such as RC4. We also report analysis results of template style attacks against an FPGA implementation of AES Rijndael. Roughly, the template attack we carried out requires five time less encrypted messages than the best reported correlation attack against similar block cipher implementations.

1 Introduction

Since their first public appearance in 1996 [6], side-channel attacks have been intensively studied by the cryptographic community. The basic principle is to monitor one (or more) unintentional channels that leak from a device such as a smart card and to match these observations with a key-dependent leakage prediction. This channel is usually monitored thanks to an oscilloscope that samples a continuous analog signal and turns it into a discrete digitalized sequence. This sequence is often referred to as a trace.

Recently, a probabilistic side-channel attack, called the Template Attack (TA), was introduced [2]. This attack was originally mounted to target stream ciphers implementation. In this context, the attacker can only observe a single use of the key, usually during the initialization step of the cipher. As it is not possible to generate different leakages from the same secret key (e.g., corresponding to different plaintexts), TAs were purposed for a more efficient way of retrieving information from side-channel traces.

There are three main reasons that make TAs more efficient than previous approaches to exploit side-channel leakages. First, TAs usually require a profiling step, in order to build a (probabilistic) noise model of the side-channel

L. Goubin and M. Matsui (Eds.): CHES 2006, LNCS 4249, pp. 1–14, 2006.

that can be used to capture the secret information leaked by a running device. Second, TAs usually exploit multivariate statistics to characterize the dependencies between the different time instant in the traces. Finally, TAs use maximum likelihood as similarity measure, that can capture any type of dependency (if the probabilistic model is found to be adequate), whereas, for example correlation analysis only captures linear dependencies [1]. In general, the cost of these improvements is a reduction of the adversarial flexibility. For example, Hamming weight leakage models can generally be used for any CMOS devices while template attacks profile the leakage function for one particular device.

TA relies on the hypothesis that leakage information is located in the variability of the leakage traces. In order to recover the secret, one has thus to focus at the time instants where the variability is maximal. However, in practice it is not clear how many and which moments exactly are important. The attacks are therefore based on heuristics, which specify these quantities according to some prior belief. For example, it is common to force the successive, relevant time instants to be one clock cycle distant.

The main contribution of this work is that we take TA a step further. Instead of applying TA directly, we first transform the leakage traces such that we are able to select the relevant features (i.e. transformed time instants) and their number automatically. Meanwhile, we do not need to determine a specific feature interdistance. Of course, when performing TA after transformation, we still take the correlations between the features into account. Now, in order to find a suitable transformation consider again ordinary TA. It is assumed that the secret information leakage is mainly hidden in the local variability of the mean traces. If this hypothesis is valid, it would be more appropriate to take the optimal linear combination of the relevant time samples and perform TA in the principal subspace of the mean traces. We call this approach principal subspace-based TA (PSTA). A principal subspace can be viewed as a lower dimensional subspace embedded in the data space[1] where each coordinate axis successively indicates the direction in which the data have maximal variability (or variance).

A standard statistical tool for finding the principal subspace of a data set is principal component analysis (PCA) [5]. PCA performs an eigendecomposition of the empirical data covariance matrix in order to identify, both, the principal directions (eigenvectors) and the variance (eigenvalues) associated to each one of them. However, practical issues may arise in the context of PSTA, as the dimension of the traces is much larger, (typically $\mathcal{O}(10^5)$) than the number of traces (typically $\mathcal{O}(10^3)$). Therefore, we propose to use a variant of PCA that is more suitable in this situation (see Section 3.1 for further details).

An attractive feature of PSTA is that the projected traces are aligned with the directions of maximal variance. These directions are nothing else than a weighted sum of all the time instants, the weights being determined such that the data variability is preserved after projection. So, in contrast to TA, which selects a relevant subset of time instants according to a heuristic, PSTA determines first the optimal (in terms of maximal variance) linear combination of these time

[1] Here, the data space is the space in which the leakage traces live.

instants. In other words, there is no need to determine an interdistance between the time samples anymore as the irrelevant ones will be assigned a small weight. Furthermore, based on the value of the eigenvalues, one can determine which (the largest) and how many directions are relevant. In order to validate our approach, we finally apply the described techniques to two implementation cases. First we target an implementation of RC4, similar to the one in [3] as a typical context where template attacks are necessary. The, we target an FPGA implementation of the AES Rijndael. For this purpose, we suggest an adaptation of template attacks that allow characterizing the leakage traces of block ciphers. We finally compare the obtained results with previously reported and observe a significant improvement of the attacks efficiency (which is, again, to be traded with less flexibility than previous attacks).

