Improving Secure Hardware Masking Using an Equalization Technique

Zhimin Chen
Virginia Tech
Blacksburg, VA 24061, USA
chenzm@vt.edu

Patrick Schaumont
Virginia Tech
Blacksburg, VA 24061, USA
schaum@vt.edu

ABSTRACT

Hardware masking is a side-channel countermeasure technique that randomizes the internal values of a digital circuit. Masking requires the use of random signals, called masks, which need to remain a secret within the circuit implementation. In currently known masking techniques, the circuit power consumption and the mask value are correlated. Using this correlation in a so-called higher-order attack, an attacker may be able to remove the effect of the mask value, and subsequently break the masking scheme. In this paper, we present a technique to reduce the correlation and to mitigate attacks on masking. The strong point of our approach is its simplicity. Small, functionality-preserving modifications to the netlist of a circuit are sufficient to make the attack fail. We demonstrate our technique on bit-level masked digital logic, and show how it can be extended to more complex cases.

1. INTRODUCTION

Side-channel attacks are passive, non-intrusive attacks on the trustworthy operation of a secure embedded system. An attacker measures externally observable properties of the system, and uses those measurements to infer the internal circuit state. Several statistical techniques have been proposed to process the measurement data, including simple power analysis (SPA), differential power analysis (DPA) [1], and templates [2]. These attack techniques differ by the amount of measurements needed and the required amount of a priori knowledge on the circuit internals.

In hardware implementations, side-channel resistant logic is able to mitigate side-channel attacks, at least partially. Side-channel resistant logic removes the correlation between the circuit power consumption and the internal circuit state, so that power measurements are no longer useful for side-channel analysis. The two techniques for side-channel resistant logic are hiding (making the power consumption constant) and masking (making the power consumption random) [3]. In this paper, we are proposing an improvement to hardware masking.

Hardware masking uses random mask signals to transform data signals into masked signals, which are uncorrelated to the original data. The circuit operations are implemented in terms of these masked signals. Previous research has demonstrated that masking is possible at the gate-level [4][5] as well as at the operation-level [6][7]. There are two known security risks for hardware masking: circuit-level timing issues, and higher-order attacks. This contribution will focus on low-cost mitigation of higher-order attacks; some solutions against circuit-level timing issues can be consulted in [8][9][10].

Higher-order attacks remove the effect of masking by combining multiple measurements that share a common mask signal [11]. Because the mask value is shared, the *joint* distribution of these measurements is no longer independent of the unmasked data signals. In a higher-order attack, the masking effect is deteriorated based on this joint distribution, and the circuit can be analyzed in a classic SPA/DPA.

Higher-order attacks on hardware are easy because each side-channel measurement reflects information for the *entire* circuit. Thus, if two masked circuit nodes use the same mask value, we can directly obtain the joint distribution for these nodes from the overall power consumption. A framework for higher-order attacks on masked hardware was presented by Waddle in [12]. Since then, practical implementations have been presented on several different forms of circuit-level and operation-level masking [13][14][15].

The traditional mitigation of higher-order attacks is to use multiple, independent mask signals. However, this becomes expensive very quickly. In the limit, this would require a unique mask signal for *each* circuit node that needs to be masked, which represents a significant circuit cost. Instead, we propose a different approach: we present a simple, functionality-preserving transformation on the netlist of a masked circuit so that the resulting power traces are harder to use in a higher-order attack. We call our technique an equalization technique because of its effect on the probability mass function of the power consumption.

The paper is structured as follows. In the next section, we describe a standard masking technique called Boolean masking, and we describe the effects of masking on the power probability density function. This will provide insight to the equalization technique described in Section 3. In Section 4, we present the results of our method by analyzing the side-channel resistance of a masked circuit, and in Section 5 we conclude the paper.

2. BOOLEAN HARDWARE MASKING

In this section, we analyze the power probability mass function (PMF) (power here is represented by integers) resulting from Boolean Hardware Masking. This analysis will then lead to the equalization method.

