

Effects of Embedded Decompression and Compaction Architectures on Side-Channel Attack Resistance

Chunsheng Liu

Computer and Electronic Engineering
University of Nebraska-Lincoln
Omaha, NE 68182, USA
cliu@engr.unl.edu

Yu Huang

Mentor Graphics
300, Nickerson Rd
Marlborough, MA 01752, USA
yu.juang@mentor.com

Abstract

Attack resistance has been a critical concern for security-related applications. Various side-channel attacks can be launched to retrieve security information such as encryption key. Prior work does not consider the presence of embedded compression architectures and their impacts on security. In this paper, we analyze the complexity of side-channel attacking on designs with embedded decompression and compaction circuit. We first present a possible attacking strategy and perform analysis on the complexity of attacking circuits with EDT architecture. We then extend the probabilistic analysis to the more general compaction schemes using distance coding. We show that successful attacking of designs with embedded decompressor and compactor is extremely difficult. The complexity is much higher than the results shown in prior work assuming no decompression and compaction. It indicates that the use of embedded compression architectures can achieve higher security level.

1 Introduction

Security has been a major concern in many applications. Hostile attacks can be launched using either software or hardware measures [6]. Attacking security ICs such as encryption/decryption chips is usually done through hardware measures, which can be either physical or side-channel attack. Physical attack is expensive because it needs specific equipments and intensive efforts such as de-packaging, probing or even reverse engineering. Side-channel attack doesn't require the targeted device to be physically opened, hence can be mounted easily.

Various side-channel attack techniques have been proposed [7] through the use of differential fault, power, timing analysis. Many of these works rely purely on functional mode analysis. Scan-based DFT has provided observability and controllability to registers for improved testability. However, it also provides back door to possible side-channel attacking [3]. Attackers can retrieve security information by switching the chip between test mode and functional mode, and unloading the scan data.

Several approaches have been proposed as possible means of side-channel attack through DFT hardware and resisting solutions are discussed [2, 8, 9, 16, 17]. However, prior work has neglected the existence of on-chip decompression/compaction circuit and its impact on security. Such circuitry is widely incorporated in today's complex designs to reduce test data volume and ATE channels, such as Embedded Deterministic Testing (EDT) [14]. Since these circuits are usually embedded within the design during test synthesis, it is difficult for the attackers to bypass them and access the scan chains directly. In addition, users can choose not to provide any bypass mode, which renders the scan-based attacking more difficult. Prior work illustrated that using EDT, test quality and diagnosis resolution in compression mode are almost as good as in bypass mode [1, 4, 14]. As a result, without bypass mode, scan-based attacking is difficult, while the test/diagnosis/debug capabilities are not compromised.

In this paper, we analyze the complexity of scan-based side-channel attack in the presence of decompression/compaction circuit. We present an attacking strategy on designs with EDT architecture and we perform analysis on the complexity of such attack. We then extend the analysis to more general decompression/compaction schemes using distance coding. In both cases, we show that scan-based attacking of such designs is much more difficult than what is reported in prior work assuming no decompression/compaction circuit.

The rest of the paper is organized as follows. Section 2 lists some related prior work on side-channel attacking. Section 3 discusses the attacking complexity in the presence of EDT architecture. In Section 4, we extend the discussion to general decompression/compaction using distance coding. Section 5 concludes this paper.

2 Related Work

Early side-channel attack techniques have been based on analyzing the chip's functional characteristics for retrieving security information [7]. However, such functional mode analysis provides limited information and is easy to be pre-

vented through enhanced secure design with certain overhead. Extensive secure enhancement can be performed on the entire design flow for increased attacking complexity, but also leads to significant overhead [15].

Scan-based design has provided improved controllability and observability to the memory elements, but can also be used as a side-channel for attacking. Attacks can target the vulnerability induced by either observability (e.g. by probing scan-enable and scan-out signals) or controllability (e.g. by hacking scan-control circuit) [3]. Specific attacking strategies targeting encryption chips with scan-design are proposed for Data Encryption Standard (DES) [17] and Advanced Encryption Standard (AES) [16]. By switching the chip between functional mode and test mode, secret key information can be scanned out or calculated.