2 Template Attacks

In this section, the underlying principle of Template Attacks (TA) is first presented. Next, we introduce principal subspace TA (PSTA). In this approach, (linear) dimensionality reduction techniques [5,4] are used to select automatically the most relevant features and their number. In this context, features can be understood as weighted sums of the most relevant trace samples. In addition, both the computational requirements as well as the prohibitive memory usage of standard TA are reduced in a principled way.

2.1 Templates

Suppose that P_k traces of a given operation O_k were recorded. The traces $\{\mathbf{t}_{p_k}\}_{p_k=1}^{P_k}$ are N-dimensional time vectors. In TA a Gaussian noise model is considered [2], meaning that $\{\mathbf{t}_{p_k}\}_{p_k=1}^{P_k}$ are assumed to be drawn from the multivariate Gaussian distribution $\mathcal{N}(\cdot|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, which is defined as follows:

$$\mathcal{N}(\mathbf{t}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = (2\pi)^{-\frac{N}{2}} |\boldsymbol{\Sigma}_k|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\mu}_k)^{\mathrm{T}} \boldsymbol{\Sigma}_k^{-1} (\mathbf{t}-\boldsymbol{\mu}_k)\right\}. \tag{1}$$

Note that the mean $\boldsymbol{\mu}_k$ and the covariance matrix $\boldsymbol{\Sigma}_k$ specify completely the noise distribution associated to the operation O_k. Constructing the templates consists then in estimating the sets of parameters $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$ and $\{\boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

A standard approach is to use the maximum likelihood principle. In this approach, we seek for the parameters that maximize the likelihood of the observations (traces) under the chosen noise model. Maximizing the likelihood is equivalent to maximizing the log-likelihood, which is given by

$$\log \mathcal{L}_k \equiv \log \prod_{p=1}^{P_k} p(\mathbf{t}_{p_k}|O_k) = \sum_{p_k=1}^{P_k} \log \mathcal{N}(\mathbf{t}_{p_k}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2}$$

where $p(\mathbf{t}_{p_k}|O_k)$ is the probability of observing trace $\mathbf{t}_{p_k}$ if we assume that operation O_k was performed on the device. Direct maximization of (2) is straightforward and leads to the following estimates:

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{P_k} \sum_{p_k=1}^{P_k} \mathbf{t}_{p_k}, \quad \widehat{\boldsymbol{\Sigma}}_k = \frac{1}{P_k} \sum_{p_k=1}^{P_k} (\mathbf{t}_{p_k} - \hat{\boldsymbol{\mu}}_k)(\mathbf{t}_{p_k} - \hat{\boldsymbol{\mu}}_k)^{\mathrm{T}}. \tag{3}$$

Note that these quantities correspond respectively to the empirical mean and the empirical covariance matrix associated to the observations $\{\mathbf{t}_{p_k}\}_{p_k=1}^{P_k}$.

2.2 Attack

Assume that the set of possible operations that can be performed on the device is $\{O_k\}_{k=1}^{K}$. In order to determine to which operation a new trace $\mathbf{t}_{\text{new}}$ (for example measured on a different device than the one on which the templates were constructed) corresponds, we apply Bayes' rule. This leads to the following classification rule:

$$\widehat{O}_k = \underset{O_k}{\operatorname{argmax}}\ \hat{P}(O_k|\mathbf{t}_{\text{new}}) = \underset{O_k}{\operatorname{argmax}}\ \hat{p}(\mathbf{t}_{\text{new}}|O_k)P(O_k), \tag{4}$$

where $\hat{p}(\mathbf{t}_{\text{new}}|O_k) = \mathcal{N}(\mathbf{t}_{\text{new}}|\hat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k)$ and $P(O_k)$ is the prior probability that operation O_k was performed. Thus, the classification rule assigns $\mathbf{t}_{\text{new}}$ to the operation O_k with the highest posterior probability. Note that when the operations are equiprobable $P(O_k)$ equals $1/K$.

3 Template Attacks in Principal Subspaces

In practice, the number of samples N per trace is very large, typically $\mathcal{O}(10^5)$ as it depends on the sampling rate of the recording device. A high sampling rate is usually mandatory in order to retain the frequency content of the side-channel. This leads to excessive computational loads and a prohibitively large memory usage. Furthermore, it is expected that only a limited number of time samples are relevant for TA.

