A Comparison of χ^2-Test and Mutual Information as Distinguisher for Side-Channel Analysis

Bastian Richter, David Knichel, and Amir Moradi

Ruhr University Bochum, Horst Görtz Institute
Bochum, Germany
firstname.lastname@rub.de

Abstract. Masking is known as the most widely studied countermeasure against side-channel analysis attacks. Since a masked implementation is based on a certain number of shares (referred to as the order of masking), it still exhibits leakages at higher orders. In order to exploit such leakages, higher-order statistical moments individually at each order need to be estimated reflecting the higher-order attacks. Instead, Mutual Information Analysis (MIA) known for more than 10 years avoids such a moment-based analysis by considering the entire distribution for the key recovery. Recently the χ^2-test has been proposed for leakage detection and as a distinguisher where also the whole distribution of the leakages is analyzed.
In this work, we compare these two schemes to examine their dependency. Indeed, one of the goals of this research is to conclude whether one can outperform the other. In addition to a theoretical comparison, we present two case studies and their corresponding practical evaluations. Both case studies are masked hardware implementations; one is an FPGA-based realization of a threshold implementation of PRESENT, and the other is an AES implementation as a coprocessor on a commercial smart card.

Keywords: chi squared test · mutual information analysis · side-channel attacks

1 Introduction

When developing real-world cryptographic applications, implementation attacks pose a serious threat. The past has shown that cryptographic applications like locking systems [5,17], one-time-password tokens [16], RFID cards [15], and mesh networks [4] not incorporating strong countermeasures are susceptible to Side-Channel Analysis (SCA) attacks. Thus, it is very important to harden these during development and to thoroughly test by performing possible attacks.

Within the area of countermeasures against SCA attacks, masking is widely considered as the most important one since it can give certain guarantees, e.g. threshold implementations [14] concerning glitches in the implementation. By splitting the computation into shares, direct leakage of the state can be prevented

and not only reduced or covered by noise like with hiding countermeasures. In the univariate case these masking schemes are developed to prevent leakage up to a certain statistical order, i.e. a first-order masking prevents extracting information via the first statistical moment but might still be attackable via the second centered statistical moment. However, higher-order statistics are more susceptible to noise, so the required number of traces to sufficiently approximate the statistics are increasing exponentially with the order [19].

Further, when implementing masking schemes, the designers always have to test whether some physical side-effects of the platform are influencing the effectiveness of the countermeasure. Here, especially coupling [10,3] is a main problem for hardware implementations which can lead to unpredicted leakages in lower orders due to undesired interaction between the shares.

An initial test is usually performed by using a leakage detection method often based on the t-test [8]. In 2018 the χ^2-test [12] was proposed as an addition to the t-test. While the t-test highlights leakage in each order individually, the χ^2-test considers the whole distribution and can thus detect leakage spread over multiple orders. Leakage detected with the χ^2-test but not in the t-tests can indicate that the noise in the measurements is not sufficient to cover the leakage in the higher orders.

However, leakage detected in these tests does not necessarily indicate exploitable leakage. To further examine if detected leakage can be exploited, attacks have to be executed. The original Correlation Power Analysis (CPA) [2] attack correlates the measurements with a hypothetical power model based on a guessed key and thus targets the first order. Fortunately, the attack can be extended to higher orders by preprocessing the measurements [19] (c.f. Section 2.5). But still, this only attacks individual orders like the t-test. To attack the whole distribution different attack methods are needed. The first one was Mutual Information Analysis (MIA) presented by Gierlichs et al. in 2008 [7] which computes the mutual information between the measurements and an assumed model to reveal the key. As this is based on histograms or a kernel distribution, it also considers the whole distribution. In addition to leakage detection Moradi et al. also proposed a distinguisher based on the χ^2-test, which tests whether the groups defined by a model are independent and thus can also reveal the keys. In combination with modern measurement methods like on-die EM measurements [6,9] which can monitor the leakage of the cryptographic core independently of the surrounding circuitry, a low noise measurement might defeat a masking implementation when targeting with the combined leakage of all orders.

1.1 Contribution

As both methods follow similar approaches and can actually be calculated on the same precomputed histograms, the question arises whether one of the methods shows an advantage over the other. To evaluate this, we first try to find a theoretical dependency between the two tests and show why these schemes are not directly related in the way the attacks are currently formulated. Further, we present two case studies which successfully exploit leakage in higher orders.

The first one is a threshold implementation of PRESENT implemented on an FPGA and the second an SCA-protected AES hardware implementation on a smart card.

2 Background

2.1 χ^2-Test and Distinguisher

Utilizing the χ^2-test to detect leakage independent of statistical moments was initially proposed by Moradi et al. in 2018 [12]. Further, they showed that it can also be used as a distinguisher similar to MIA.