Traditional countermeasures against scan-based attacks are usually based on the use of fuses in test circuit. After production test, these fuses are permanently blown out, hence accessing the test circuit is impossible. However, these approaches not only need design modification, but also render in-field test and debug impossible [3]. Recently, modified secure scan architectures have been proposed as an alternative [2, 3, 8, 9, 11]. These approaches are designed to counter attacks on either controllability by protecting scan control logic, or observability by protecting the scan chains. As pointed in a panel [5], test compression logic can be modified to prevent the extraction and reverse-engineering of a design’s IP. In this paper, however, we illustrate that even without any modification, the existing test compression technologies are good enough to resist side-channel attack.

One basic approach is to prevent the chip from being switched between functional mode and test mode, such that secure information captured in functional mode can not be scanned out in test mode [2]. Scan control logic can also be hardened by extra protection logic [2]. On the other hand, scan chains can be protected by inserting extra hardware and secure keys [3, 8, 9]. A secure key can be embedded either in test control protocol or scan data, and identified by a verification circuit before scan operation. If verification is successful, normal scan will follow; otherwise, the scan-out data will be scrambled or encrypted.

3 Attacking Scan-Based Design with Decompression/Compaction Circuits

3.1 Preliminaries

None of the prior work has considered the widely adopted decompression/compaction circuit. They usually assume the scan-in and scan-out ports are directly accessible, which is not true in the existence of on-chip decompressor and compactor. These circuits are usually inserted during test synthesis and cannot be easily extracted from the rest of the design. Scan-in and scan-out are now hiding behind the decompressor and compactor, as shown in Fig 1.

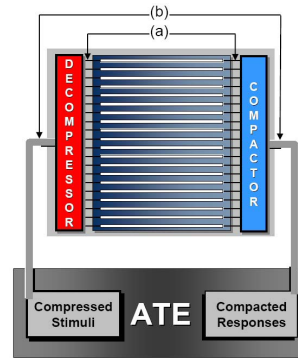


Figure 1. Attacking access points without (a) and with (b) decompressor/compactor.

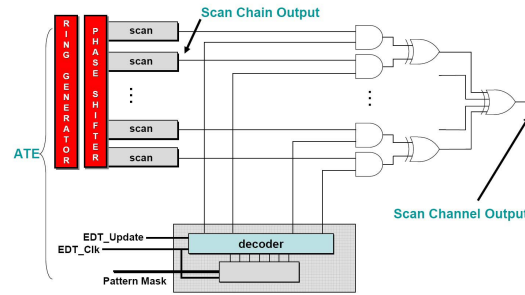


Figure 2. EDT architecture.

The observability is also significantly compromised due to the high compression and compaction ratio, since only a small number of ATE channels are now available for observation. As a result, the existence of such circuit can substantially increase the difficulty of scan-based side-channel attacks.

In order to make our discussion more focused, we will hereinafter base our analysis on the strategies proposed for attacking DES [17] and AES [16] designs, which assume that one or more secret keys are the target of attack. Either the secret key registers or other critical registers, based on which the key can be inferred, are included in scan chains. We refer to these registers as *key registers*. We also assume that scan control logic is not protected so attackers can switch the chip between functional mode and test mode. Although these methods are for specific encryption algorithms such as DES and AES, they represent typical applications of side-channel attacks and the strategies can be easily extended to other secure designs. For the sake of succinctness, we neglect the details of these attacking methods, which are available in [16, 17].

In this section we also assume the designs use EDT methodology [14], because it represents a generic form of decompression/compaction mechanism. Fig 2 illustrates EDT architecture. The decompressor consists of ring generator and phase shifter, while the compactor contains masking logic and XOR tree. Note that our analysis does not rely on the implementation details of EDT, hence it is also suitable for other similar decompression/compaction archi-

tectures.