2.1 Single-bit masking

First, consider the masking of a single bit wire *a*.

$$a_m = a \oplus m \tag{1}$$

In this equation, a_m is the masked data wire, a is the unmasked data wire and m is the mask. For each new data value a, the mask m will take on a random value from {0,1}. A masked hardware circuit will process masked data a_m, and perform its intended operation without ever revealing the value of the unmasked data a. Clearly, if m is random, then a_m is uncorrelated to a. As a result, the power signature of a masked circuit is uncorrelated to the unmasked data a.

However, to obtain side-channel resistance, m needs to remain secret. A secret m implies that we cannot extract any information about a from observing a_m. For example, choosing a pseudo-random sequence for m is sufficient to obtain a statistically uniform distribution for a_m. However, if it is possible to predict this pseudorandom sequence, then we would still be able to obtain the trace of the unmasked data a. We also note that a perfect prediction of m is not required for a successful attack on a masked circuit - an estimate is sufficient. A correct *estimation* of m means that one is able to predict the correct value of m with a probability higher than 0.5.

In case of a single-bit masking scheme on a larger netlist, $\boldsymbol{a_m}$ and $\boldsymbol{a}$ become vectors, with each entry representing a wire of the circuit. In Boolean hardware masking, all wires use the same mask bit, so that we now may write

$$\boldsymbol{a_m} = \boldsymbol{a} \oplus m \tag{2}$$

For side-channel analysis applications based on energy, we are interested in the Hamming weight of $\boldsymbol{a_m}$, the number of '1' bits in $\boldsymbol{a_m}$. The Hamming weight *HW* is a useful model for an attack that uses the average power consumption over short periods, e.g. one clock cycle. We will now analyze the PMF of the Hamming weight of $\boldsymbol{a_m}$ for two different cases. Each case makes a different assumption on the PMF of the Hamming weight of $\boldsymbol{a}$. We assume that m is a uniformly distributed random signal, i.e. P(m=0) = P(m=1) = 0.5.

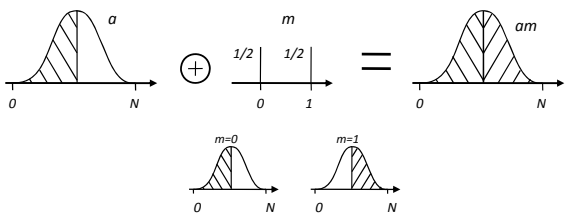

Figure 1: Effect of masking on the PMF of the Hamming Weight of a vector of uniform random signals.

2.2 Case 1: Masking a uniform random signal

Assume that the netlist has N wires, so that there are N signals to mask.

$$\boldsymbol{a} = \{a_0, a_1, ..., a_{N-1}\} \tag{3}$$

In this case, the lowest and highest Hamming weight for $\boldsymbol{a}$ is 0 and N respectively. If $\boldsymbol{a}$ is a uniformly random vector, then each of the elements in $\boldsymbol{a}$ is an independent random bit. In that case, the PMF of the Hamming weight of **a** is given by the well-known binomial distribution.

$$PMF_a(i) = \binom{i}{N} \tag{4}$$

The masked signal $\boldsymbol{a_m}$ is given by

$$\boldsymbol{a}_m = \{a_0 \oplus m, a_1 \oplus m, ..., a_{N-1} \oplus m\} \tag{5}$$