χ^2-Test of Independence Pearson's χ^2-test of independence checks whether two random variables are independent. For two random variables X and Y, it tests whether

$$H_0 : Pr[X = x|Y = y] = Pr[X = x]$$

which means that they are independent. Let $P \in \mathbb{R}^{r\times c}$ be the matrix with p_{ij} standing for the joint probabilities $Pr[X = x_i \wedge Y = y_j]$ that X takes its i-th category *and* Y takes its j-th. If H_0 holds, the multiplication rule states that $Pr[X = x_i \wedge Y = y_j] = Pr[X = x_i] \cdot Pr[Y = y_j]$. So in a random experiment with N repetitions, the expected frequency that $X = x_i$ and $Y = y_j$ should be $N \cdot Pr[X = x_i] \cdot Pr[Y = y_j]$. Since $Pr[X = x_i]$ and $Pr[Y = y_j]$ are not known, they have to be estimated from the contingency table $F = (f_{i,j}) \in \mathbb{R}^{r\times c}$ where $f_{i,j}$ is the number of times that x_i occurred together with y_j. To estimate $Pr[X = x_i]$ we sum up all frequencies in the corresponding row and divide it by the total number of experiments N. To estimate $Pr[Y = y_j]$ we sum up all frequencies in the corresponding column and divide it by N. This results in the expected frequency being calculated as

$$e_{i,j} = \left(\frac{\sum_{k=0}^{c-1} f_{i,k}}{N}\right)\left(\frac{\sum_{k=0}^{r-1} f_{k,j}}{N}\right) \cdot N = \frac{\left(\sum_{k=0}^{c-1} f_{i,k}\right)\left(\sum_{k=0}^{r-1} f_{k,j}\right)}{N}. \tag{1}$$

Now the Z value is a metric of how much the actual frequencies $f_{i,j}$ differ from the expected ones $e_{i,j}$. It is computed in the same fashion as shown in Equation (2):

$$Z = \sum_{i=0}^{r-1}\sum_{j=0}^{c-1} \frac{(f_{i,j} - e_{i,j})^2}{e_{i,j}}. \tag{2}$$

With the degree of freedom as $v = (r-1)\cdot(c-1)$. The p-value

$$p = \int_Z^\infty \mathsf{f}(Z, v)\, dx, \qquad \mathsf{f}(Z, v) = \begin{cases} \frac{Z^{\frac{v}{2}-1}e^{-\frac{Z}{2}}}{2^{\frac{v}{2}}\Gamma\left(\frac{v}{2}\right)} & Z > 0 \\ 0 & Z \leq 0 \end{cases}, \tag{3}$$

with the gamma function Γ can be used as a metric in order to test H_0. It expresses the probability whether the null hypothesis is accepted. We use this test for specific leakage assessment and as a distinguisher in a Differential Power Analysis (DPA). More precisely, we test whether the distribution depends of the value of a hypothetical power model. In this case, X corresponds to the value measured by the ADC of the oscilloscope and Y corresponds to the value of the power model. It can also be used in addition to the non-specific t-test [12], where Y is simply 0 or 1, depending of whether a random plaintext or a fixed plaintext is used.

Distinguisher Very similar to the way the Pearson correlation coefficient is used as a distinguisher in a DPA/CPA, we may use the result of the χ^2-test. In DPA/CPA, it is assumed that for the correct key guess, the values of the power model correlates well with the actual power consumption. In χ^2-test the assumption is that for the correct key guess, the traces depend on the values of the power model, whereas the power consumption is independent of the model for wrong key guesses. For every key guess this can be checked with the χ^2-test of independence. Similar to the t-test, we can state with a certain confidence that the assumption H_0 is wrong, which in our case means that the values of the power consumption depends on the values of the model.

2.2 Mutual information Analysis

Mutual Information Analysis (MIA) for SCA was initially proposed by Gierlichs et al. in 2008 [7] as a more generic information-theoretic distinguisher.

Mutual Information The Mutual Information (MI) I is a measure for the information shared between two random variables X and Y.

As the entropy $\mathrm{H}(X)$ is a measure of the information contained in X we can substract the conditional entropy $\mathrm{H}(X|Y)$ as this is exactly the portion of the entropy which is not covered by Y to get the mutual information $\mathrm{I}(X;Y)$.

$$\begin{aligned} \mathrm{I}(X;Y) &= \mathrm{H}(X) - \mathrm{H}(X|Y) \\ &= \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y) = \mathrm{I}(Y;X) \end{aligned} \tag{4}$$

Suppose that X and Y are random variables of the discrete spaces $\mathcal{X}$ and $\mathcal{Y}$, we can also formulate the mutual information $\mathrm{I}(X;Y)$ as

$$\begin{aligned} \mathrm{I}(X;Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} & Pr[X=x, Y=y]\cdot \\ & \log_2\left(\frac{Pr[X=x,Y=y]}{Pr[X=x]Pr[Y=y]}\right), \end{aligned} \tag{5}$$

knowing the joint probability $Pr[X=x, Y=y]$ and the marginal probabilities $Pr[X=x]$ and $Pr[Y=y]$ which can be calculated from the contingency tables.

Distinguisher Considering this in the side-channel application, we set one variable as our observation of the side-channel and the other a model of the leakage depending on the secret key. Hence, we set X as our observation of the power consumption and Y as the assumed distribution of our leakage model. Hence, if the power consumption behaves similar to our leakage model, the mutual information increases. Only if the key is correct, our observations have the same or a similar distribution as the power model and thus a high mutual information.

