Practical Template Attacks

Christian Rechberger[1,2] and Elisabeth Oswald[1,2]

[1] A-SIT Secure Information Technology Center - Austria
Inffeldgasse 16a, A–8010 Graz, Austria
[2] Institute for Applied Information Processing and Communciations (IAIK)
Graz University of Technology
Inffeldgasse 16a, A–8010 Graz, Austria
{christian.rechberger, elisabeth.oswald}@iaik.tugraz.at ***

Abstract. Side-channel attacks pose a serious threat to implementations of cryptographic algorithms. In the pioneering article of Chari, Rao and Rohatgi, the general idea behind template attacks was introduced. Template attacks apply advanced statistical methods and can break implementations secure against other forms of side-channel attacks.
However, in our research it turned out that several details, which are essential to practical implementations of template attacks, still need to be answered. In this article we provide answers to open issues, such as how to select points of interest in an efficient way, or how to preprocess noisy data. In addition, we show the benefits of trial classifications and we point out that in practice so-called amplified template attacks have to be considered as a potential threat.

Keywords: Side-Channel Analysis, Template Attack, DPA, DEMA

1 Introduction

Devices performing cryptographic operations can be analyzed by various means. Traditional cryptanalysis looks at the relations between input and output data and the used keys. However, even if the implemented algorithms are secure from a cryptanalysis point of view, side-channel attacks pose a serious threat. Side-channel attacks are a subgroup of implementation attacks. Examples thereof are timing attacks [Koc96], power attacks like DPA or SPA [KJJ99], EM attacks [AARR02], error message attacks [Ble98,KPR03], or combinations of different sources [WT01,ARR03].

Traditional DPA/DEMA style attacks assume the following threat model: The secret key stored in the device is used to perform some cryptographic operations. The attacker monitors these operations using captured side-channel information like power consumption or electromagnetic emanation. The attack is successful if the used secret key can be reconstructed after a certain number of operations. If the number of operations is limited by the protocol used to initiate

*** This work was supported by A-SIT and by the FWF "Investigations of Simple and Differential Power Analysis" project (P16110-N04).

these operations, the attacker has an upper bound on the number of operations he can observe.

If the operation, which leaks usable side-channel information, is executed just once, the threat model is different: The attacker has to reconstruct the secret key using a single trace of side-channel information. Besides protocol limitations, ephemeral keys can be the reason for such a constraint. Techniques like SPA or SEMA [QS01] are a general way to tackle this problem. These techniques use easily distinguishable features of operations like double and add, or add and multiply, to infer key-bits. The majority of the available literature deals with these two types of scenarios.

If the observed signal-to-noise ratio is not high enough, or the implementation is done in a way that ensures the used operations being independent of the key(*i. e.* no key-dependent jumps), SPA/SEMA style attacks are not possible anymore. The attacker has to think of other ways to get hold of the secret key: One way to do this is to use a similar device and build a model of it. Using this model, an attacker might now be able to recover the secret key. Only very few publications [BS99,FP99,CRR03] deal with this model.

A very general and powerful way to perform such a two-stage attack, called template attack, is given in [CRR03]. The key concept is to store the probability distribution of leaking information for each device state. Signal classification techniques are subsequently used to assign a captured trace from a device to one device state. This technique is computationally intensive, execution time and storage requirements are very high. The result is a reduced set of probable secret keys.

1.1 Contribution of this Work

In order to make template attacks more practical, two goals need to be considered. The first one is to decrease the computational requirements of the attack. Secondly, the size of the set of probable keys should be kept as low as possible, subsequently referred to as the accuracy of the attack. To reach these goals, we address issues in this article such as:

- **Trial classification**. A separate step in the course of a template attack named *trial classification* is introduced in Section 3.1. We propose to use trial classifications to improve the accuracy of a template attack.
- **Points of interest**. For a template attack to be practical, it is paramount that not all points of a trace are part of the template. To reduce the number of points, a standard technique called principal component analysis could be used. Due to the high computational requirements of this technique, we propose a much simpler and faster approach in Section 3.2. Additionally, we suggest several properties of the selection of points of interest. Hence the algorithm is improved even further.
- **Preprocessing**. We introduce a *preprocessing step* in Section 3.3. We show that the use of discrete Fourier transformation on traces significantly improves attack results in practice. This is illustrated using side-channel information from power consumption and EM emanation. In Section 4, we give

evidence that in cases where too much data independent (ambient) noise in the input data renders a template attack impossible in the *time domain*, a transformation of input traces into the *frequency domain* is highly advantageous.
- **Amplified template attacks**. By extending the threat model, the classification results of template attacks can be increased. This approach is especially useful when the number of allowed iterations of critical operations is not high enough to allow a DPA style attack.

The remainder of the paper is organized as follows: The next section revisits template attacks and covers the ideas behind it. Requirements of an attack to be practical are stated. Section 3 proposes several means to meat these requirements. In Section 4 we illustrate the power of our proposals considering a concrete example.

2 Template Attacks

The method used in the template attack is as follows: First a strategy to build a model of all possible operations is carried out. The term "all possible operations" usually refers to the cryptographic algorithm (or a part of it) being executed using all possible key values.

The model of captured side-channel information for one operation is called *template* and consists of information about the typical signal to expect as well as a noise-characterization for that particular case. After that, the attack is started and the trace of a single operation (*e. g.* the key-scheduling of an algorithm) is captured. Using the templates created before which represent all key-values, the side-channel information of the attacked device is classified and assigned to one or more of the templates. The goal is to significantly reduce the number of possible keys or even come up with *the* used key.

However, this idealized approach faces several real-world problems. To lay the foundations for a successful attack, these problems are dealt with now.

2.1 Approaching the Template Attack

Assuming the captured trace consists of b sampled points and each sampled point consists of a signal and a noise part, one gets a b-dimensional noise-vector per trace. In general, all elements of this vector are drawn from different unknown probability distributions. A key point here is that the data-dependance of the noise is reflected by the probability distribution it is coming from. The challenge of modeling this noise vector can for example be simplified by assuming a normal distribution for each element and linear combinations thereof. This leads to the well-known multivariate-gaussian noise model. Simpler noise models like univariate models have been shown to be inadequate for practical purposes [CRR03].

2.2 Building Templates

In order to classify a trace, a template for each of the possible operations has to be built. The template reflects the statistical properties of such a trace, or in other words, the properties of the probability distribution of all its points. Using the multivariate gaussian model, a template consists of a vector of means and a matrix of covariances. To create such a template, a number of traces have to be captured; more traces lead to a more accurate model.

Let us assume that n different operations need to be distinguished and for every operation a number of p traces named t_1, .. t_p have been captured. Subsequently, we consider just one out of n operations.
The vector of means can be calculated as follows:

$$\bar{t} = M = \frac{1}{p}\sum_{j=1}^{p} t_j \tag{1}$$

Keep in mind, that all traces t_j are taken from the same operation. The next step is to calculate the noise vectors. For every trace t_j of a operation, the corresponding noise vectors N_j are $t_j - M$. They are the basis for the covariance matrix of the noise, which is the second part of the template. But first we define the covariance, which is sometimes referred to as a means to measure the linear dependance between two random variables. The empirical covariance of two random variables X and Y is defined as

$$\operatorname{cov}(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y}) \tag{2}$$

In order to describe the covariances between more than two random variables, a matrix of covariances is needed. In our case, the entries in the noise covariance matrix can be defined as:

$$CM(u,v) = \operatorname{cov}(N_u, N_v) \tag{3}$$

Hence the covariance of all pairs of noise vectors is included. Note that the covariance matrix is symmetric and that elements on the diagonal of the covariance represent the variance for that column.

For each of the n possible operations, the corresponding template is the tuple (M, CM).

2.3 Classifying Traces

Once all templates are derived using a programmable device, they can be used to classify a trace t captured from the device under attack. Using the maximum-likelihood principle, the goal is to find the template, that “fits” best to the captured trace. The following steps have to be done to accomplish this.