When the mask bit is 0, then $\boldsymbol{a_m}$ is equal to $\boldsymbol{a}$. When the mask bit is 1, then $\boldsymbol{a_m}$ contains the inverted bits of $\boldsymbol{a}$. This means that the PMF for HW($\boldsymbol{a_m}$) is related to the PMF for HW($\boldsymbol{a}$). The effect of masking on the PMF for HW($\boldsymbol{a_m}$) is graphically illustrated in Figure 1. The PMF of HW($\boldsymbol{a_m}$) is the combination of two copies of the PMF for HW($\boldsymbol{a}$), for *m=0* and *m=1* respectively. These sub-PMFs are mirror images of each other. Indeed, assume that the Hamming weight of $\boldsymbol{a}$, HW($\boldsymbol{a}$), would be k when *m=0*, then HW($\boldsymbol{a}$) for the same circuit state would be *N-k* when *m=1*. This is simply a consequence of single-bit masking, which flips all the bits of $\boldsymbol{a}$. However, since the unmasked PMF for HW($\boldsymbol{a}$) is already symmetric around *N/2*, the two sub-PMF perfectly overlap when forming the overall PMF for HW($\boldsymbol{a_m}$). Therefore, the PMF of HW($\boldsymbol{a_m}$) looks identical to that of HW($\boldsymbol{a}$.).

This perfect overlap is important to make single-bit masking effective as a side-channel countermeasure. To illustrate this, assume that we take a power sample of $\boldsymbol{a_m}$, for which HW($\boldsymbol{a_m}$) = j. In order to perform a successful side-channel analysis, we need to obtain an estimate for the unmasked Hamming Weight HW($\boldsymbol{a}$). However, it is impossible to make a correct estimate, since HW($\boldsymbol{a_m}$) = j means that, with equal probability, HW(**a**) = j or HW(**a**) =

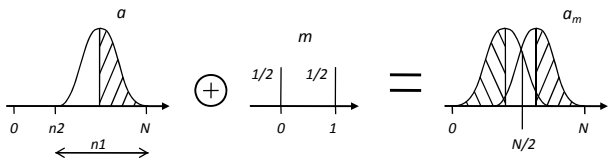

Figure 2: Effect of masking on the PMF of a vector of non-uniform random signals containing constant '1'.

N-j. Because of the symmetry, we can no longer make a better-than-random guess on the Hamming Weight of the unmasked data value. As a result, we can no longer extract any useful side-channel information from the power trace of the circuit.

Next, we will investigate the effect of masking on the PMF of non-uniform random signals.

2.2 Case 2: Masking a non-uniform random signal

We start from a netlist with N wires, so that there are N signals to mask. To create a non-uniform random distribution, we have to assume that some of these wires will not carry uniformly random data. There are many different possible distributions, and our analysis will therefore focus on a corner case: we will assume that some of the wires in the unmasked netlist do not change at all, but instead carry constants. This is not an overly artificial assumption. For example, [13] shows that the masked key schedule of a block cipher contributes a large number of constants. Let's assume that $\boldsymbol{a}$ contains *n1* random wires and *n2* wires which are tied to 1. $\boldsymbol{a}$ is given by

$$\mathbf{a} = \{a_0, a_1, ..., a_{n1-1}, \overbrace{1,...,1}^{n2}\} \quad with\ N = n1 + n2 \tag{6}$$

The PMF of *HW(**a**)* still has a binomial distribution. However, in this case the distribution does not start at the origin, but rather at *n2*. The width of the binomial is *n1*.

$$PMF_a(i) = \begin{cases} 0 \ when\ 0 <= i < n2 \\ \binom{i-n2}{n1} \ when\ \ n2 <= i < N \end{cases} \tag{7}$$

The effect of masking on the resulting distribution is illustrated in Figure 2. The same argument as before can be made. The overall PMF for HW($\boldsymbol{a_m}$) is the combination of the PMF for HW($\boldsymbol{a}$) when the mask bit is 1 and 0 respectively. When the mask bit is 1, all bits of $\boldsymbol{a}$ will be inverted, so the resulting hamming weight is mirrored around *N/2*. The result is that the PMF for HW($\boldsymbol{a_m}$) contains two distributions, which are non-overlapping (one is aligned to the origin, the second is aligned to N). If the proportion of constant signals *n2* would increase further, the separation would become even more pronounced. In the limit, assuming that *n2 = N*, the PMF of HW($\boldsymbol{a_m}$) would be identical in shape to the PMF of *m*. In that case, it would be possible for an outside observer to perfectly determine *m*.