2.3 Implementation of χ^2-Test and MIA

The first step needed is the calculation of the histograms for the different models and key candidates. For each key candidate the traces are grouped by the key dependent model. The histograms are then calculated for each group and each point in time. Here, the original oscilloscope quantization (8-bit values as the result of its Analog-to-Digital Converter (ADC)) is kept and the results saved as starting point for the different metric calculations. When using the same model this can be performed as a common precomputation step for χ^2-test and MIA, as both methods can perform their computation on the same histograms.

Based on these basic histograms the next step differs for the two methods. The χ^2-test has a fixed rule for handling empty bins by ignoring them. This can result in different numbers of bins for each point in time. But since we use the p-value for comparison of the candidates, this does not matter as it is accounted for in the degrees-of-freedom v (see Equation (3)). For MIA the original histograms have to be rebinned. The lowest (highest, respectively) bin contains the lowest(highest, respectively) value measured with the bins in between filled with the corresponding ratio of the original bin counts. Additionally, the number of bins has to be the same for each point in time to be able to compare these. At the same time, also the success of the attack highly depends on the choice of the number of bins as shown by Moradi et al. in [11]. Thus, MIA has to be performed multiple times with different parameters to find the best attack setting which results in a high overhead in comparison to the χ^2-test. These adjustments to the bins, are only performed in memory within the respective calculation.

2.4 Relation Between χ^2-test and Mutual Information

While at first glance MIA and the χ^2-test seem very similar and can show similar results (see Sections 3.1 and 4.6), they are based on different approaches. There are different tests which can be performed with a statistical measure like the χ^2-test. Two common ones to perform on data sets are the *test of goodness-of-fit* and the *test of independence* which was already introduced in Section 2.1.

The *test of goodness-of-fit* examines whether the contingency table based on an observation f_i of a random variable fits the expected occurrences e_i of a theoretical model. In contrast to the *test of independence* the expected values e_i are given by a theoretical model and not by the observations.

For the same applications as the χ^2-test there is the G-test, which can be used as an alternative for *test of goodness-of-fit* and *test of independence*. It

is also based on histograms/contingency tables and also approximates the χ^2 distribution.

$$G = 2 \sum_{\forall i} f_i \cdot \ln \left(\frac{f_i}{e_i} \right) \tag{6}$$

As shown in [13] Equation (6) can also be expressed in terms of probabilities $p_i = \frac{1}{N} \sum_i f_{i,j}$ and $p_{i,j} = \frac{1}{N} f_{i,j}$ with N the total number of observations.

$$G = 2N \sum_{\forall i, \forall j} p_{i,j} \cdot (\ln(p_{i,j}) - \ln(p_i) - \ln(p_j)) \tag{7}$$

By considering the definition of entropy $\mathrm{H}(x) = -\sum_{x \in \mathcal{X}} p(x) \cdot \ln p(x)$ and joint entropy $\mathrm{H}(x, y) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \cdot \ln p(x, y)$ based on the natural logarithm and distributing the sum to the sub-terms we can further express G through entropies and following from these mutual information.

$$G = 2N \left(\mathrm{H}(x) + \mathrm{H}(y) - \mathrm{H}(x, y) \right) = 2N \cdot \mathrm{I}(x; y) \tag{8}$$

Please note that the entropy and mutual information in computer science are usually calculated with $\log_2$ which introduces an additional factor of $\frac{1}{\ln 2} \approx 1.443$ in Equation (8).

Based on this, there should only be a constant factor between mutual information and the G value. As G-test and χ^2-test both approximate the χ^2-distribution, these should lead to the same results for reasonable sample sizes. So, there seems to be a connection between χ^2 and mutual information. However, the two tests are currently used in different approaches. For the χ^2-test as presented in [12] we split up the observations into sets based on a model and then perform a test of independence, i.e. whether the distributions of the sets are independent. In contrast, for MIA we calculate the mutual information between our model and the observations which is more like a test of goodness-of-fit, i.e. whether the observed distribution is similar to the theoretical model. Accordingly, the relation via the G-test does not apply. Since we cannot give a direct theoretical connection we further evaluate their behavior by two case studies.

2.5 Higher-Order CPA

CPA as introduced by Bier et al. [2] uses the Pearson correlation coefficient between measurements and hypothetical leakage to extract the secret key. The hypothetical leakage is calculated for each challenge using a key dependent model.

In order to attack masked implementations, it is possible to perform a univariate (i.e., every point in time is considered individually) CPA at higher orders by preprocessing the measurements. To this end, the point-wise mean is subtracted from the measurements t and the results are taken to the power of the order d as $t' = (t - \bar{t})^d$. It is shown by Schneider et al. [20] in 2016 how to efficiently perform these computations.