The prerequisite for scan-based attacks is to find what scan cells correspond to bits in the key registers. Based on specific applications, the exact correspondence between a bit in the key register and its position in scan chains may or may not be needed [16, 17]. Subsequent steps are then needed for the success of attacking. In this paper, we base our analysis on the complexity of identifying key register bits in the scan chains. Note that this is the bottom line for a successful attacking. A high complexity (or low successful probability) of this step indicates either the effort for a successful attack is gigantic or it is virtually impossible.

3.2 Attack without EDT Circuit

Identification of key registers without decompression/compaction circuit is rather straightforward [16, 17]. In attacking DES algorithm [17], one bit change in plain text input results in a single bit change in the key register. Hence identifying an N -bit register requires only N rounds of the following operations: (1) use a plain text as input, run a predetermined number of cycles in functional mode; (2) unload scan chains in test mode; (3) input a plain text different from the last one in only 1 bit, run the same number of functional clocks; (4) unload scan chains in test mode again; (5) compare the two scan-out streams, the changed bit is a key register bit.

In attacking AES algorithm [16], one bit change in plain text input results in multiple changes in the key registers. The goal is to make each bit in key registers to change (i.e., identified) at least once after k rounds of attempt. Without loss of generality, we can assume that in a single round, the probability that a bit cannot be changed is $1/2$. Then after k rounds, the probability that a certain bit is still not identified is $(\frac{1}{2})^k$. It is then easy to see that after k rounds, the probability P that all N bits are identified is $P = (1 - (\frac{1}{2})^k)^N$. Obviously, as k increases, P will quickly approach to 1. This corroborates the result reported in [16]: for $N = 32$, on average only 6 rounds are needed to identify all bits. Therefore, identifying key registers without decompression/compaction circuit is rather easy.

3.3 Attack with EDT Circuit

The compactor usually provides a masking circuit to handle 'X's, such as EDT shown in Fig 2. It is set when a test pattern is scanned in. Since it is hard to be extracted through reverse engineering and can be protected by various means [2], here we assume the attackers don't know the exact design of decompressor and compactor, and the BYPASS is not provided in the design. As a result, they cannot load desired data into scan chains, cannot set a specific pattern in the mask registers and do not know the exact scan-out values before compaction. However, they know the general architectures of the EDT circuit and the encryption algorithm, and can switch the chip between test and functional modes.

Procedure *Key_register_EDT*

1. **While** not all bits identified
2. Current bit targeted = i ;
3. **For** each j , $1 \leq j \leq N$
4. Test mode: scan in initialization pattern p_j ;
5. Functional mode: input a plain text and run required number of cycles;
6. Test mode: scan out response and scan in p_j ;
7. Functional mode: input another plain text with 1-bit difference;
8. Test mode: scan out response and compare;
9. Find initialization pattern set C_1 ;
10. Repeat 3-8 using C_1 as initialization pattern set and change a different bit of input in functional mode;
11. Find initialization pattern set C_2 , bit i identified.

Figure 3. Attacking procedure for identifying key registers.

The masking circuit in EDT can selectively scan out the content of a single scan chain, or all scan chains simultaneously, through a one-hot encoder. Since the attackers cannot specifically set the mask registers, they don't know whether the current scan-out comes from one scan chain or all scan chains, or whether the masking register is properly set. Moreover, they don't know from which scan chain the data is scanned out. Therefore, under these realistic assumptions, the attackers cannot map the scan chain contents to key registers in a deterministic manner. Statistical methods have to be used.

Here we propose a possible attacking approach with EDT circuit. As mentioned earlier, we only attempt to identify the key register bits in scan chains, subsequent steps are omitted for simplicity. Let us assume there are a total of S scan chains and the mask register has r bits, where $2^r > S$ in general. For simplicity, we also assume the DES algorithm is targeted, hence 1 bit change in a plain text input can lead to 1 bit change in the key registers [17]. The attacking procedure is shown in Fig 3.