Thus, from the perspective of side-channel analysis, the separation of the overall PMF of HW($\boldsymbol{a_m}$) into two smaller PMFs is a weakness. Indeed, if we can identify individual PMF, we are able to use the observed power level directly as an estimate for the mask bit. If the power level is smaller than a certain threshold value (equivalent to N/2), we assume that the sample must be part of the first PMF, otherwise, we assume it is part of the second PMF. This is the basis for the attack presented in [13].

We can evaluate the amount of separation of the two 'peaks' in the PMF of $\boldsymbol{a_m}$ as follows. The first peak is located at *n1/2 = (N-n2)/2*. The second peak is located at *N - (N-n2)/2*. The distance between both peaks is the difference of these two, which is *n2*. In other words, the amount of separation depends only on the number of constant signals in the netlist.

We can do this analysis also for a distribution of $\boldsymbol{a}$ that contains constant-1 wires, constant-0 wires, and random wires:

$$\boldsymbol{a} = \{a_0, a_1, ..., a_{n1-1}, \overbrace{1,...,1}^{n2}, \overbrace{0,...,0}^{n3}\} \quad (with\ N = n1 + n2 + n3) \tag{8}$$

Adding constant-0 wires to a netlist will not directly affect the PMF of the unmasked netlist, because the 0-wires do not contribute Hamming Weight. However, if we mask a netlist with constant-0 wires, then those wires will directly reflect the value of the mask bit, and thus affects the PMF of HW($\boldsymbol{a_m}$). The analysis of this case is similar to the one above, and we only summarize the conclusions. For a netlist with a distribution as given by (8), the PMF of HW($\boldsymbol{a_m}$) will contain two smaller PMF distributions which are mirrored around N/2, and which are separated by *abs(n2 - n3)*, i.e., the difference between the number of constant-1 and the number of constant-0 wires.

These results are for an idealized case with uniformly random wires and constant wires. In real circuits, there will be wires which are neither uniformly random nor constant. For example, the output of an gate which ANDs two uniformly random signals will be skewed so that the output contains a '1' on 1/4 of all the AND evaluations. Such skewed signals introduce asymmetry in the PMF of the unmasked netlist. Also asymmetry results in non-perfect overlap of the masked PMF, which, in turn, may enable a side-channel attack on the masked netlist.

In the following section, we describe a simple technique, and its associated implementation, to improve the overlap of the sub-PMF of the masked netlist. Using a low-cost preprocessing technique, we can improve the side-channel resistance of Boolean masking.

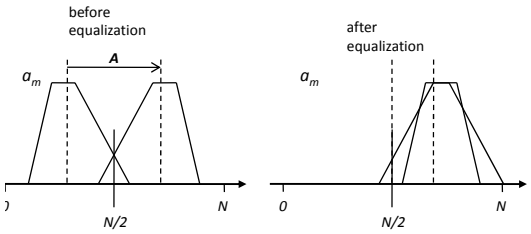

Figure 3: The objective of Equalization is to increase overlap in the PMF of HW($\mathbf{a_m}$).

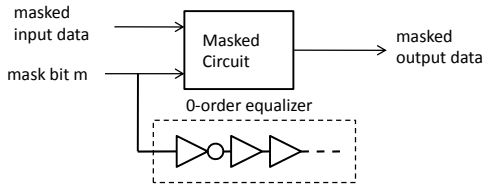

Figure 4: Example implementation for a zero-order equalizer circuit.

3. AN EQUALIZATION SCHEME

The idea of equalization is illustrated in Figure 3. As a result of Boolean masking, the PMF of the Hamming Weight HW(***a***) is mirrored around half of the total Hamming weight of the netlist. By equalizing a masked PMF, we intend to reduce the separation between the PMF for *m=1*, and the PMF for *m=0*. As shown in Figure 3, we do not make an assumption on the distribution of **a**. In Section 4, we will demonstrate that the PMF for actual circuits has a similar asymmetrical shape. We will present two strategies to achieve the effect of Figure 3: zero-order equalization and first-order equalization. We will also present sample implementations and the design steps needed to transform a masked netlist into an equalized masked netlist.