1. For each template (which represents a certain key value), compute the noise N_i of the trace as if the device under attack was using the key value the template is associated to. M_i refers to the vector of means of the used template.
$$N_i = M_i - t \tag{4}$$
2. The probabilities of observing this n-dimensional noise vector N_i can be calculated using the noise covariance matrix of each template. Assuming that every point of the trace is drawn from a gaussian shaped probability distribution, the formula for the n-dimensional multivariate gaussian probability distribution can be used:
$$p(N_i) = \frac{1}{\sqrt{(2\pi)^n |CM_i|}} \cdot e^{-\frac{1}{2} N_i^T CM_i^{-1} N_i} \tag{5}$$
If the assumption of a gaussian shaped noise holds, then the maximum likelihood approach of selecting that template with the highest probability of observing the calculated noise is optimal.

2.4 Requirements for an Attack in Practice

In this subsection we focus on some properties of the template attack, which can render it impractical if they are not considered properly.

Classification Performance It is not possible to perform the classification process on the entire key space at once, since building that many templates is not feasible. The iterative approach to cope with this problem, called extend and prune strategy, is explained in [CRR03].

The goal is to keep the number of possible keys which remain after the pruning step small at every iteration. More efficient classification performance results in better pruning steps. Depending on the power of the classification, or their *accuracy*, the number of required templates explodes more-or-less towards the end of the attack. For a successful attack, it is sufficient that the exhaustive search on the remaining possible keys is feasible.

Trace Length Depending on the measurement setup and the data acquisition strategy, captured traces can be quite big (*i. e.* the number of sampled points is high). This has several implications for the attack:

– In the multivariate model, the covariance matrix has the highest storage requirements. The size of a covariance matrix depends on the number of points considered. Hence, the memory requirements of the templates grow quadratically with the number of points.
– Calculating the observation probability involves a matrix inversion. As a consequence, the running time of the device characterization step grows nearly cubically (depending on the used algorithm for matrix inversion) with the number of points.

Especially the latter observation can have devastating effects on the practicality of an attack. Consequently a general way has to be found to deal with this problem. The goal is as follows:

> Having a trace of size N, select those $n < N$ points of the trace, which provide the most information on the used key.

Subsequently these n points are referred to as *points of interest* or *selected points*. Choosing a reasonable number (which is explained subsequently) always leads to template sizes significantly smaller than the size of one trace. For a template attack to be practical it is paramount that n is small enough to allow a fast calculation of observation probabilities. Our efficient way to tackle this problem is shown in Section 3.2.

The classification performance is equally important, since it directly affects the pruning step which in turn limits the number of templates to be built. In Section 3.3 we propose a novel way to increase classification performance in practice.

3 Our Way to make Template Attacks more Practical

An RC4 [Sch96] implementation on an 8-bit microcontroller is considered subsequently. Side-channel leakage in the form of power consumption or EM emanation of the initialization part of RC4, also known as the key scheduling, is used to identify the used key. A more detailed description of the attacked implementation and the measurement setup can be found in Appendix A.

3.1 Trial Classification

We introduce an extension to the general two-stage model: The programmable device in the first step of the attack can not only be used to build templates but also to perform trial classifications in order to obtain good parameters for the actual attack. Note that this step is possible for every template attack and significantly improves results. Following this approach, the efficiency of building templates carries even more weight.

3.2 Efficiently Choosing Points of Interest

In order to find those n out of all N points providing the most information for a template, principal component analysis [Jol02] (PCA) can be used. PCA is mainly used in multivariate statistics to reduce the dimensionality of a data set. This procedure rotates existing axes to new positions such that maximum variabilities are projected onto the axes. This way, the most interesting features of the data-set are preserved. It was applied to similar problems in side-channel analysis as well [BNSQ03]. Adapted to our problem, this approach delivers a ranking of the points in our traces concerning their dependance on keys. However, this comes at the price of high computational requirements. Therefore we suggest the following simpler and more efficient strategy:

– Take the vectors of means calculated while building the templates.
– Compute differences of each pair of mean vectors and sum them up.
– Select n point among the highest peaks.

Note that the last step needs to be done, even if PCA is used. For each trial classification, the process of choosing points of interest needs to be repeated. Hence any improvement in this step has an even greater impact on the practicality of an attack. We achieved a speedup factor 1000 for a typical setting, without affecting the classification performance.