Several attempts were made to address these practical issues. Chari, *et al.* [2] select time samples showing the largest difference between the mean traces $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$. Rechberger and Oswald [8] used a similar method; their selection rule is based on the cumulative difference between the mean traces. In addition, the traces are pre-processed by a Fast Fourier Transform (FFT) in order to remove high frequency noise. Another, simple rule is to select the points (after pre-processing) where the the largest variance of the mean traces occur. All these approaches assume that the relevant samples are the ones with the highest variability. However, they only provide heuristics and are therefore by no means optimal. Furthermore, they require to chose an arbitrary minimum distance between successive points (for example the clock cycle) in order to avoid redundancy and there is no satisfactory rule to determine how many such samples are needed to attack optimally.

Another, more systematic approach, which also relies on the data variability, is to select the relevant points based on principal component analysis (PCA) (see for example [5,4]). PCA is a standard statistical tool for dimensionality reduction. It looks for a linear transformation that projects high-dimensional

data into a low-dimensional subspace while preserving the data variance (i.e., it minimizes the mean squared reconstruction error). In order to minimize the loss of relevant information, PCA works in two steps. First, it looks for a rotation of the original axes such that the new coordinate system indicates the successive directions in which the data have maximal variance. Second, it only retains the M most important directions in order to reduce the dimensionality. It assumes therefore that the variability in the discarded directions corresponds to noise. An example is shown in Appendix A.

3.1 Trace Principal Subspaces

Consider a set N-dimensional observations $\{\mathbf{t}_k\}_{k=1}^K$, which are the empirical mean traces associated to the set of operation $\{O_k\}_{k=1}^K$. PCA looks for the first principal directions $\{\mathbf{w}_m\}_{m=1}^M$ such that $N \geq M$ and which form an orthonormal basis of the M-dimensional subspace capturing maximal variance of $\{\mathbf{t}_k\}_{k=1}^K$. It can be shown [5] that the principal directions are the eigenvectors of the empirical covariance matrix, which is given by

$$\bar{\mathbf{S}} = \frac{1}{K}\sum_{k=1}^{K}(\mathbf{t}_k - \bar{\mathbf{t}})(\mathbf{t}_k - \bar{\mathbf{t}})^{\mathrm{T}}. \tag{5}$$

The quantity $\bar{\mathbf{t}} = \sum_{k=1}^K \mathbf{t}_k$ is the average of the mean traces.

In TA, N is typically $\mathcal{O}(10^5)$, meaning that $\bar{\mathbf{S}} \in \mathbb{R}^{N\times N}$ is beyond computation capabilities. Furthermore, the total number of mean traces K is much smaller than N. Matrix $\bar{\mathbf{S}}$ is of rank $K-1$ (or less) and has therefore only $K-1$ eigenvectors. Fortunately, one can compute the first $K-1$ eigenvectors without having to compute the complete covariance matrix $\bar{\mathbf{S}}$ [4].

Let $\mathbf{T} = (\mathbf{t}_1 - \bar{\mathbf{t}}, \ldots, \mathbf{t}_K - \bar{\mathbf{t}}) \in \mathbb{R}^{N\times K}$ be the matrix of the centered mean traces. By definition the empirical covariance matrix is given by $\frac{1}{K}\mathbf{T}\mathbf{T}^{\mathrm{T}}$. Let us denote the matrix of eigenvectors and eigenvalues of $\frac{1}{K}\mathbf{T}^{\mathrm{T}}\mathbf{T}$ by respectively $\mathbf{U}$ and $\boldsymbol{\Delta}$, the latter being diagonal. We have $(\frac{1}{K}\mathbf{T}^{\mathrm{T}}\mathbf{T})\mathbf{U} = \mathbf{U}\boldsymbol{\Delta}$. Left multiplying both sides by $\mathbf{T}$ and rearranging leads to

$$\bar{\mathbf{S}}(\mathbf{T}\mathbf{U}) = (\mathbf{T}\mathbf{U})\boldsymbol{\Delta}. \tag{6}$$

From this expression, we see that $\mathbf{T}\mathbf{U}$ is the matrix of the K eigenvectors of $\bar{\mathbf{S}}$. In order to form an orthonormal basis, they need to be normalized. The normalized principal directions are given by

$$\mathbf{V} = \frac{1}{\sqrt{K}}(\mathbf{T}\mathbf{U})\boldsymbol{\Delta}^{-\frac{1}{2}}. \tag{7}$$

The principal directions $\{\mathbf{w}_m\}_{m=1}^M$ are the columns of $\mathbf{V}$ corresponding to the M largest eigenvalues of $\boldsymbol{\Delta}$. Subsequently, we will denote these eigenvalues by the diagonal matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{M\times M}$ and the corresponding matrix of principal directions by $\mathbf{W} \in \mathbb{R}^{N\times M}$.