3 Case Study 1: PRESENT Threshold Implementation

Our first case study is a threshold implementation of the PRESENT cipher as presented in [18]. For better comparison we evaluate the same implementation used by Moradi et al. in [12]. To achieve first-order security the state of the cipher is split into three Boolean shares (x_1, x_2, x_3) where $x = x_1 \oplus x_2 \oplus x_3$. It is saved in three 16-by-4 bit shift registers and from there shifted 4 bits per clock cycle. After key addition, the state is shifted into the S-box which is split up into two functions G and F separated by a register. The S-box lookups are then run as a pipelined serial computation which takes 17 clock cycles with the PLayer run in parallel in one clock cycle after the S-boxes.

The design is implemented on a Xilinx Spartan-6 FPGA on a SAKURA-G board [1]. To collect the measurements, we used the integrated amplifier of the SAKURA-G board and sampled the power consumption at a sampling rate of 1 GS/s. The core was running at a frequency of 160 MHz and the initial sharing performed in the control FPGA to prevent leakage from the inputs, i.e. the target FPGA receives masked plaintext and issues masked ciphertext.

Performing a random-vs-fixed (non-specific) t-test on the traces revealed minimal leakage ($t = 8.2$) in the second and significant leakage ($t = 39.55$) in the third order using 50,000,000 traces. Since we attack the first round of the encryption and the major leakage is right at the beginning of the measurement, we only consider the first 500 ns for the attack.

3.1 Results

As we are analyzing a nibble-serial implementation, we chose the Hamming distance of two consecutive S-box lookups as our power model $\mathrm{HD}(S(p_i \oplus k_i), S(p_{i+1} \oplus k_{i+1}))$. It results in 8-bit key candidates, as it is based on the distance between two consecutive nibbles. To decrease the complexity of the attack, we can assume that we perform the 8 bit attack only for the first distance, continuing from there we always already know one of the two key nibbles and can work with 4 bit candidates. To find the optimal number of bins for the MIA, we first tested which settings lead to the best result for the nibble and then performed the attack with this optimal number of bins.

We performed a key recovery on one of the key nibbles with χ^2-test and MIA using 50,000,000 traces. Figure 1 (a) and (c) show the χ^2-p-value and the mutual information after all traces were processed. Both attacks are successful and the correct key can be clearly distinguished. As both methods use the same model they highlight the correct key at the same point in time but the period during which the correct key stands out is longer for MIA. However, correct key and wrong candidates are more clearly separated in the χ^2-test. This is also confirmed by Figures 1 (d) and (b) which plot the maximum MI (p-value, respectively) over the number of traces in the calculation. The χ^2-test needs 30,000,000 traces for the key to stand out while MIA needs 36,000,000 traces for the correct key to be more likely than the ghost peak.

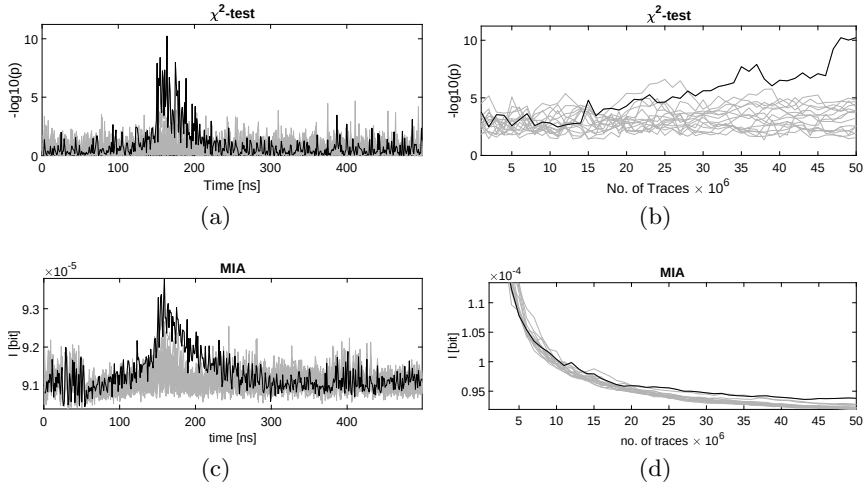


Fig. 1: Results of χ^2-test and MIA on PRESENT threshold implementation.

We also performed CPAs from the first to the third order but were not able to recover the key. We therefore omitted the figures. This indicates that combined leakage in higher order can indeed be better exploited in our case by moment-free methods.

4 Case Study 2: Smart Card

Our second target is a commercially available smart card implementing the Java Card standard with multiple cryptographic hardware cores. In this case study we target the AES encryption which is implemented by a dedicated circuit of the card.

4.1 Measurements

We performed on-die near-field EM measurements on the backside of the die, exposing it by removing the center pad of the smart card contacts and the underlying material.

For the measurements we used a Teledyne-Lecroy Waverunner 8254M with a sampling rate of 5 GSamples/s and the full bandwidth of 2.5 GHz. This high sampling rate and bandwidth is needed since the on-die EM signal includes sharp peaks reflecting the high frequency of the signal. As the EM probe we used a Langer EMV ICR HH150-27 near-field microprobe with a diameter of 150 µm and a bandwidth of 1.5 MHz to 6 GHz.