In Figure 1, the sum of differences trace for one round of the RC4 key-schedule is plotted. Several distinct sets of traces (each containing 100 traces of $100\mu s$ length) were used as input. The area between the points $40\mu s$ and $70\mu s$ is clearly visible as the most interesting part of the plot, since the rest of the trace is not dependent on the key.

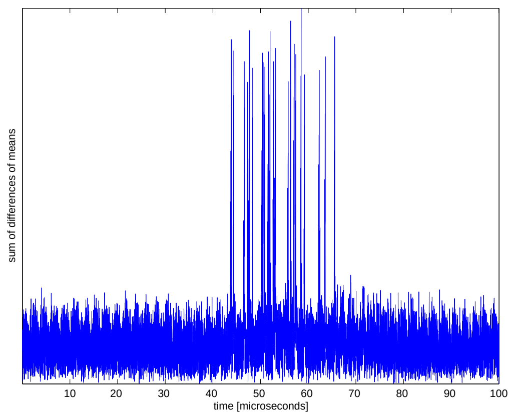

Fig. 1. Sum of differences trace

Choosing the n highest points to serve as the points of interest has been proven to be non-optimal during our experiments. A more efficient strategy is possible if some properties of good selections are known in advance. We propose, that selected points should have the following properties:

– The minimum distance between these points should be approximately a clock cycle or more, since additional points in the same clock cycle do not provide additional information.
– The minimal height of a selected point should be higher than the noise floor of the sum of differences trace.

An algorithm for choosing the n highest peaks that follows these constraints was used in all our subsequent experiments. Disobeying these constraints leads to poor classification performance even if a higher number of points of interest is chosen.

To find a good trade-off between these parameters, trial classifications should be used. To highlight the influence of the number of selected points on the classification performance, a number of experiments were carried out. A template attack on a small part of the used key was performed with different numbers of points of interest. Figure 2 shows the classifications performance as a function of the number of selected points. The considered numbers of points range from 1 to 40. Additionally three different minimum distances were chosen. Results show that 15-20 is a good number of points of interest. If the minimum distance was chosen to be 20, not more than 20 points of interest were found above a level of 40% of the highest peak. The level starting from 20 is indicating this fact.

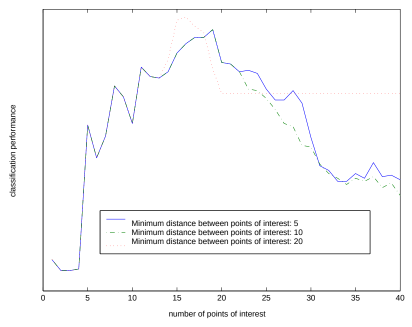

Fig. 2. Performance as a function of the number of selected points

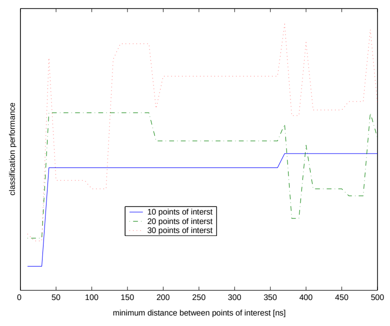

Fig. 3. Performance as a function of the minimum distance

Figure 3 gives another point of view processing the same behavior. For each graph, the number of points of interest was kept constant and the minimal distance between these points was varied between 10 and 500 ns. Even though the graph seems to be chaotic, one conclusion can be drawn: The minimal distance is this attacking setup should be at least 50.

Even though the given values depend on the implementation and can not be used directly for a template attack on another implementation, the outlined strategy can be used for other implementations. Additionally, our results lead to the conclusion, that classification results are declining if the number of points of interest gets too high.

3.3 Preprocessing

In practical side-channel analysis, the raw input data is often preprocessed. Sometimes this is just due to simplicity or efficiency reasons, *e. g.* summarizing sampled points. There are however cases where the preprocessing step heavily affects the results. Even if no thinkable transformation can add additional information to a signal, information extraction procedures do improve. The template attack under consideration is such a case and a lucrative preprocessing transformation is described subsequently.

