Slicing Up a Perfect Hardware Masking Scheme

Zhimin Chen
Electrical and Computer Engineering Dept.
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
chenzm@vt.edu

Patrick Schaumont
Electrical and Computer Engineering Dept.
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
schaum@vt.edu

***Abstract*—Masking is a side-channel countermeasure that randomizes side-channel leakage, such as the power dissipation of a circuit. Masking is only effective on the condition that the internal random mask remains a secret. Previous research has illustrated how a successful estimation of the mask bit in circuit-level masking leads to successful side-channel attacks. In this paper, we extend this concept to algorithmic masking, which uses multi-bit masks. Our key observation is that the power dissipation of a masked circuit and the mask value are not independent. We exploit this property by using a slice of the power samples obtained by partial selection. This slice has a statistically biased mask, even when the mask signal itself is generated with a uniform distribution. We demonstrate this attack by showing how a perfectly masked AES SBox can be broken using part of the observed power samples, while the same circuit remains secure if we use all of the observed power samples.**

Keywords-Side-channel Attack; Differential Power Analysis masking; slice

I. Introduction

Differential Power Analysis (DPA) [1] is a side channel attack technique that uses statistical analysis methods to infer internal secret circuit nodes, such as secret keys, from externally observable circuit properties such as the overall power dissipation. The basis of a DPA is the statistical correlation between the processed data and the power dissipation. Masking is a popular countermeasure technique [2, 3, 4, 5] that eliminates this correlation. It makes use of random numbers, called masks, to randomize the internal circuit nodes. This makes side-channel leakage through power dissipation harmless as long as the mask value remains unknown. For this reason, mask signals are internally generated, near the circuit that they protect. Mask signals need to remain secret for the attacker.

There are two approaches to masking: Boolean masking, which works at the bit-level, and algorithmic masking, which works at the word-level. Boolean masking can be formulated as a systematic circuit-level transformation, with logic styles such as Random Switching Logic (RSL) [6] and Masked Dual-rail Pre-charge Logic (MDPL) [7] as two examples. The random mask in these circuits consists of a single bit. In contrast, algorithmic masking operates at word-level and uses word-level masks (multi-bit). This method aims to transform the operations of a cryptographic algorithm such that every internal value is statistically independent of the input and output of that algorithm. If this statistical independence is achieved, we call it a perfect masking scheme [8]. Special care must be taken when masking non-linear functions, since no systematic masking transformation is known for these. Researchers already presented several solutions for such non-linear modules, for example the AES SBox [9].

Recently, it was shown that a Boolean masking scheme can be broken by first estimating the mask bit and then mounting a DPA based on these estimates [10, 11]. These attacks were demonstrated on single-rail as well as dual-rail masked logic, and they demonstrated a practical implementation of the so-called zero-offset second-order DPA [12].

Our understanding is that, in a perfect masked circuit, the mask should be treated as a secret. Further, if we consider the mask as a random variable, it should be uniformly distributed and unbiased. Indeed, it is known that mask bias can enable a direct DPA attack, although no practical results have been demonstrated outside of software-based masking on microcontrollers [13]. In this contribution, we focus on hardware circuits. We show that, in existing hardware masking methods, the power probability density function (PDF) of the power dissipation is not independent of the mask. This leaves a backdoor to estimate the secret mask value. By selecting a slice of the PDF (a partial selection of power samples whose power value fall into a particular range), a bias is introduced in each mask bit (meaning that the probability of this mask bit being 1 is different from 0.5). We will demonstrate that such a sliced PDF can be successfully used in a DPA. Our attack is, similar to [10, 11], a high-order attack. However, in contrast to [10, 11], we are able to attack multiple mask bits at the same time, also, our assumptions on the impact of masking on the power probability distribution are more general.

This paper is organized as follows. The basic ideas of algorithmic masking are reviewed in Section 2. In Section 3, we analyze previous work on Boolean masking attacks and extend the idea to algorithmic masking. To demonstrate our analysis, we present a simulation-based DPA in Section 4. Finally, Section 5 concludes the paper.

II. Algorithmic Masking

In CMOS circuits, logic-0 and logic-1 are represented by different voltages, which results in different power dissipations. Therefore, the power dissipation of the circuit is data dependent. By capturing the power dissipation of the complete circuit as a time-trace, the resulting signal contains information from all internal circuit nodes. DPA is a statistical method to filter out the information of interest (such as a secret key) from the overall power signal.