The procedure attempts to target each bit in the key registers until all of them can be identified, as indicated in Lines 1 and 2. In Line 4, we set the chip in test mode and scan in an initialization pattern p . Since we don't know the specific design of the decompressor and masking circuit, the scan chain content after scan-in and the content in masking register are unknown. Hence any random pattern can be used.

Next in Line 5, we switch the chip to functional mode and apply a plain text input, then start the normal execution for a predetermined number of cycles until the key registers receive intermediate results. If a reset is needed to start functional mode, we assume the mask register won't be cleared.

In Line 6, we switch back to test mode and perform scan-out. Meanwhile, scan in the same pattern as in Line 4. This is to guarantee that the mask register is set to the same value.

In Line 7, we switch to functional mode and apply a plain text input which is different from the first input by only 1 bit. Run the same number of cycles such that the 1-bit change will be captured in the key registers.

Finally in Line 8, we switch it back to test mode and perform scan-out.

Since the initial states of the two functional mode executions in Lines 5 and 7 are identical except for 1-bit change of input, the scan chain contents in these two steps will have only 1-bit difference, corresponding to a bit in the key registers. In order to observe this bit after compaction, the masking register has to contain an initialization pattern (the same pattern in the two steps) that selects only the scan chain containing this bit for scan-out.

However, we do not know what pattern has been set in the masking register since the initialization pattern p is scrambled by the decompressor. It is easy to see the probability that a single scan chain is selected for scan-out is $1/2^r$ (based on the one-hot encoder used in EDT). The probability that all scan chains are selected is $\frac{2^r - S}{2^r}$. In Line 3, the above steps are repeated for a number of N times, each run with a different initialization pattern p_i (hence will likely result in different masking register contents). The average number of times that we can detect the 1-bit change after the compactor should be close to $N_1 = N \frac{2^r - S}{2^r}$. We record this set of random initialization patterns as C_1 in Line 9.

Note that there are two types of patterns in C_1 : those that can uniquely select a single scan chain for scan-out denoted by C_2 , and those that can select all scan chains $C_1 \setminus C_2$. If we can distinguish these two types, we can obtain a set of initialization patterns that can uniquely select a specific single scan chain for scan-out, which contains the changed bit in the key registers. And we have full observation of that bit. This way, each bit of the key registers can be eventually associated with a unique set of initialization patterns. In the subsequent attacking procedures as proposed in [16, 17], applying one such pattern will allow us to observe one bit in the key registers. Hence the whole key registers become observable.

In order to distinguish the two types of patterns in C_1 , we repeat the steps from Line 4 to Line 8 for N_1 times, with the initialization patterns from C_1 , and in functional mode we change a different bit of the plain text input. If the corresponding bit change in the key registers happens to be in the same scan chain with the targeted bit i , then they can be both identified if we can uniquely select that scan chain. So we can proceed to select another plain text input bit to change.

If the plain text input bit we select to change leads to a bit change in the key register, which is not in the same scan chain with the targeted bit i , then the initialization pattern set C_2 that can uniquely select the scan chain cannot detect this change. Hence we can easily find C_2 by observing the scan-out after compaction. As a result, we can use any pattern in C_2 to initialize the chip and the key register contents in a specific scan chain can be observed. From Line 4 to 8, it can be seen that determining the position of a tar-

get bit in a specific scan chain is trivial: we simply observe the compacted stream and find the changed bit. Therefore, using this method we can successfully identify all key register bits by associating each bit with a set of initialization patterns. However, it needs repetitive scan-in and scan-out operations, hence much more efforts than the cases without decompression/compaction circuit [16, 17].

We also note that if resetting the chip will also reset the masking register, a more complex procedure is needed, which is not presented here due to the lack of space. It then follows that we can resist this kind of attack by protecting the masking circuit. Moreover, the attacker can also use this method to infer the number of scan chains S .

3.4 Statistical Analysis

The above procedure shows only one potential attacking method, other similar means are also possible. However, they all need a number of trial runs for the attackers to be confident about the results.