3.1 Zero-order Equalization

In zero order equalization, we aim to implement a linear shift of the PMF corresponding to a single mask bit value, as illustrated in Figure 3. A linear shift corresponds to modifying the Hamming weight of the entire netlist, which can be done by adding a net with specified Hamming weight (or loading) and with a specified value. The value and the weight of the net must be chosen such as to shift the lower PMF on top of the upper PMF. This reduces the distance *A* in Figure 3 to zero. A possible implementation of this technique is illustrated in Figure 4. In this example we assume that the lower PMF corresponds to a mask value of 0. We then feed the *inverse* of the mask bit in a chain of buffers, so that when the mask bit is 0, the Hamming weight of the overall circuit increases. When the mask bit is 1, the Hamming weight of the overall circuit is not modified. Note that it is also possible that the lower PMF corresponds to a mask value of 1. In that case, we need to feed the mask bit directly into the buffer chain, rather than inverting it first. It is easy to determine the required amount of capacitive loading. For example, we can simulate the masked netlist while tying the mask bit to 0, and determine the average Hamming weight of the circuit. Next, we perform the same simulation while tying the mask bit to 1, and determine again the average Hamming weight. The difference of these two weights is A, and this corresponds to the required loading.

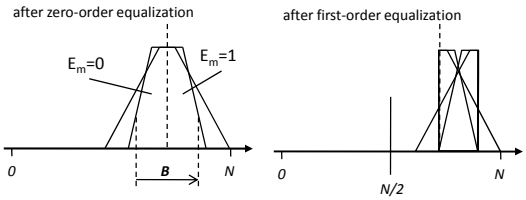

Figure 5: First-order Equalization uses two different correction factors A1 and A2, depending on the value of a selection bit E_m.

3.2 First-order Equalization

Zero-order equalization makes a global adjustment on a masked netlist. First-order equalization goes one step further, and attempts to compress the PMF resulting from zero-order equalization. The idea of first-order equalization is illustrated in Figure 5. We start from the PMF resulting from zero-order equalization, and we partition the PMF again into two parts. This is done through the generation of a selection signal E_m. The subscript *m* indicates that E_m is a masked signal, and that it is obtained through masked logic. The correction mechanism for first-order masking is similar to that of zero-order masking. When E_m is low, we will increase the Hamming Weight of the netlist by a constant amount *B*. When instead E_m is high, we will maintain the same Hamming Weight. The value of B is a function of the circuit under consideration, and can be derived from the PMF for the circuit under a constant mask value.

E_m is a one-bit estimate for the current Hamming weight of the circuit. If the current Hamming weight is below the average, E_m should be zero. If the Hamming weight is above the average, E_m should be one. If we assume a combinational circuit, the value of E_m depends only on the primary inputs of the circuit. The selection function that generates E_m can be created as follows.

1. Create the Hamming weight PMF for the masked circuit while tying the mask bit to a constant 0. Obtain the average *W* for this PMF.

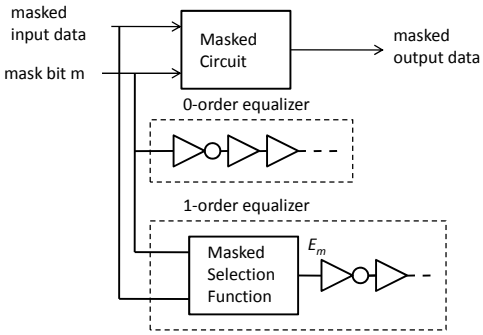

Figure 6: Implementation of a first-order equalization scheme for hardware masking.

2. Enumerate all primary inputs of the circuit and tabulate the corresponding Hamming weight for the circuit. For each entry, determine if that value is above or below *W*. If it is below, the selection value *E* should be 0. Otherwise, the selection value *E* should be 1.
3. Implement the selection function by logic synthesis.
4. Finally, create a masked version of the selection function, so that the signal E_m is obtained. The masking of the selection function is important to avoid the introduction of additional side-channel leakage.