Algorithmic masking is a countermeasure that converts deterministic internal logic values into random ones. The basic concept of algorithmic masking is described by Equation 1.

$$o = f(a) = f_1(a \oplus m) \oplus f_2(m)\ . \qquad (1)$$

In Equation 1, *f(a)* represents a cryptographic module with unmasked input *a*. We create a masked version of this module by introducing a random mask *m* and by partitioning the original function into two sub-functions f_1 and f_2 which process the masked signal $a \oplus m$ and the random mask *m* respectively. The decomposition of *f* into f_1 and f_2 depends on the original cryptographic module. This is a design problem in itself, in particular when *f* contains non-linear terms. Nevertheless the decomposition has been demonstrated to be feasible [9]. It has also been demonstrated that if the mask *m* is unbiased and independent from the input *a*, the masked circuit's power dissipation becomes independent from the unmasked data *a*. An unbiased mask means that *m* has a uniform distribution.

In this paper, we take a masked AES SBox [9] as an example. The main component of the SBox, as illustrated in Figure 1, is a masked GF(256) inversion. We can see this block is an application of Equation 1. It takes $a \oplus m$, *m* and *fm* as the input. The input $a \oplus m$ is the masked value of *a*, *m* is the mask, and *fm* is called a fresh-mask, and its purpose is to mask intermediate results within the inversion block. The masked inversion is implemented using a combination of modular addition and modular multiplication. Implementation details can be found in [9].

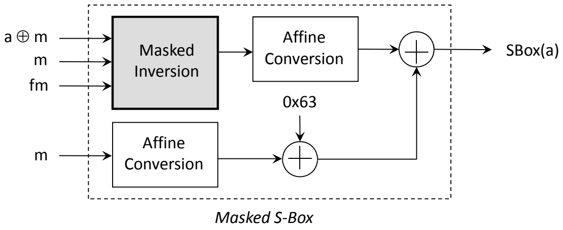

Figure 1. Algorithmic Masking on AES SBox.

III. An Attack by Slicing the Power Samples

Let's call *p* the power dissipation of a masked circuit, which is the power from $f_1(a \oplus m) \oplus f_2(m)$ in Equation (1). For a perfect masked circuit, the correlation between *p* and the unmasked input *a* must be zero. Indeed, if there would be any correlation left, then it would mean the masked circuit is still susceptible to DPA. So we can write

$$corr(p,a) = 0 \qquad (2)$$

However, masked circuits are designed without considering the effect of the mask *m* on the power. In general, the power dissipation will still be correlated to the mask *m*. This is considered harmless because *m* is a random number, without useful information to the attacker. But if *p* and *m* are correlated to some extent, this means that

$$corr(p,m) \neq 0 \qquad (3)$$

As we discussed before, the mask *m* should remain as a secret to maintain full side-channel resistance. Observing (3), the question which comes to mind is: can we make use of the power *p* to estimate the mask *m*? Indeed, this question has been answered in a positive manner for the specific case of single-bit Boolean masking schemes [10, 11]. However, there is no reason why this should be limited to single-bit masking. We will therefore derive a method that works on multi-bit algorithmic masking.

In the following, we treat the mask value *m* as well as the power dissipation *p* as discrete random variables. This assumption is valid if we approximate *p* for example with the total hamming weight of the nets of the circuit under consideration. We will further see how this can be generalized to measured power values with a continuous distribution.

We can write the joint probability of the circuit consuming power *p* with mask value m as

$$\Pr(Power = p, Mask = m) = \Pr(p,m) \qquad (4)$$

Since the mask has a uniform distribution, the marginal probability of *m* should be a constant. The marginal probability of *m* is found by summing out the joint probability over *p*.

$$\Pr(m) = \sum_p \Pr(p,m) = Const \qquad (5)$$

The conditional probability of the power can now be found as follows.

$$\Pr(p|m) = \frac{\Pr(p,m)}{\Pr(m)} \qquad (6)$$

The conditional probability in (6) is the probability that the power dissipation of the circuit is *p*, given that the mask equals *m*. Note that we indicated earlier that the power is correlated to the mask. This means that the conditional probability in (6) cannot be independent of *m*. Thus, if we consider a given power measurement p_1, then the conditional probability of this power measurement will change as the mask changes. We can express this as follows.

$$\Pr(p_1|m) = \frac{\Pr(p_1,m)}{\Pr(m)} \neq Const \qquad (7)$$

The inequality of (7) becomes an attack method when we express the conditional probability of *m*. Through Bayes' theorem we can write

$$\frac{\Pr(p|m)}{\Pr(p)} = \frac{\Pr(m|p)}{\Pr(m)} \qquad (8)$$

We can now write the inequality of (7) as follows. Assume that we have a given power measurement p_a and two possible (and different) mask values m_1 and m_2, then according to (7) and (8):

$$\Pr(m_1|p_a) = \frac{\Pr(p_a, m_1)}{\Pr(p_a)} \neq \Pr(m_2|p_a) = \frac{\Pr(p_a, m_2)}{\Pr(p_a)}\Bigg|_{m_1 \neq m_2} \qquad (9)$$