Note that initially the attackers don't know masking register length r , the number of scan chains S and the maximum scan chain length. In order to be confident that a bit change observed at the scan-out after compaction is because a unique scan chain is indeed selected, not because all scan chains are selected, nor some exceptions (e.g. the mask register is never properly set), a number N of runs is required. In the ideal case, the attackers will be most confident about his result when the size of C_2 for uniquely selecting every scan chain is approximately $\frac{N}{2^r}$.

This confident level can be evaluated using statistical analysis. Let X be the random variable representing the result of each trial run. If a specific scan chain i is selected $X_i = 1$, otherwise $X_i = 0$. From Central Limit Theorem, for N trial runs, if the mean of the N samples $\{X_1, X_2, \dots, X_N\}$ is \bar{X} , then \bar{X} follows normal distribution $N(\mu, \frac{\sigma}{\sqrt{N}})$. Note that \bar{X} is simply the size of C_2 divided by N , which in the ideal case should equal $1/2^r$. μ and σ can be expressed as $\mu = 1/2^r, \sigma = \frac{1/2^r}{1-1/2^r}$ if the attackers are confident about their approaches. Otherwise, they can be calculated from observed samples as $\mu = \frac{1}{N} \sum_{i=1}^N X_i, \sigma = \frac{1}{N-1} \sum_{i=1}^N (X_i - \mu)^2$.

The confidence level in normal distribution can then be calculated using its confidence interval, which represents the percentage of sample means that lies within certain standard deviations. The confidence interval can be obtained by $\bar{X} \pm z_{\alpha/2} \frac{\sigma}{\sqrt{N}}$. Fig 4 shows the confidence interval of 99% with $\alpha = 0.01, z_{\alpha/2} = 2.576$. It means that if we take a large number of sample trials, 99% of the time, the unknown population mean is between $\bar{X} - 2.576 \frac{\sigma}{\sqrt{N}}$ and $\bar{X} + 2.576 \frac{\sigma}{\sqrt{N}}$.

In the attacking problem, the confidence level can be elaborated as: the attackers want their observed value of \bar{X} be as close as possible to the theoretical value of $1/2^r$, be-

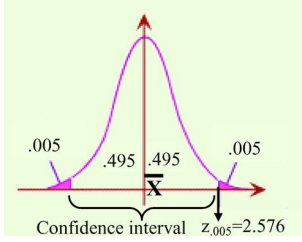


Figure 4. Confidence interval of 99% confidence.

cause that way they will be most confident about the results. E.g., if the attackers want that for a probability of 99%, \bar{X} is deviated from $1/2^r$ by no more than 5%, then the requirement for confidence interval is: $2.576 \frac{\sigma}{\sqrt{N}} < 5\% \mu$. E.g. if $r = 8$, then $N > 2600$.

Since the attackers don't know the length of masking register r , they have to be pessimistic to gain high confidence level. r is directly related to compaction ratio (the number of scan chains to the number of ATE channels). Usually each output channel is associated with an XOR compactor and a masking register, hence we can view the compaction ratio as the number of scan chains. In modern compaction methodologies such as EDT, the compaction ratio can be in the order of 1000, so $r \geq 10$.

In practice, in order to be more confident, the samples of $X = 1$ cannot be too low. If for 1000 trials, only 1-2 times a single scan chain is selected, then its reliability will be questionable. E.g. if the attackers need $X \geq 10$, then $N > 10/\frac{1}{2^r} > 10000$. If the targeted key registers have 32 bits, then in the worst case the attackers need to run the scan and functional operations from Line 4 to Line 8 of Fig 3 for 320,000 times. This is much more difficult than the effort reported in earlier work assuming no decompressor/compactor [16, 17]. Considering other extra overhead, e.g. the scan-in patterns in C_2 may need to be reloaded to ATE each time, this attack requires significant effort and time and can be virtually impossible in practice.