As discussed above, PCA can be performed when the number of data vectors is (much) lower than their dimension. Still, one may question the pertinence of the

solution, as a subspace of dimensionality $K-1$ goes exactly through K points. However, the solution found by PCA makes sense if the intrinsic dimension of the data manifold is much lower than number of observations. In other words, the solution is valid if most of the relevant information can be summarized in very few principal directions. Fortunately, this is the case in the context of Template Attacks (see Section 4). Note that the same problematic arises in Computer Vision in the context of automatic face recognition. Here, the very high dimensional vectors are the face images. The principal characteristics are then found by following a similar approach, which is known as *eigenfaces* [12].

3.2 Principal Subspace Based Templates

In the previous section, we showed how standard PCA can be modified in order to be used with very high-dimensional vectors such as traces. This provides us with the projection matrix $\mathbf{W}$, which identifies successively the directions with maximal variance. Now, in order to build PSTA, we assume a Gaussian noise model after projection. So we need to estimate the projected means $\{\boldsymbol{\nu}_k\}_{k=1}^K$ and the covariance matrices of the projected traces along the (retained) principal directions $\{\boldsymbol{\Lambda}_k\}_{k=1}^K$. These parameters are respectively given by

$$\boldsymbol{\nu}_k = \mathbf{W}^{\mathrm{T}}\hat{\boldsymbol{\mu}}_k, \quad \boldsymbol{\Lambda}_k = \mathbf{W}^{\mathrm{T}}\widehat{\boldsymbol{\Sigma}}_k\mathbf{W}. \tag{8}$$

As in standard TA, the noise model is here given by a multivariate Gaussian distribution. However, it is expected that the number of principal directions M is much smaller than N. Note that a direction can be considered as not being principal when the associated eigenvalue is small compared to the largest one. This will be further discussed in Section 4.

Next, in order to classify a new trace $\mathbf{t}_{\mathrm{new}}$, we apply Bayes' rule. This leads to the following classification rule (or attack):

$$\widehat{O}_k = \underset{O_k}{\operatorname{argmax}}\ \hat{p}(\mathbf{W}^{\mathrm{T}}\mathbf{t}_{\mathrm{new}}|O_k)P(O_k), \tag{9}$$

where the distribution in projection space is given by $\hat{p}(\mathbf{W}^{\mathrm{T}}\mathbf{t}_{\mathrm{new}}|O_k) = \mathcal{N}(\mathbf{W}^{\mathrm{T}}\mathbf{t}_{\mathrm{new}}|\boldsymbol{\nu}_k, \boldsymbol{\Lambda}_k)$.

4 Experimental Results

In the experiments, the recorded traces are power leakages. We validate PSTA both on stream ciphers (RC4) and block ciphers (AES Rijndael). Two examples of leakage traces for each encryption algorithm are shown in the Figures of Appendix B.

From a practical point of view, considering a very small number K of different operations/keys can lead to a degenerate solution as only very few principal directions can be identified. This in turn may lead to poorly performing attacks. Therefore, it is convenient to augment the number of mean traces artificially in this case. For example, one can compute for each operation a pre-defined number of mean traces by picking several traces at random in the training set. Another approach is to use resampling techniques from statistics (see for example [3]).

4.1 RC4

The first experiments were carried out on a PIC 16F877 8-bit RISC-based microprocessor [7]. The microchip was clocked at a frequency around 4 MHz. This microprocessor requires four clock cycles to process an instruction. Each instruction is divided into four steps: (i) fetch (update of the address bus), (ii) decode and operands fetch (driven by the bus), (iii) execute and (iv) write back. We monitored the power consumption of a device by inserting a small resistor at its ground pin or power pin. The resistor value is chosen such that it disrupts the voltage supply by at most 5% from its reference[2]. The 1-Ohm method[3] was used to attack the device at the ground pin and a differential probe in the case of targeting the power pin.