It turns out that the transformation of the input traces from the time domain into the frequency domain is such a lucrative transformation. In our practical work, an FFT algorithm was used to accomplish this transformation (a fast algorithm to calculate the discrete Fourier transform, for background information refer to [BP85]). In order to show the impact of this preprocessing step a number of experiments were carried out. First some characteristic differences between time domain analysis and frequency domain analysis are illustrated. Afterwards, to highlight the influence of the number of selected points on the classification performance in the frequency domain, a number of experiments were carried out.

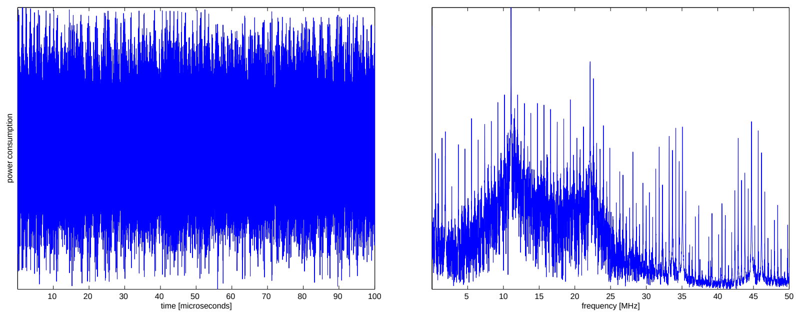

Fig. 4. Power trace of one round of the RC4 key schedule in the time domain (0-100μs) and in the frequency domain(0-50MHz)

Figure 4 contrasts both possibilities. It shows the power trace of one round of the RC4 key schedule in the time domain (ranges from 0-100μs) and in the frequency domain(ranges from 0-50MHz). The transformed input can also be interpreted as the frequency spectrum. The regions with the highest peaks indicate the clocking frequency and multiples of it. To amplify lower peaks, the vertical axis is logarithmical. In Figure 5, the sum of differences of means trace is plotted. Naturally, the key dependent areas are also in the range of the clocking frequency and multiples thereof.

After preprocessing, the resulting traces can be used to perform a template attack in exact the same way as without preprocessing. There is however a difference in the number of points to consider. Figure 6 shows the classifications results as a function of the number of selected points after preprocessing. The considered numbers of points are ranging between 1 and 40. Additionally three different minimum distances where chosen. Results show that much less points are sufficient in comparison to a template attack without the preprocessing step.

At the price of performing an FFT on every input trace (those used to build up the templates as well as those to classify) we get a major advantage. The template classification can be done much faster and storage requirements are lower, since the size of n influences those requirements as outlined in Section 2.4.

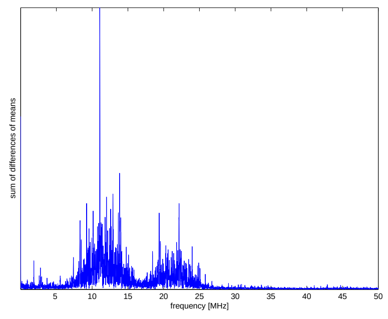

Fig. 5. Sum of differences of means trace for the preprocessed inputs)

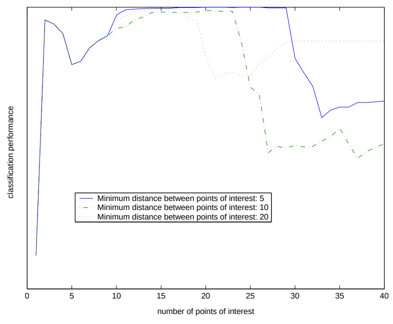

Fig. 6. Performance as a function of the number of selected points

Attention has to be paid that considerations on the lower bound for selected peaks - as done in the time domain - are not directly applicable. In practice the lower bound is much smaller in the frequency domain.