A sample realization of a first-order equalization scheme is illustrated in Figure 6. Next to the 0-order equalizer, we attach a 1-order equalizer, which combines the circuit inputs and the mask bit to generate a selection signal E_m. The selection signal then drives a capacitive load proportional to B (See Figure 5). Optionally, we may iterate step 2 and step 3 multiple times in order to include the Hamming weight of the equalizers in the overall circuit Hamming weight. In our experience however, the equalizer circuits are small compared to the masked circuit they protect (< 10% in size), so that their impact on the selection function is small.

In the next section, we will present a series of experiments we did on a small cryptographic circuit to evaluate the circuit cost and the efficiency of the equalizers.

4. EXPERIMENTAL RESULTS

The simulation setup we use is illustrated in Figure 7. A stream of random 8-bit input samples *in* is masked in a single-bit Boolean masking scheme (Eq. 5), and connected to an SBOX (a cryptographic lookup table). The SBOX is implemented in Random Switching Logic [4]. The implementation is in terms of NAND, NOR and inverter gates, and the netlist was obtained through logic synthesis using Synopsys Design Compiler. A secret key is added to the output of the SBOX, and the output is eventually unmasked. We implement a logic simulation of this system, and collect the Hamming weight of the netlist during

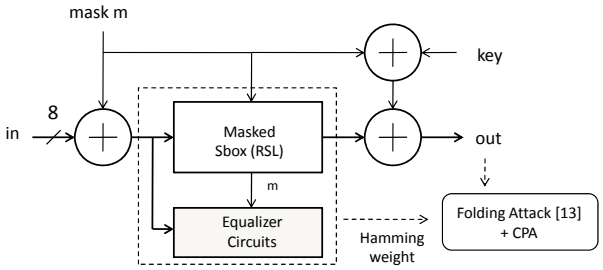

Figure 7: The simulation setup includes an SBOX in Random Switching Logic, and evaluates a higher-order attack based on folding [13].

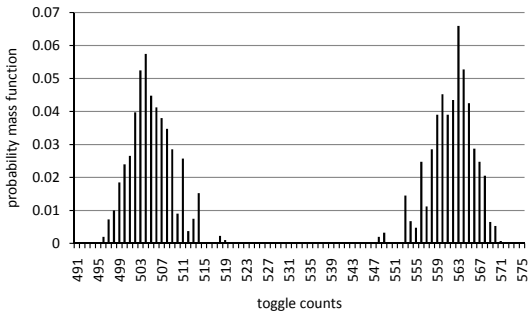

Figure 8: Hamming weight PMF for the SBOX (Figure 7) without any buffer for equalization. Two non-overlapping sub-PMF are clearly distinguishable.

simulation. We only include the weight of the masked gate outputs in the Hamming weight; and the weight of the mask signal is not included.

To evaluate the side-channel resistance of this setup, we mount a higher-order attack on the secret key value, using the folding mechanism described in [13]. The power model used by the correlation analysis is Hamming weight of the input.

4.1 PMF Analysis and Equalization

We initially simulated the RSL SBOX without any buffer for equalization and derived the PMF as shown in Figure 8. The maximum Hamming weight of the netlist is 1067, and the graph is symmetric around 534. Two non-overlapping PMF are clearly distinguishable, and the relative separation between their averages is 55.

To implement zero-order equalization, we evaluate which part of the masked PMF corresponds to *m=0*. Then, we add a correction circuit as illustrated in Figure 4. The amount of loading added by the correction circuit is proportional to the average separation (55) between the non-overlapping PMF. We added a chain of 55 buffers. The resulting PMF of the entire SBOX is illustrated in Figure 9.