To find the optimal position on the die we scanned over the die and by visual inspection chose a position which showed a characteristic round pattern of the AES encryption.

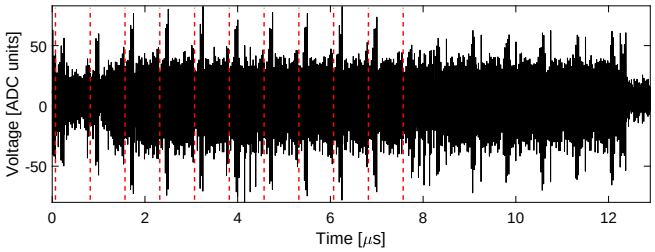


Fig. 2: Mean trace of 2000 aligned traces with clearly visible round structures.

4.2 Architecture

Based on our measurements we were able to identify the rounds of the AES implementation. The rounds are formed by the repeating pattern in Figure 2 of approx. 0.75 µs length. The rounds are marked by red lines in Figure 2. We confirmed this using correlation on the key schedule, which is executed at every round. Each round needs 25 clock cycles including the key schedule of 7 clock cylces to complete. The three high peaks within the round pattern are the end of the key schedule. Interestingly, the last round seems to be not shorter than the other ones although the MixColumns operation is missing in the last round of the AES algorithm.

4.3 Countermeasures

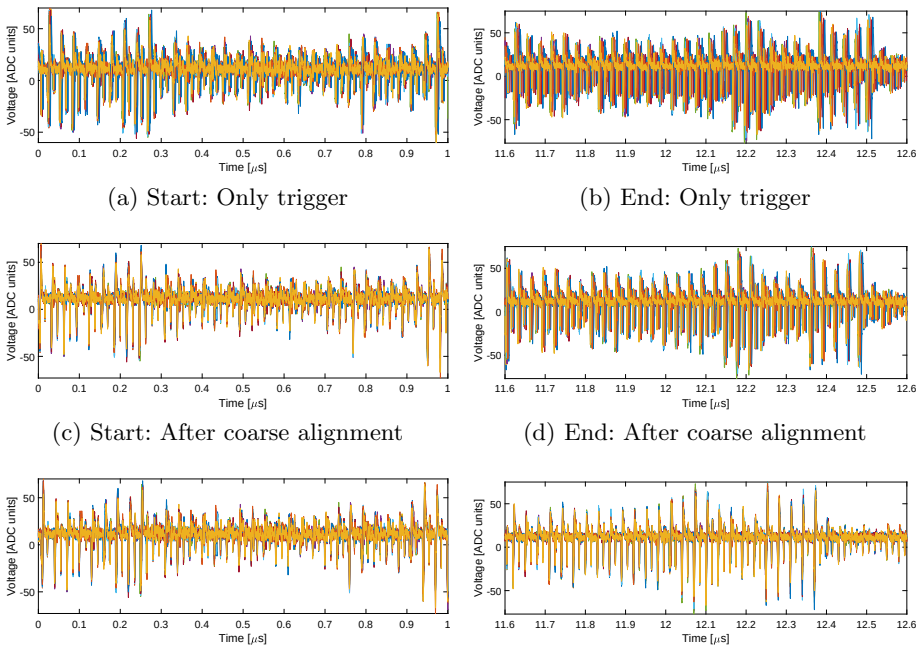
While it is based on a smart card IC which is also used in Common Criteria certified software and hardware combinations, the Java card we are attacking is not certified and most likely does not implement all countermeasures which are included in a certified product. Still, we expect that the circuit realizes hardware countermeasures.

A visual inspection of multiple measurements reveals strong random delays of the encryption in relation to the communication and additional high jitter of the clock. Also, we were able to get first order correlations on some plaintext bytes at the beginning of the realigned traces but no first order correlation on intermediates of the first round (see Section 4.6). Due to this low first order leakage, we believe that the card also incorporates some kind of masking countermeasures.

From a certified product one would expect additional countermeasures. A typical one would be dummy rounds, which we can exclude here since the traces clearly show 10 rounds, also the leakage of single rounds occurs only within one round pattern and not distributed over multiple ones.

4.4 Alignment

As we already observed random timing and jitter countermeasures we first need to align the traces. In the following, we explain how we did this to achieve the aligned mean trace shown in Figure 2.



(a) Start: Only trigger

(b) End: Only trigger

(c) Start: After coarse alignment

(d) End: After coarse alignment

(e) Start: After peak alignment

(f) End: After peak alignment

Fig. 3: Beginning and end of ten traces each after the three different stages of alignment.

Trigger on high peak To compensate for the long random delays, we used an advanced trigger setting making use of both IO communication signals of the card and the EM signal itself. This approach is possible since we are performing a localized EM measurement which exhibits the highest amplitudes when the encryption is running. Using such a trigger results in traces with only small temporal variation of the beginning of the encryption block as shown in Figure 3 (a) and 3 (b).