3.4 Amplified Template Attack

Even if the aim of a template attack is to recover the secret key using a single trace, in many real world settings implementations allow for several iterations of the same operation with the same secret key. The application of template attacks is not restricted to stream ciphers like RC4 and can be applied to block ciphers as well. Since every symmetric cipher contains some sort of key scheduling mechanism which processes the secret key, this generalization is possible. Smartcards often use block ciphers for encryption or authentication, hence let us consider the following example: A malicious petrol station tenant, named Eve, is using a modified smartcard based payment terminal. Everytime a customer uses this terminal, Eve captures one trace of side-channel information. This single trace could already be used by Eve to carry out a template attack.

However, some customers are coming again and Eve gets hold of another trace. The template attack can easily be extended to take advantage of such situations, e.g. by adding up noise-probabilities $p(N_i)$ of every captured trace and applying the maximum-likelihood approach on these sums. As a consequence, the power of the attacker is *amplified*. Using this approach, if n is the number of iterations, the error probability of template classification is reduced by the factor $\sqrt{n}$. We have experimentally verified this improvement.

This improvement illustrates several important advantages over DPA style attacks. The first one is a property of the template attack: there is no need to know input or output data of the used algorithm. Additionally, in cases where too few traces are available to facilitate a DPA style attack, template attacks can take advantage of every single trace using the amplified template attack.

4 Applying the Improvements in Practice

We give an example where a classification is not possible at all without preprocessing in Section 4.1. Thereafter, in Section 4.2, we show that with the same measurement setup and with preprocessing, the template attack is successful. The measurement setup used in both scenarios is described in Appendix A.

4.1 Template Attack without Preprocessing

If measurements of the power consumption of the attacked device are not possible for an attacker, EM probes are the next alternative. Even if information about the power consumption is available for the attacker it can be advantageous to consider EM channels additionally [ARR03].

Compared to power measurements, EM measurements are much noisier. Even if the character of the noise is an important part of the built templates, ambient noise originating outside the device is of no value for the attack. In order to highlight the difference we call this noise data independent noise. As in Section 3.2, in a first step we capture traces while using different key values and sum up their pairwise differences. The result is depicted in Figure 7. Compared to Figure 1 there are no key-dependant areas observable anymore. Subsequent classification trials delivered no usable result.

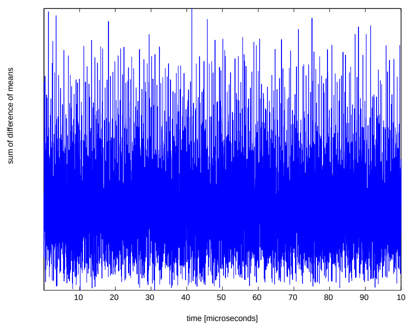

Fig. 7. Sum of differences of means trace for captured EM traces

If data independent noise is superposing the signal originating from the device, averaging techniques have to be used to eliminate it. However this can be quite cumbersome. In our particular setup, increasing the number of needed traces to build a template from 100 (which was sufficient for power traces) to 1000 did not change the bad classification results. Increasing the number of traces even more might lead to better results at some point, but on the other hand it would render the actual template attack for the complete keyspace impractical.

4.2 Template Attack with Preprocessing

Using the preprocessing step explained in Section 3.3, increasing the number of traces is however not necessary and still leads to much better classification results. In this case the error probability was 2% on average. The resulting sum of differences of means trace is depicted in Figure 8. Key dependent frequency areas are clearly visible.

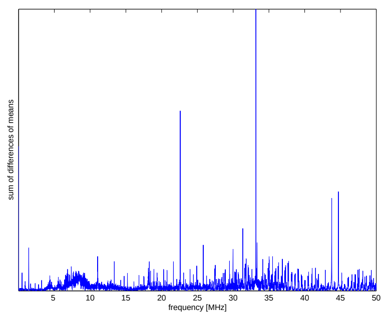

Fig. 8. Sum of differences of means trace for the preprocessed EM traces

There are several similarities to the experiment using power traces as input as shown in Figure 5. Again the sum of differences trace is reflecting the spectral density of the original power traces. However the highest peaks are not located at the clock frequency, but at some multiples of it.

Figure 9 shows the performance of our approach as a function of the number of selected points of interest. Compared to the similar experiment on traces gained from power measurements (see Figure 6) a much bigger number of selected points is needed.