In other words, Equation 9 shows that one mask value, say m_1, is more likely to correspond to a given power measurement p_a than another mask value. Thus if we measure a given power level, then we can estimate with a better-than-random guess what the mask value would be. In practice, power values are measured as continuous quantities, and it is impossible to choose just one discrete power value. Therefore, to implement the test of Equation 9, we will choose *p* over a range of possible values, and all those *p* samples fall into this range build up the slice of samples we want. Accordingly, the inequality becomes a sum:

$$\sum_{p \in range} \Pr(m_1|p) \neq \sum_{p \in range} \Pr(m_2|p)\Bigg|_{m_1 \neq m_2} \qquad (10)$$

In the case of continuous power values (such as obtained through physical measurements), the summations in Equation 10 become integrals. Clearly, there is a limit to this inequality. When we would integrate the power over the full range of possible values, Equation 10 becomes Equation 5 and turns from an inequality into an equality. So the key is to perform a partial selection of *p* over the possible range of values.

We can now summarize our attack method as follows.

Step 1. Collect power traces from the masked cryptographic circuit, and establish the possible range of power values.

Step 2. Select a slice of the possible range of power values, and discard all measurements which fall outside this range.

Step 3. Perform a DPA on the set of measurements obtained through step 2.

In the next section, we will demonstrate this surprisingly simple technique by means of a simulated attack on a masked SBox.

IV. Experimental Results

In this section, we present experimental results based on logic-level simulation. The main idea of the experiment is to count the number of logic-1s in a circuit to estimate the power dissipation of a circuit. We call this a hamming-weight simulation. Using hamming weight to simulate the power dissipation is an approximation and also an easy way for analysis. Our purpose is to show that existing masked hardware circuits that successfully hide the unmasked data still show dependence between the power and the mask. Furthermore, we make use of this dependence to introduce mask bias and successfully mount a power attack.

First, we describe the circuit module used for test in this experiment. We use a masked AES SBox (already mentioned in Section 2) with a key addition as the design under test. The masking methodology is given by [9]. The structure can be found in Figure 2.

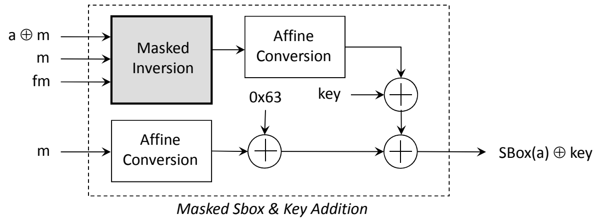

Figure 2. Design under Test.

We include an addition of a secret key value at the SBox output. The objective of our side channel analysis will be to find the secret key. In the simulation presented in this paper, the key is 35.

The simulation is based on the gate-level netlist of the above design. To get this netlist, we first implement the masking scheme with GEZEL [14]. Next, we convert the GEZEL code to VHDL code and do synthesis with Design Compiler. During synthesis, we must ensure that the XOR operations remain atomic elements. Therefore, the XOR operations are implemented in a separate hierarchy, and every XOR gate is a separate module. During synthesis, we then set a *don't-touch* attribute to the XOR modules, which prevents them from further logic expansion and/or optimization. The resulting synthesized VHDL netlist from Design Compiler is then converted back to GEZEL for hamming-weight simulation. As testbench, we exhaustively enumerate all three inputs from the masked SBox: the masked data input *ax*, the mask signal *m*, and the fresh-mask *fm*. For each triplet at the input, we obtain the hamming weight of the netlist, and we record the value of the SBox output. These data sets are next subjected to our Side Channel Analysis method.

Figure 3a shows the probability density function as the joint probability of the mask m and the power level p. Figure 3b is a detailed view of the elliptical area labeled in Figure 3a. Figure 3c illustrates the joint probability distribution of power *p* and mask *m* for two different values (128 and 135) of the mask.

Actually, Figure 3a and 3b are 3-d graphs with power, mask, and joint probability labeled on 'X-axes', 'Y-axes', and 'Z-axes' respectively. The reason why only power and mask are shown here is that we draw the graphs from the perspective along with the Z-axes (XY view). The magnitude of probability is represented by the brightness. Figure 3c (XZ view) shows the joint probability distribution when m = 128 and m = 135. As we can see, the probability changes as the mask changes, which demonstrates the dependence between the power and the mask.