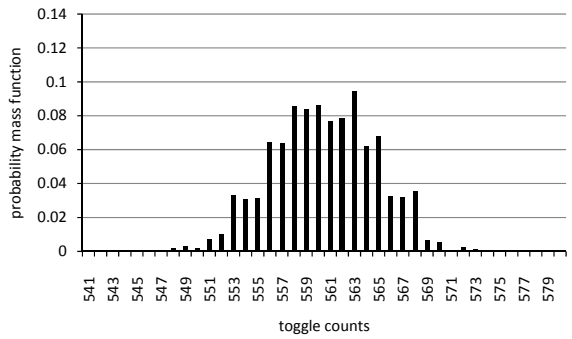

Figure 9: Hamming weight PMF for the SBOX with 0-order equalization.

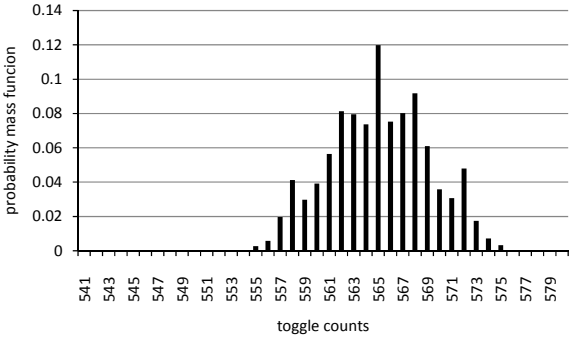

Figure 10: Hamming weight PMF for the SBOX with 1-order equalization.

To implement first-order equalization, we first design the selection function as described in Section 3.2. The selection function drives a chain of buffers with length proportional to the width of the zero-order equalized PMF. We added a chain of 10 buffers. The resulting PMF of the entire SBOX is illustrated in Figure 10.

4.2 Side-channel analysis

As we expected, the setup with the non-equalized SBOX can be broken using a folding attack followed by CPA [13]. The height of the correlation peak obtained by the CPA, and based on the Hamming weight of the input is 0.3568.

Next we consider the case of 0-order equalization. We can plot how the height of the correlation peak evolves as we gradually equalize the Hamming weight of the circuit by adding more buffers to the chain. This is shown in Figure 11. The Y-axis of this curve illustrates the ratio of the correlation peak for the correct key to the maximal correlation peak of all other key guesses achieved by CPA. As long as the absolute value on Y axis is above 1, the CPA will identify the correct key. When it drops below 1, CPA points out the wrong key. As we can see from Figure 11, the X-axis is labeled with the number of buffers used for 0-order equalization. When 25 buffers are added, the

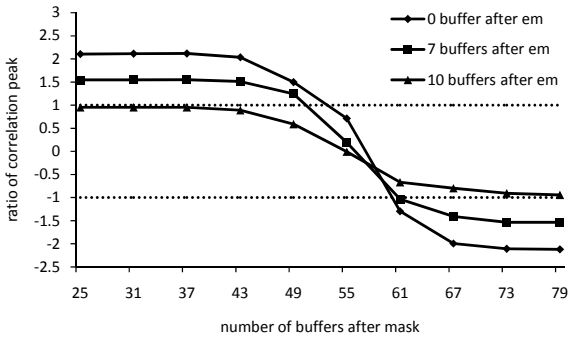

Figure 11: CPA Correlation peak in function of equalizer weight.

two previously separated sub-PMFs begin to overlap. Accordingly, the ratio of the correlation peak starts decreasing. When around 55 (the average separation between the previous non-overlapping PMF) buffers are integrated, the two sub-PMFs overlap to each other to the largest extent. The ratio turns out to be around 0.7 (<1) and the attack is unsuccessful. As the number of buffers increases further, the two sub-PMFs separate in an opposite direction, so the circuit can be successfully attacked again (|ratio| > 1).

Finally, we consider the case of 1-order equalization. The basic idea of 1-order equalization is to reduce the information leakage through compressing the PMF. Comparing Figure 8 and Figure 9, we can find after 1-order equalization, the width of PMF decreases around 10, which is the number of buffers added after E_m. The effects of 1-order equalization is shown in Figure 11. As expected, 1-order equalization makes larger part of the curves fall into the range [-1, 1], which means the circuit has less side-channel leakage and gets stronger protection.