RC4 is a stream cipher working on a 256-byte state table denoted S hereafter. It generates a pseudo-random stream of bits which is mixed with the plaintext using a XOR function to yield a ciphertext. The state S is initialized with a variable key length (typically between 40 and 256 bytes) using the following key-scheduling algorithm:

```
for i from 0 to 255
  S[i] := i
j := 0
for i from 0 to 255
  j := (j + S[i] + key[i mod keylength]) mod 256
  swap(S[i],S[j])
```

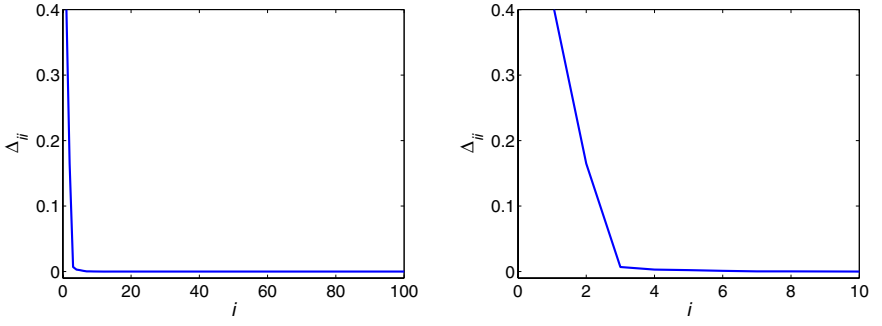
The power consumption of the first iteration was monitored; the dependence on the first byte of the key is here obvious. The 256-byte state was placed in the data memory by allocating 64 bytes per bank. Therefore, it is expected to be easier to distinguish the keys located in different banks even if they have the same Hamming weight.

In the RC4 experiments, 10 keys that are believed to be "close" are considered. For each one, 500 traces are used to construct the models and 300 to validate them. In other words, 500 traces are used to estimate the parameters and 300 to assess the performance. For each trace, there are 300,000 time samples. Figure 1 shows the eigenvalues in decreasing order. Clearly, most of the variance is located in very few components. In practice, 7 components are sufficient to ensure an average rate of correct classification of 93.3% (see Figure 2), meaning that most of the test traces are correctly classified at once.

By contrast, in [2] 42 test samples were selected according to some heuristic. The noise model was chosen to be multivariate Gaussian as in (1). When considering a diagonal covariance matrix (i.e., the time samples are considered

[2] This is advised in IEC 61967-3: Integrated circuits - Measurement of electromagnetic emissions, 150kHz to 1GHz Part 3: Measurement of radiated emissions, surface scan method (10kHz to 3GHz), 47A/620/NP, New Work Item Proposal (July 2001).

[3] See IEC 61967-4: Integrated circuits - Measurement of electromagnetic emissions, 150 kHz to 1 GHz - Part 4: Measurement of conducted emissions 1Ω / 150Ω. Direct coupling method, 47A/636/FDIS, Final Draft International Standard, Distributed on 2002-01-18.



(a) Eigenvalues. (b) Largest eigenvalues.

Fig. 1. Eigenvalues in descending order for RC4

independent) the classification errors reported by [2] were up to 35% for similar keys. Since the power of the attack strongly depends on the implementation and the measurement noise, we also reproduced the experiments for a fully multivariate Gaussian noise model (i.e., for full covariance matrices) for comparison purposes. The samples were selected as the ones where maximal variance occurred. The minimal distance between successive samples was chosen to be equal to the clock cycle. For 42 time samples , the average classification success was 91.8%, which is already considerable. However, note that this approach requires to choose a particular distance between the samples a priori, which affects the performances considerably. For example here, a distance of half the clock cycle leeds to an average classification error of only 80.5%. A similar loss of performance is observed when choosing to few samples to construct the multivariate noise model, but when too many samples are taken, the model reliability might be questionable. Indeed, when the dimension of the data space increases, the number of observations to reliability estimate the parameters needs to increase as well. In the case of standard TA with a 42 points of interest, estimating the mean and the covariance matrix of the multivariate Gaussian noise model requires to fit $M(M+3)/2 = 945$ parameters. However, there is only a limited number of measurements (or traces), typically few hundreds. The number of constraints increases linearly with the dimension M. There are thus only very few measurements to estimate each model parameter.

An important advantage of PSTA over TA is that the number of relevant features can be inferred from the eigenvalues. Only the significant ones need to be retained; the remaining ones are thought of as being noise. Clearly, from Figure 1, it can be observed that only the first two components are important, and indeed, the average correct classification rate for two components is already 88.7% (see Figure 2). The next few components only slightly increase the power of the attack. Furthermore, in the 7-dimensional principal subspace of the traces only 70 parameters need to be estimated (as opposed to 945), while the number of data is the same. The model parameters are thus expected to be more reliably estimated. Note also that a minimal distance between the features needs not to be

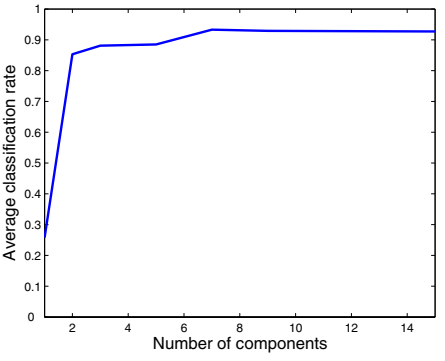


Fig. 2. Average correct classification rate for RC4 as a function of the number of components

chosen in the case of subspace TA. As a matter of fact, the principal components are a weighted sum of many time samples, the weights being determined as the ones minimizing the loss of variance in the data.