In contrast to the power measurements, there is a distinctive leap considering different minimal distances between selected points. This can be seen in Figure 10, where the performance of the sum of differences of means method for preprocessed EM traces is plotted as a function of the minimum distance between two points of interest. Similar to the time domain results for power measurements, choosing points within a certain distance in frequency does not help classification. The slight variations of classification performance after the leap are due to the selection algorithm and no additional implications result from

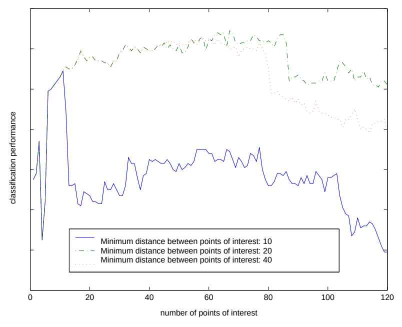

Fig. 9. Performance as a function of the number of selected points

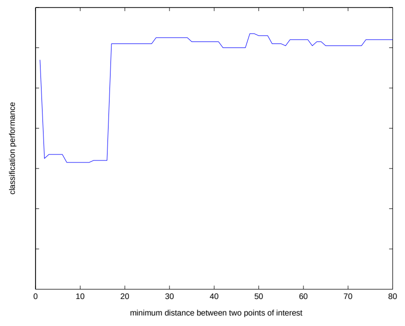

Fig. 10. Performance as a function of the minimum distance

them. They result from choosing slightly different peaks in the same frequency areas in order to meet the minimum distance constraints.

5 Conclusion

Starting from the pioneering work of [CRR03], where several important questions concerning the implementation of template attacks are left open, we proposed several improvements.

First, we outlined a way to perform a template attack in more detail: In order to find optimal parameters for the attack like number and position of points of interest, a first step named trial classification was introduced. Using this trial classification, we ensured the practicality of the attack by introducing a efficient algorithm that replaces principal component analysis. We increased the speed of selecting points of interest by a factor of 1000 without impairing the classification performance.

Additionally, we introduced the preprocessing step. A transformation of traces from the time domain into the frequency domain is suggested. This improved classification results and lowered runtime and storage requirements.

Finally, the power of our proposals is illustrated on a concrete example. Due to much ambient noise, classification is not possible at all without preprocessing in this example. Using our proposed preprocessing technique combined with trial classifications however proved to be very valuable and resulted in practical classification probabilities.

References

[AARR02] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The EM Side-channel(s). In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Proceedings of CHES 2002*, volume 2535 of *LNCS*, pages 29–45. Springer, 2002.

[ARR03] Dakshi Agrawal, Josyula R. Rao, and Pankaj Rohatgi. Multi-channel Attacks. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *Proceedings of CHES 2003*, volume 2779 of *LNCS*, pages 2–16. Springer, 2003.

[Ble98] Daniel Bleichenbacher. Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, number 1462 in LNCS, pages 1–12. Springer, 1998.

[BNSQ03] Lilian Bohy, Michael Neve, David Samyde, and Jean-Jacques Quisquater. Principal and Independent Component Analysis for Crypto-systems with Hardware Unmasked Units. In *Proceedings of e-Smart 2003*, 2003.

[BP85] CS S. Burrus and Thomas W. Parks. *DFT/FFT and Convolution Algorithms and Implementation.* John Wiley & Sons, 1985.

[BS99] Eli Biham and Adi Shamir. Power Analysis of the Key Scheduling of the AES Candidates. In *Second Advanced Encryption Standard (AES) Candidate Conference*, Rome, Italy, 1999.

[CRR03] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template Attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Proceedings of CHES 2002*, volume 2535 of *LNCS*, pages 13–28. Springer, 2003.

[FP99] Paul N. Fahn and Peter K. Pearson. IPA: A New Class of Power Attacks. In Çetin Kaya Koç and Christof Paar, editors, *Proceedings of CHES'99*, volume 1717 of *LNCS*, pages 173–186. Springer, 1999.