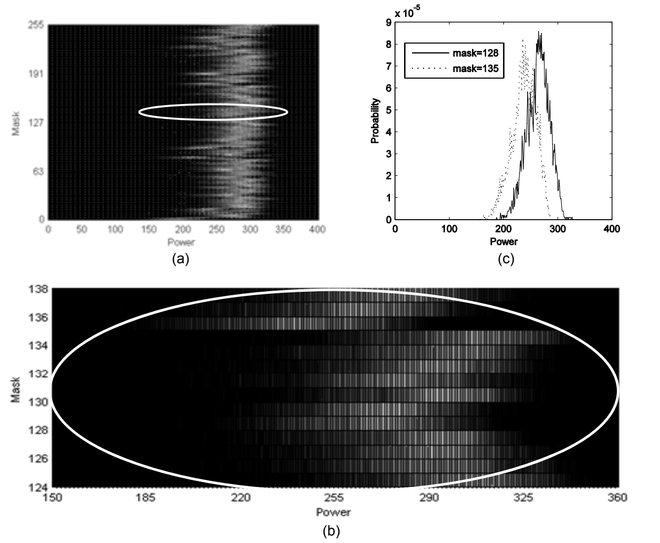

Figure 3. (a) Joint Probability Pr(p, m) for a masked AES SBox
(b) Detailed view of the elliptical area labeled in Figure 3a
(c) Joint probability Pr(p, m) when m = 128 and m = 135 illustrate dependence of the power on the mask.

The next step is to find a way to exploit biased mask. As was demonstrated with Equation 10, we can get a biased mask signal by selecting samples from a restricted range of power levels. This reduced set of samples can then be used for side-channel analysis. Of course, the range of power levels must be sufficiently large so that a DPA can still succeed. In our first attempt, we only included samples with a power level in the slice from [0:200] (See Figure 3a).

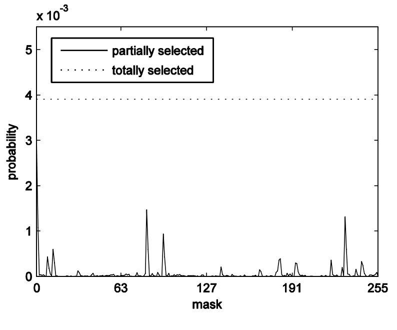

Figure 4. Total and partial probability mass function of the mask. The partial probability mass function is over selected range of power values ([0:200]).

We can calculate the conditional probability mass function (PMF) of the mask as a parameter of the power level range selected. (We use PMF here instead of PDF, since, in the simulation, the power dissipation is a discrete random variable.) The marginal probabilities for the mask values are illustrated in Figure 4. We included two different curves. The dotted line is the marginal probability when we choose to include all samples. As expected, the dotted line is constant, which means that we are using an unbiased mask signal. The solid line in Figure 4 represents the marginal probability of the mask signals for partially selected samples with a power level in the slice [0:200]. This line is not a constant, which means that the mask is biased.

We can further analyze the probability for each bit in the mask. This probability is represented in Figure 5. This figure illustrates that, for power samples in the slice [0:200], the probability for mask bits to be 1 is almost always less than 0.5. In other words, a reasonable estimate for the mask for all power samples in the slice [0:200] is an all-zero value. What should be mentioned is that the slice [0:200] is a simple choice but maybe not the optimal one. However, this example is enough to demonstrate our analysis in Section 3.

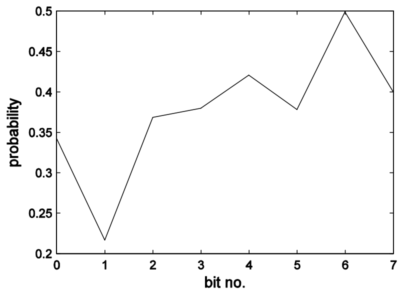

Figure 5. Resulting mask bias for the selected range of power values ([0:200]).

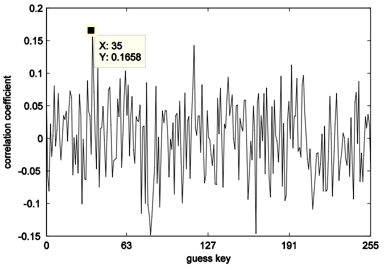

Figure 6. DPA correlation graph on signal with mask bias.