For algorithms like AES, one bit flipping in input can cause avalanche effect in key register and multiple scan cells will change. In this case, success of the above attacking strategy is also possible but needs much more effort. We omit the procedure here. We can then conclude that attacking designs with decompressor/compactor, such as EDT, is extremely difficult. And such attack can be easily prevented, e.g., by protecting the masking circuit in EDT.

4 Analysis with Compaction using Distance Coding

4.1 Preliminaries

We now extend our discussion to a more general situation, where we assume any decompression mechanism is possible and the compactor is based on a generic distance coding. Even more general, we assume that 1 bit change in the plain text input may cause multiple scan cells to change. Note that the key to a successful attack is to identify the

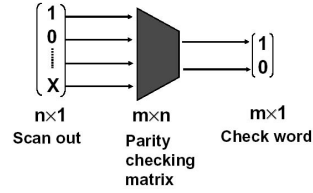


Figure 5. Compaction based on distance coding

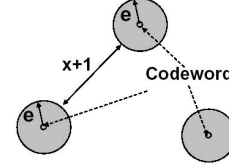


Figure 6. Condition for recovering positions of '1's.

changing scan cells from the compacted scan output. Hence here we focus on the safety of compactor.

Various compaction schemes based on different coding mechanisms are proposed [10, 13], and many are based on distance coding. Following the basic strategy introduced in Section 3, if the number of scan cell changes in each scan-out cycle exceeds the tolerance of the coding, the changes (key register bits) cannot be correctly identified.

The compaction can be abstracted as shown in Fig 5. Scan-out data in one scan clock can contain '1' (scan cell is changed), '0' (scan cell no change) or 'X'. The compactor can be viewed as a parity checking matrix. Each scan-out data block is multiplied by the matrix (compacted) to produce a check word (compaction result). For successful attacking, the attackers have to recover the positions of all '1's in the scan-out data from the check word. Only '1's are important because the key register bits cannot be 'X'. However, the existence of 'X's can compromise the capability of recovering all '1's [10, 12].

For a distance code C_D that has a distance of D between any code words, let the number of bit changes ('1's) and the number of 'X's in one scan-out cycle be e and x , respectively. Then the condition for recovering the positions of all '1's is $D \geq 2e + x + 1$ [12]. If $2e + x + 1 > D \geq e + x + 1$, then the attackers know there is one or more bit changes in this scan-out cycle but cannot determine their positions. In fact, the XOR tree used in EDT is a distance-2 coding and it is impossible to recover all '1's if all scan chains are selected for scan-out. Hence it achieves best security.

The condition is depicted in Fig 6. The center of each circle represents a codeword. Radius of the circle represents the extension of codewords due to bit flipping. In order for attackers to distinguish a codeword from all others, the distance between any two circles has to be greater than x .

4.2 Success Rate of Attack

We now give a quantitative analysis on the probability that an attack can be successful, with the existence of a compactor based on distance coding. Let the number of scan chains be n , the average length of scan chains be l (assuming balanced scan chains), and the probability that a scan

cell can contain an 'X' be P_X , respectively.

If one-bit change of plain text input can cause a total of E scan cells to change in all the scan chains, then the average probability that a scan cell can be changed is approximately $P_E = \frac{E}{nl}$. For attacking DES chip, $E = 1$; for attacking AES chip, $E \geq 1$ and on average $E = R/2$, where R is the total length of key registers.

Attacking will be applied in each scan-out cycle, and the number of bits before compaction is n . If in all n bits, there are j bits of 'X's, then a successful attack will need the number of changes ('1's) to be $i \leq \lfloor \frac{D-j-1}{2} \rfloor$. The probability that all '1's can be recovered is correspondingly $\sum_{i=0}^{\lfloor \frac{D-j-1}{2} \rfloor} \binom{n-j}{i} P_E^i (1 - P_X - P_E)^{n-j-i}$. However, we notice that the number of 'X's j can also vary from 0 to $D - 1$. Therefore, the probability P_C that all '1's can be recovered in a single scan-out cycle is:

$$P_C = \sum_{j=0}^{D-1} \binom{n}{j} P_X^j (1 - P_X)^{n-j} \sum_{i=0}^{\lfloor \frac{D-j-1}{2} \rfloor} \binom{n-j}{i} P_E^i (1 - P_X - P_E)^{n-j-i}.$$

Finally, the probability that all changed bits in all scan-out cycles can be successfully identified is approximately $P = P_C^l$.