4.3 Circuit Cost

Gate counts for the normal RSL SBOX, 0-order equalized SBOX and 1-order equalized SBOX are presented in Table 1.

Table 1: Circuit Costs (RSL gate counts)

normal RSL SBOX	0-order SBOX	1-order SBOX
972	1027	1132

From the normal RSL SBOX to the one with 0-order equalization, the increase of gate count is 55 (the number of buffers on mask). Compared with a 0-order SBOX, 1-order SBOX costs 105 more gates, which is a combination of the E_m generation circuit and the buffers after E_m. We can see that the costs for the equalizers are not high, especially the 0-order equalizer. However, they provide us with much better security performance. Actually, in a real

circuit, besides adding buffers, there are many other methods to implement equalization. For example, any way to increase the capacitance of E_m can be considered as a solution for 1-order equalization.

5. CONCLUSION

This paper aims to reduce the correlation between the mask and the power consumption of a masked circuit. Its objective is to thwart higher-order attacks, which first remove the effect of the mask, and subsequently break the masking scheme. By analyzing Boolean hardware masking, we presented a solution to equalize the power PMF, including 0-order and 1-order equalizations. The first type, 0-order equalization, minimizes the average separation between the previous non-overlapping sub-PMFs while the second one, 1-order equalization, compresses each sub-PMF. The experimental results demonstrate that both 0-order and 1-order equalizations are two effective and low cost solutions.

6. ACKNOWLEDGEMENTS

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

7. 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, J. R. Rao, P. Rohatgi, “Template Attacks,” Proc. of the Workshop on Crypto-graphic Hardware and Embedded Systems (CHES 2002), LNCS 2523, p. 13-28, Springer, 2002.

[3] S. Mangard, E, Oswald, and T. Popp, “Power Analysis Attacks – Revealing the Secrets of Smart Cards,” Springer, 2007.

[4] 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.

[5] 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.

[6] N. T. Courtois and L. Goubin, “An Algebraic Masking Method to Protect AES Against Power Attacks,” IACR Eprint Archive, 2005/204, http://eprint.iacr.org/2005/204.pdf.

[7] C. Clavier and J.-S. Coron, “On Boolean and Arithmetic Masking against Differential Power Analysis,” In Proc. 2000 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2000), LNCS 1965, p. 231-237, Springer, 2000.

[8] S. Mangard and K. Schramm, “Pinpointing the Side-channel Leakage of Masked AES Hardware Implementation,” In Proc. 2006 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2006), LNCS 4249, p. 76-90, Springer, 2006.

[9] D. Suzuki, M. Saeki, "Security Evaluation of DPA Countermeasures using Dual-Rail Pre-charge Logic Style," ,” In Proc. 2006 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2006), LNCS 4249, p. 255-269, Springer, 2006.

[10] Z. Chen, Y. Zhou, "Dual-rail Random Switching Logic: A Countermeasure to Reduce Side-channel Leakage," In Proc. 2006 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2006), LNCS 4249, p. 242-254, Springer, 2006.

[11] T. S. Messerges, “Using Second-Order Power Analysis to Attack DPA Resistant Soft-ware,” In Proc. 2000 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2000), LNCS 1965, p. 238-251, Springer, 2000.

[12] J. Waddle, D. Wagner, “Towards Efficient Second-order Power Analysis,” Proc. of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), LNCS 3156, p. 1-15, Springer, 2004.

[13] K. Tiri and P. Schaumont, “Changing the Odds against Masked Logic”, Selected Areas in Cryptography 2006 (SAC 2006), LNCS 4356, p. 134-146, 2006.

[14] P. Schaumont and K. Tiri, “Masked 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.

[15] Z. Chen, P. Schaumont, "Slicing Up a Perfect Hardware Masking Scheme," Proc. of the Hardware-Oriented Security and Trust Workshop (HOST 2008), 21-25, 2008.