Finally, we can mount a DPA attack with this slice of power samples by assuming that the mask is always 0. While our guess for the mask is not deterministically correct, it is a correct guess from a statistical point of view. We have implemented the DPA based on the correlation of the hamming weight of unmasked input *a*. Figure 6 shows the resulting correlation graph. The highest peak (0.1658) appears at 'guess_key = 35' which is correct. Three other peaks can also be distinguished at 79 (-0.1476), 112 (0.1433), and 159 (-0.1459). In our simulated attack, guess_key 35 is the correct one. Figure 6 demonstrates how we successfully identified the correct key, even though the figure indicates that there are several other candidates with a correlation value close to the optimal one. The cause of this effect is that our mask estimation is obtained through a stochastic process. What's more, the fresh-mask *fm* is still unknown, this makes it harder to get a obvious peak. Still, we see that due to partial observation of the power samples, the key guess space due to masking is reduced from 256 possibilities to only 4. Right now the power model used for attack is very simple. As the model improves, the attack result should be better.

Conclusion

This paper illustrated how mask bias can be obtained for masked hardware circuits. The power dissipation of masked hardware circuits is uncorrelated to the unmasked data values, and therefore cannot be used for DPA. However, we showed that the power dissipation of a masked hardware circuit may still be correlated to the mask. Because of this correlation, it is possible to bias the mask by selecting only a small slice over the entire power probability density function. We applied this technique using an AES SBox with perfect masking. Using logic-level simulation, we demonstrated the dependence between the power dissipation and the mask value. By slicing the power PDF before mounting a DPA, we can bias each bit from the mask. In our case, we introduced a bias towards logic-0. Our conclusion is that hardware masking remains susceptible to direct DPA by making clever use of the power probability density function. In the future work, we will look into applying this kind of attack to a real circuit.

Acknowledgment

This work was supported in part by NSF Grant No. 0644070.

We would like to express our gratitude to Kris Tiri. His comments helped us much during writing the paper.

References

[1] P. Kocher, J. Jaffe, and B. Jun, “Differential Power Analysis,” In proceeding of Advances in Cryptology - CRYPTO ’ 99, pp. 388-397, Springer, 1999.

[2] S. Chari, C.S. Jutla, J.R. Rao, P. Rohatgi, “Towards Sound Approaches to Counteract Power-Analysis Attacks,” Proc. of Advances in Cryptology - (CRYPTO 1999), pp. 398-412, Springer, 1999.

[3] L. Goubin, J. Patarin, “DES and Differential Power Analysis - The “Duplication” Method”, Proc. of the Workshop on Cryptographic Hardware and Embedded Systems - (CHES 1999), pp. 158-172, Springer, 1999.

[4] M. Akkar, C. Giraud, “An Implementation of DES and AES, Secure against Some Attacks,” In Proc. 2001 Workshop on Cryptographic Hardware and Embedded Systems, (CHES 2001), LNCS 2162, p. 309-318, Springer, 2001.

[5] J. D. Golic and C. Tymen, “Multiplicative Masking and Power Analysis of AES,” In Proc. 2002 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002), LNCS 2523, p. 198-212, Springer, 2002.

[6] D. Suzuki, M. Saeki, T. Ichikawa, “Random Switching Logic: A New Countermeasure against DPA and Second-order DPA at the Logic Level,” IEICE Trans. Fundamentals, Vol. E90-A, no 1, p. 160-168, Jan. 2007.

[7] T. Popp, S. Mangard, ”Masked Dual-Rail Pre-charge Logic: DPA Resistance without the Routing Constraints,” Proc. of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005), LNCS 3659, p. 172-186, August 2005.

[8] J. Blőmer, J, Guajardo, and V. Krummel, “Provably Secure Masking of AES,” Selected Areas in Cryptography 2005, LNCS 3357, p. 69-83, 2005.

[9] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, “A Side-Channel Analysis Resistant Description of the AES S-Box,” Fast Software Encryption 2005, LNCS 3557, p. 413-423, 2005.

[10] P. Schaumont and K. Tiri, “Masking and Dual-Rail Logic Don’t Add Up,” In Proc. 2007 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2007), LNCS 4727, p. 95-106, 2007.

[11] K. Tiri and P. Schaumont, “Changing the Odds Against Masked Logic,” Selected Areas in Cryptography 2007, LNCS 4356, p. 134-146, 2007.

[12] J. Waddle and D. Wagner, "Towards efficient second-order power analysis," In Proc. 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), LNCS 3156, pp.1–15, 2004

[13] S. Mangard, E. Oswald, and T. Popp, “Power Analysis Attacks – Revealing the Secrets of Smart Cards,” p. 250. ISBN 978-8-9810801-2-4.

[14] P. Schaumont, I. Verbauwhede, "A Component-based Design Environment for Electronic System-level Design," IEEE Design and Test of Computers, special issue on Electronic System-Level Design, 23(5), pp. 338-347, September-October 2006.