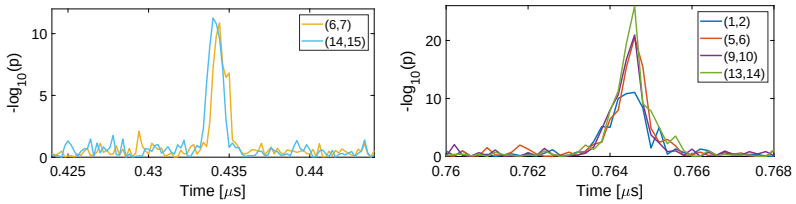
Coarse Alignment of AES Block We selected a reference pattern at the beginning of the first trace. To recover the offset of the other traces, we then correlated the pattern over a window at the beginning. Shifting the traces by the found offset results in Figure 3 (c) and (d). The beginning is now aligned but due to the clock jitter, the difference between the traces increases to the end.

Fine Alignment Against Clock Jitter To overcome the clock jitter, we followed a windowing approach. For each clock cycle in each trace we searched for the minimum and selected a window around it. By only keeping the part of the trace belonging to the windows of the 150 clock cycles, we created traces

whose clock cycles are aligned. As shown in Figure 3 (e) and (f) all peaks are now aligned at the beginning as well as at the end.

4.5 Key Recovery

After performing the alignment, we were able to conduct an attack on a subset of the key bytes. To this end, we used a Hamming distance (HD) model between outputs of the S-box operation $\mathrm{HD}(S(p_i \oplus k_i), S(p_j \oplus k_j))$. We found certain pairs $(i, j) = \{(1, 2), (5, 6), (9, 10), (13, 14), (6, 7), (14, 15)\}$ which lead to successful key recovery. As the model targets the distance between two bytes the size of the key candiates is 16 bits for the first four pairs and 8 bits for the last two ones since one of the bytes is already known from a previous pair. Interestingly, the first four pairs resemble the byte-wise distance between the second and third row of the AES state matrix.



(a) (b)

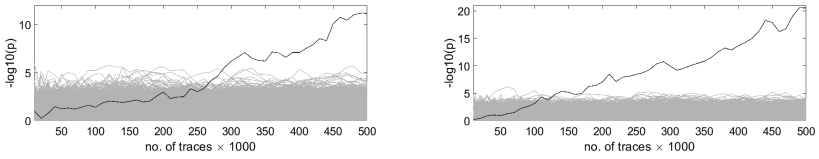
Fig. 4: Results of χ^2-test over time (a) and zooms of the two peaks we chose for the attack (b) and (c).

The results of the χ^2-test can suffer from outlier categories with only low counts. To prevent this from influencing the analyses, we modified the initial model. Instead of using the normal Hamming distance to categorize the traces, we merged the less frequent HDs which results in the following five categories $[\{0, 1, 2\}, 3, 4, 5, \{6, 7, 8\}]$. In the following we denote this grouping as HD′.

Considering the leakage of the different pairs over time we observed that that the pairs $\{(1, 2), (5, 6), (9, 10), (13, 14)\}$ leak at three different times while pairs $\{(6, 7), (14, 15)\}$ only leak at one point. For the attack we chose the peaks with the highest p-value for the respective pairs which are shown in more detail in Figure 4 (a) and (b).

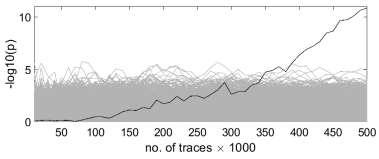
Figure 5 shows the progress of the attack for the different pairs of key bytes. The pair $(13, 14)$ shows the highest probability and can be recovered with less than 200000 traces. With 350000 required traces pair $(6, 7)$ is the most difficult to recover.

Using 350,000 traces only 6 out of 16 key bytes remain unknown with this attack. The remaining 48 bits of entropy are within brute-force range even without specialized hardware.

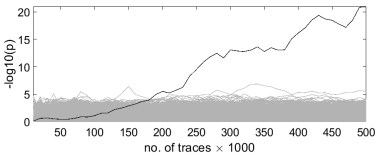


(a) $\mathrm{HD}'(S(p_1 \oplus k_1), S(p_2 \oplus k_2))$

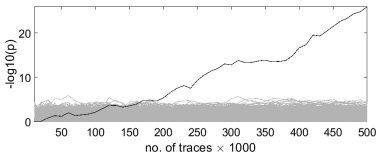
(b) $\mathrm{HD}'(S(p_5 \oplus k_5), S(p_6 \oplus k_6))$



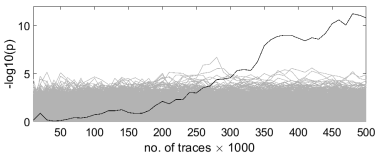
(c) $\mathrm{HD}'(S(p_6 \oplus k_6), S(p_7 \oplus k_7))$



(d) $\mathrm{HD}'(S(p_9 \oplus k_9), S(p_{10} \oplus k_{10}))$



(e) $\mathrm{HD}'(S(p_{13} \oplus k_{13}), S(p_{14} \oplus k_{14}))$



(f) $\mathrm{HD}'(S(p_{14} \oplus k_{14}), S(p_{15} \oplus k_{15}))$

Fig. 5: Progress of the attack results with the χ^2-distinguisher. Correct key highlighted in black.