4.2 AES Rijndael

Template attacks are usually applied to stream ciphers, key scheduling algorithms and pseudo-random number generators. This is motivated by the fact that such primitives are difficult to target with standard side-channel attacks like the DPA, since the attacker can only observe a single use of the key. However, in general, one could apply template attacks to any kind of cryptographic primitive in order to take advantage of a more efficient information extraction from side-channel observations. For example, in this section we show that an adaptation of subspace based TA can be applied to FPGA implementations of block ciphers. Such a context is practically interesting since it allows to evaluate how the construction of templates may be affected by (large) amounts of algorithmic noise. It also yields particular constraints since the objective is to characterize only a part of the implemented design.

For illustration purposes, let us observe the simplified block cipher of Figure 3, where only one round is represented. In this picture, let us also assume that we want to build templates for the key bits entering the first (upper) substitution box s. Clearly, if we only want to identify the power consumption patterns of this s-box (more specifically, we want to identify the dark grey computations in the scheme, before the application of a diffusion layer), it is important to randomize all the other points in the implementation. They will then contribute to the overall leakage as random noise source. That is, all the inputs to the other s-boxes should be feed with a random number generator. Therefore, we will construct our templates according to the following procedure:

1. Select the target key bits in the implementation.
2. For each key candidate:

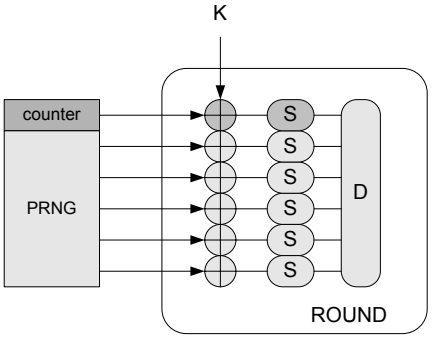


Fig. 3. Simplified view of one round in AES Rijndael. The counter feeds a particular sequence of messages to the device. PRNG is a pseudo-random generator producing arbitraty message sequences. K is the encryption key, S denotes an s-box and D is the diffusion layer of the round.

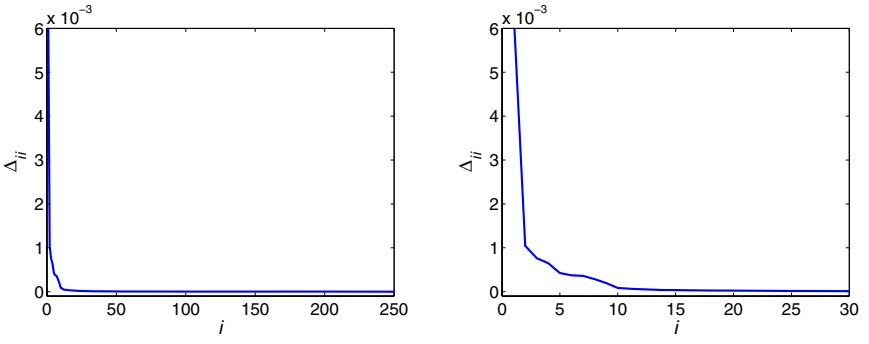
- Feed the s-box corresponding to these target key bits with a deterministic sequence of plaintexts (e.g., a counter).
- Feed the other s-boxes in the scheme with random inputs[4].
- Build the templates from the measurement of these computations.

An important feature of this process is that each key candidate will be characterized by a number of encryptions. This is because every value in the counter will give rise to a computation that identifies these candidates. As a matter of fact, this will allow us to evaluate the efficiency of our template attack, by checking the number of encryptions required to reach a successful classification and therefore to compare our results with previous attacks against similar implementations.

In practice, we targeted an FPGA implementation of the AES Rijndael [11]. Basically, we selected a loop architecture with only one round implemented in the circuit. The key scheduling was not implemented on-the-fly, but executed once, before the execution of our encryptions. However, note that the possible implementation of an on-the-fly key scheduling would not affect the construction of the templates as long as the key is fixed and therefore, once initialized, the key scheduling does not lead to any switching activity anymore.