[Jol02] Ian T. Jolliffe. *Principal Component Analysis.* Springer, 2nd edition, 2002.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Michael Wiener, editor, *Advances in Cryptology - CRYPTO '99*, volume 1666 of *LNCS*, pages 388–397. Springer, 1999.

[Koc96] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.

[KPR03] Vlastimil Klima, Ondrej Pokorny, and Tomas Rosa. Attacking RSA-Based Sessions in SSL/TLS. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *Proceedings of CHES 2003*, volume 2779 of *LNCS*, pages 426–440. Springer, 2003.

[QS01] Jean-Jacques Quisquater and David Samyde. ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards. In Isabelle Attali and Thomas P. Jensen, editors, *Proceedings of Smart Card Programming and Security*, volume 2140 of *LNCS*, pages 200–210. Springer, 2001.

[Sch96] Bruce Schneier. *Applied Cryptography, 2nd Edition.* John Wiley & Sons, 1996.

[WT01] Colin D. Walter and Susan Thompson. Distinguishing Exponent Digits by Observing Modular Subtractions. In David Naccache, editor, *Topics in Cryptology - CT-RSA 2001*, volume 2020 of *LNCS*, pages 192–207. Springer, 2001.

A Attack Implementation and Measurement Setup

For our practical experiments on the template attack, we used RC4. The algorithm was implemented on an ATMEL AT89S8252. This 8 bit microcontroller is operated on a clock frequency of 11.059MHz. For data acquisition we used the digital oscilloscope Lecroy LC584AM, having a resolution of 8 bit. In most of our experiments we used a sampling rate of 100 MSamples/s.

RC4 specifies a whole family of algorithms whose differences lie in the used word-size n; typically the word-size is 8. The algorithm consists of two parts which are executed sequentially:

- An initialization phase or KSA (Key Scheduling Algorithm) with 2^n rounds
- An output phase or PRGA (Pseudo Random Generation Algorithm) outputting one word per round.

Both parts access an internal table of size $n * 2^n$ bits. In order to keep our implementation of RC4 completely inside the internal RAM, the word size was reduced to 7 bit. Read/write access to external RAM resources would have leaked too much side-channel information and would therefore have simplified the attack unrealistically. For our template attack on RC4, we just considered the KSA.

Figure 11 depicts our measurement setup. It shows our evaluation board with the ATMEL microcontroller, the EM probe used for our EM measurements and the differential probe used for our power measurements.

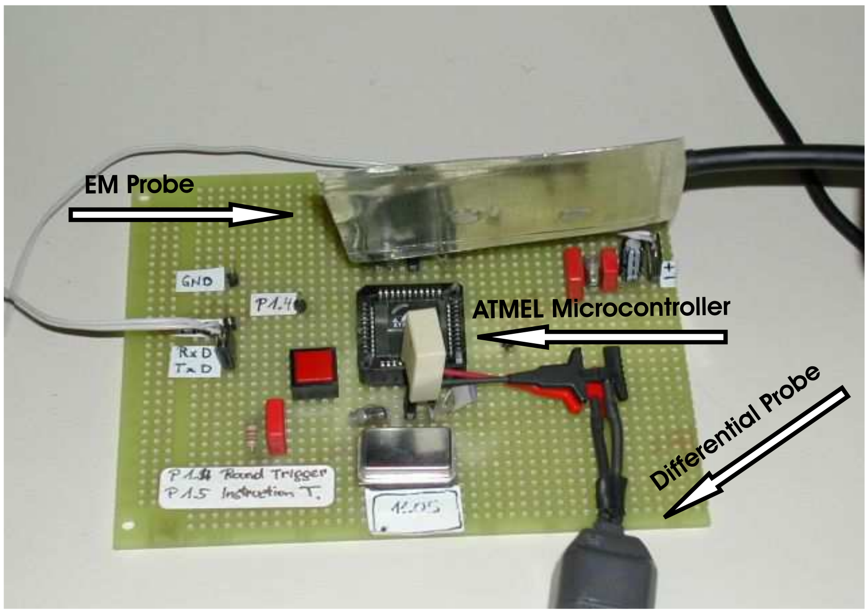

Fig. 11. Measurement setup