4.6 χ^2-Test vs. MIA vs. HOCPA

In order to compare the different attacks we ran a χ^2-test, a MIA and CPAs from the first to the third order. To speed up the analyses we only used an 8-bit candidate and a small window of the traces. As the target, we picked the pairs $(13, 14)$ and $(6, 7)$ which are the ones requiring the least and most number of traces to succeed.

The CPAs at 1st to 3rd order were not successful in recovering the secret. Further, we used the aforementioned HD' model for all attacks. Since CPA needs a linear dependency between the power model and the measurements, we also examined the normal HD model but the attacks at all three orders were still not successful.

In contrast, χ^2-test and MIA were both able to recover the keys. For the pair $(13, 14)$ (shown in Figure 6) both attacks represent a clear peak for the correct key candidate. The χ^2-test needs around 200,000 traces and MIA requires slightly more traces (230,000). The attack targeting the pair $(6, 7)$ show a different behavior shown in Figure 7. While it was the worst performing pair for the χ^2-test with 350,000 traces, it performs even better than the other pair with MIA. It needs only 180,000 traces to identify the correct candidate. Interestingly,

the optimal number of bins for MIA is very different for the two considered key pairs. While the first showed best results with rebinning to 33 bins, the second one was optimal with only 8 bins.

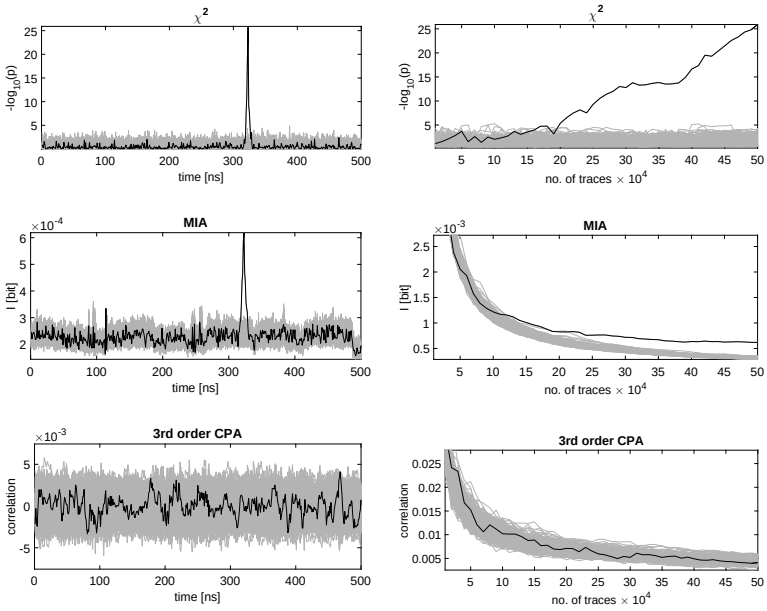


Fig. 6: Results of χ^2-test, MIA and 3rd order CPA attacks on key byte pair $(13, 14)$.

5 Conclusion

As explained in Section 2.4, for the current use of χ^2-test and MIA there is no direct relation. This is also shown in the two case studies presented here. While for the PRESENT TI the χ^2-test performed better, in the second case study we also presented an example in which MIA outperforms the other.

Independent of the presented results, there are differences in the application of the tests. While the computational effort needed to execute a single attack is similar for χ^2-test and MIA especially when using common histograms, the settings of MIA need to be optimized. In the histogram-based attacks an optimal number of bins has to be found for optimal results. This can result in the need to run the attack many times. The χ^2-test in contrast has defined rules how to handle empty bins. Thus, the χ^2-test might not necessarily be the best attack but it is easy to apply and does not need tuning, especially when already using it for leakage detection.

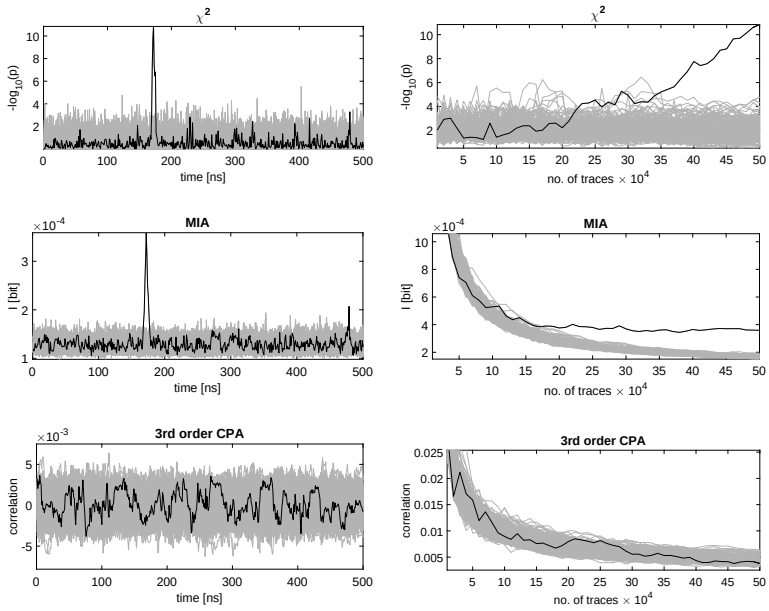


Fig. 7: Results of χ^2-test, MIA and 3rd order CPA attacks on key byte pair $(6, 7)$.