In the experiments, 10 different keys were considered. For each one, 500 traces were used to estimate the model parameters and 500 to validate the resulting models. The number of samples per trace is equal to $500,000$. Figure 4 shows the eigenvalues for AES Rijndael. Again, it can be observed that most of the variance in the data can be summarized with relatively few components. For example, with 20 components and for 128 encrypted messages the average classification success is equal to 86.7% (see Figure 5). Compared to the results with

[4] Random inputs are used not only when constructing the templates, but also when evaluating the performance of the attack. Therefore, this set up mimics a device with unknown inputs for the other s-boxes as desired. Note that a convenient way to generate these random inputs is to use the feedback from the block cipher outputs.



(a) Eigenvalues. (b) Largest eigenvalues.

Fig. 4. Eigenvalues in descending order for AES Rijndael

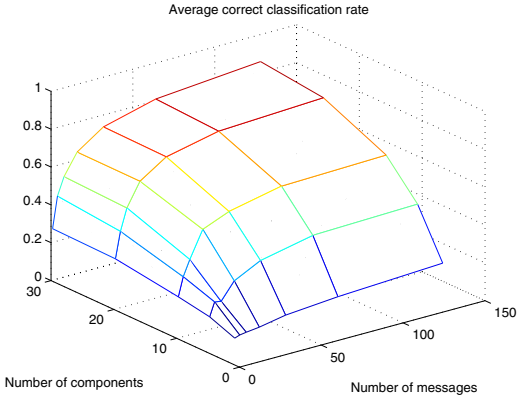


Fig. 5. Average correct classification rate for AES, as a function of the number of encrypted messages and the number of retained components

RC4, a higher number of components is necessary for a comparable classification accuracy. This result can be explained by the fact that the power traces are here much noisier (due to the parallel hardware implementation).

Although, there are relatively few significant components needed with respect to the number of encrypted messages, it is important to realize that it does not mean that the information in most of them is discarded. Indeed, in PSTA, the PCA-step seeks of the optimal projection in the feature space. Each component corresponds thus to a weighted sum of a possibly high number of time samples. Therefore, the information leakage due to a possibly high number of encrypted messages is summarized in a single component.

Figure 5 shows the average correct classification rate as a function of the number of retained components and the number of messages. As expected, when the number of encryptions decreases, the performances drops. This is due to the fact that there is less information leakage available. Similarly, when the number of

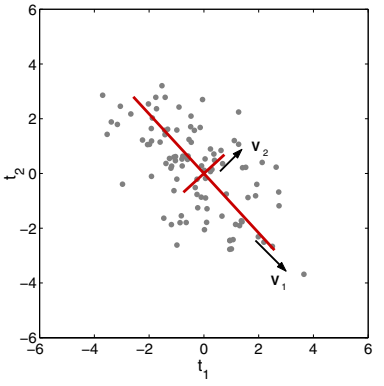


Fig. 6. Illustration of principal component analysis (PCA)

components is small, there is only a poor capacity to classify correctly, as too many relevant features have been discarded. However, when the number of messages and the number of components increases the average correct classification rate rapidly increases.

Compared to recent correlation-based power analysis attacks of AES Rijndael (also on FPGA), the number of message required to recover the correct key bytes is much smaller. The factor of proportionality ranges from 2 to 5 depending on the fact that the attack uses trace averaging [10] or not [9]. Note also that correlation attacks require in general to carefully preprocess the traces, for example using several filters. By contrast, PSTA is much more practical as it exploits the information in the raw data directly and does not require to adjust any tuning parameters, but the number of components to retain.

5 Conclusion

In this work, we introduced principal subspace template attacks and showed that they can be successfully applied to both stream and block ciphers. Preprocessing the leakage traces beforehand by PCA allows avoiding the practical issues of ordinary template attacks. Principal subspace template attacks are motivated by the fact that template attacks consider the time instants having a great variability as being important to discriminate. If this assumption is correct, then PCA is the optimal (linear) transformation to identify the most relevant features. Besides, the eigenvalues provide a systematic rule for determining how many and which features should be selected to mount a powerful attack. Finally, it is also important to realize that the main difference between both attacks resides in the way they extract information from traces. In template attacks M of the N samples are used to mount the noise model, the selection being based on heuristics, while in principal subspace template attacks M linear combinations (preserving maximal variance) of these N samples are used.