The presented second case study highlights the importance of thorough testing and certifying of cryptographic implementations. The common higher-order attacks (CPA) cannot reveal the secret while more sophisticated ones are able to do so. In case the underlying hardware AES implementation should be certified, such moment-free distinguishers also need to be examined.

Future Works. As the two analysis methods χ^2-test and MIA are currently used in different test types, it might be interesting to see how the χ^2-test performs in a *test of goodness-of-fit* scenario. Since the methods might converge differently, they still may lead to different results. It might also be interesting to see whether using the G-test instead of the χ^2-test leads to a faster key recovery.

Acknowledgments

This work is partly supported by the German Research Foundation (DFG) through the project 393207943 "Security for Internet of Things with Low Energy and Low Power Consumption (GreenSec)", and Germany's Excellence Strategy - EXC 2092 CASA - 390781972.

References

1. Side-channel AttacK User Reference Architecture. `http://satoh.cs.uec.ac.jp/SAKURA/index.html`

2. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: CHES. Lecture Notes in Computer Science, vol. 3156, pp. 16–29. Springer (2004)
3. Cnudde, T.D., Ender, M., Moradi, A.: Hardware masking, revisited. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(2), 123–148 (2018)
4. Dinu, D., Kizhvatov, I.: EM analysis in the iot context: Lessons learned from an attack on thread. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 73–97 (2018)
5. Eisenbarth, T., Kasper, T., Moradi, A., Paar, C., Salmasizadeh, M., Shalmani, M.T.M.: On the power of power analysis in the real world: A complete break of the keeloqcode hopping scheme. In: CRYPTO. Lecture Notes in Computer Science, vol. 5157, pp. 203–220. Springer (2008)
6. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: CHES. Lecture Notes in Computer Science, vol. 2162, pp. 251–261. Springer (2001)
7. Gierlichs, B., Batina, L., Tuyls, P., Preneel, B.: Mutual information analysis. In: CHES. Lecture Notes in Computer Science, vol. 5154, pp. 426–442. Springer (2008)
8. Goodwill, G., Jun, B., Jaffe, J., Rohatgi, P.: A testing methodology for side channel resistance validation. In: NIST non-invasive attack testing workshop (2011)
9. Heyszl, J., Mangard, S., Heinz, B., Stumpf, F., Sigl, G.: Localized electromagnetic analysis of cryptographic implementations. In: CT-RSA. Lecture Notes in Computer Science, vol. 7178, pp. 231–244. Springer (2012)
10. Moradi, A., Mischke, O.: On the simplicity of converting leakages from multivariate to univariate - (case study of a glitch-resistant masking scheme). In: CHES. Lecture Notes in Computer Science, vol. 8086, pp. 1–20. Springer (2013)
11. Moradi, A., Mousavi, N., Paar, C., Salmasizadeh, M.: Information security applications. chap. A Comparative Study of Mutual Information Analysis Under a Gaussian Assumption, pp. 193–205. Springer-Verlag, Berlin, Heidelberg (2009)
12. Moradi, A., Richter, B., Schneider, T., Standaert, F.: Leakage detection with the x2-test. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 209–237 (2018)
13. Morris, A.: An information theoretic measure of sequence recognition performance. Tech. rep., IDIAP (2002)
14. Nikova, S., Rijmen, V., Schläffer, M.: Secure hardware implementation of nonlinear functions in the presence of glitches. J. Cryptology **24**(2), 292–321 (2011)
15. Oswald, D., Paar, C.: Breaking mifare desfire MF3ICD40: power analysis and templates in the real world. In: CHES. Lecture Notes in Computer Science, vol. 6917, pp. 207–222. Springer (2011)
16. Oswald, D., Richter, B., Paar, C.: Side-channel attacks on the yubikey 2 one-time password generator. In: RAID. Lecture Notes in Computer Science, vol. 8145, pp. 204–222. Springer (2013)
17. Oswald, D., Strobel, D., Schellenberg, F., Kasper, T., Paar, C.: When reverse-engineering meets side-channel analysis - digital lockpicking in practice. In: Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 8282, pp. 571–588. Springer (2013)
18. Poschmann, A., Moradi, A., Khoo, K., Lim, C., Wang, H., Ling, S.: Side-channel resistant crypto for less than 2, 300 GE. J. Cryptology **24**(2), 322–345 (2011)
19. Prouff, E., Rivain, M., Bevan, R.: Statistical analysis of second order differential power analysis. IEEE Trans. Computers **58**(6), 799–811 (2009)
20. Schneider, T., Moradi, A., Güneysu, T.: Robust and one-pass parallel computation of correlation-based attacks at arbitrary order. In: COSADE. Lecture Notes in Computer Science, vol. 9689, pp. 199–217. Springer (2016)