References

1. Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.
2. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 2523 of *Lecture Notes in Computer Science, 13–28. Springer*, 2002.
3. B. Efron and R.J. Tibshirani. *An introduction to the Bootstrap*. Chapman and Hall, London, 1993.
4. K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Elsevier, New York, 1990.
5. I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.
6. Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *16th Annual International Cryptology Conference (CRYPTO)*, volume 1109 of *Lecture Notes in Computer Science, 104–113. Springer*, 1996.
7. Microship. PIC16F877 datasheet. url: ww1.microchip.com/downloads/en/ DeviceDoc/30292c.pdf, 2001.
8. Christian Rechberger and Elisabeth Oswald. Practical template attacks. In Chae Hoon Lim and Moti Yung, editors, *5th International Workshop on Information Security Applications (WISA)*, volume 3325 of *Lecture Notes in Computer Science, 440–456. Springer*, 2004.
9. F.-X. Standaert, S.B. Ors, and B. Preneel. Power analysis of an FPGA implementation of Rijndael: Is pipelining a DPA countermeasure? In Marc Joye and Jean-Jacques Quisquater, editors, *6th International Workshop Cryptographic Hardware and Embedded Systems (CHES)*, volume 3156 of *Lecture Notes in Computer Science, 30–44. Springer*, 2004.
10. F.-X. Standaert, E. Peeters, F. Macé, and J.-J. Quisquater. Updates on the security of FPGAs against power analysis attacks. In *proceedings of ARC 2006*, LNCS 3985, pp. 335-346, 2006.
11. F.-X. Standaert, G. Rouvroy, J.-J. Quisquater, and J.-D. Legat. Efficient implementation of Rijndael encryption in reconfigurable hardware: Improvements and design tradeoffs. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *5th International Workshop Cryptographic Hardware and Embedded Systems (CHES)*, volume 2779 of *Lecture Notes in Computer Science, 334–350. Springer*, 2003.
12. M. Turk and A.Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

A Appendix

An illustration of PCA is shown Figure 6. The data is drawn from a 2-dimensional Gaussian distribution. The two principal directions $\mathbf{v}_1$ and $\mathbf{v}_2$ are shown by the solid lines. The length of the lines is proportional to the variance of the projected data onto the corresponding direction. If we remove the second dimension (after rotation) and describe the data only by the first one, then we will minimize the loss of information (i.e., loss of variance) due to this new representation.

B Appendix

The examples of the recorded RC4 and AES Rijndael power traces are shown respectively in Figure 7 and 8.

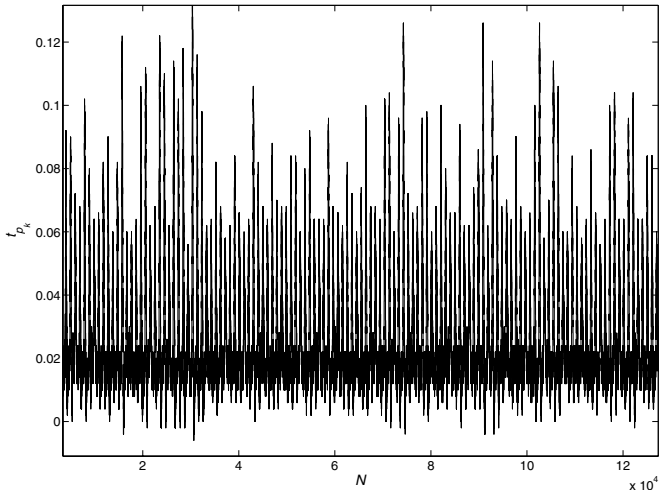


Fig. 7. Example of a RC4 power trace

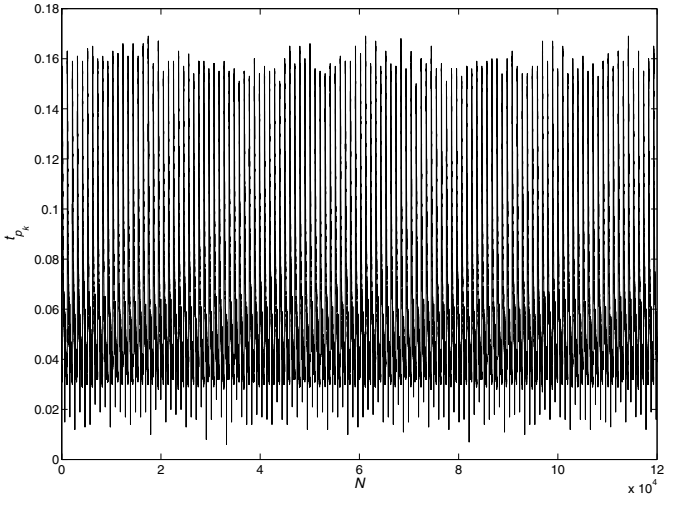


Fig. 8. Example of an AES Rijndael power trace