In Fig 7, we show the values of P v.s. small values of P_X given typical design parameters $n = 128$, $l = 128$, $D = 4$. We show results for DES ($E = 1$) and AES ($E = 16$). It can be seen that for $P_X > 3 \times 10^{-4}$, the success probability is almost 0. In practice, the percentage of 'X's is typically much higher than the order of 10^{-4} . Therefore, success rate is extremely low.

Note that this expression is also related to compaction ratio. This is because compaction ratio determines the parity checking matrix, and hence the value of D . In practice, large D can never be used due to the excessive hardware overhead on compactor. As a result, for a practical compactor design and typical percentage of 'X's, it is virtually impossible to attack the key registers.

5 Conclusions

In this paper, the complexity of scan-based side-channel attack on designs with embedded decompression/compaction circuit is analyzed. We have presented a possible attacking strategy targeting designs using EDT architecture. We have performed statistical and probabilistic analysis and shown such attacks require significant effort. We have extended the analysis to the more general compaction schemes using distance coding. In both cases, we have shown that the complexity of attacking designs with embedded decompressor/compactor is much higher than the results shown in prior work, which did not consider decompressor and compactor. Such a high complexity renders scan-based side-channel attacks extremely difficult, or impossible in practice. It indicates that the use of embedded decompression/compaction architecture leads to higher security level.

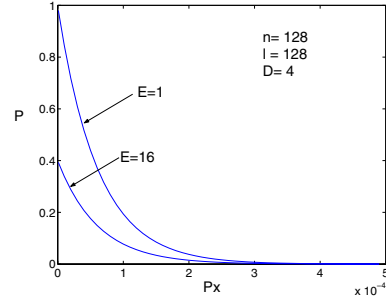


Figure 7. Probability of successful attack with compactor.

References

- [1] W. Cheng et al., Compactor Independent Direct Diagnosis. *Proc. Asian Test Symp.*, pp. 204-209, 2004.
- [2] D. Hély et al., Test Control for Secure Scan Designs. *Proc. European Test Symp.*, pp. 190-195, 2005.
- [3] D. Hély et al., Scan Design and Secure Chip. *Proc. Int. On-Line Testing Symp.*, pp. 219-224, 2004.
- [4] Y. Huang, W. Cheng and J. Rajski, Compressed Pattern Diagnosis For Scan Chain Failures. *Proc. Int. Test Conf.*, paper 30.3, 2005.
- [5] R. Kapur, Security vs. Test Quality: Are They Mutually Exclusive? *Proc. Int. Test Conf.*, p1414, 2004.
- [6] P. Kocher et al., Security as a New Dimension in Embedded System Design. *Proc. Design Automation Conf.*, pp. 753-760, 2004.
- [7] P. Kocher, J. Jaffe and B. Jun, Differential Power Analysis. *CRYPTO*, pp. 388-397, 1999.
- [8] J. Lee et al., Securing Scan Design Using Lock and Key Technique. *Proc. Int. Symp. DFT*, pp. 51-62, 2005.
- [9] J. Lee et al., A Low-Cost Solution for Protecting IPs Against Scan-Based Side-Channel Attacks. *Proc. VLSI Test Symp.*, pp. 94-99, 2006.
- [10] S. Mitra and K. S. Kim, X-Compact: An Efficient Response Compaction Technique. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, pp. 421-432, 2004.
- [11] D. Mukhopadhyay et al., CryptoScan: A Secured Scan Chain Architecture. *Proc. Asian Test Symp.*, pp. 348-353, 2005.
- [12] J.H. Patel, S.S. Lumetta, and S.M. Reddy, Application of Saluja-Karpovsky Compactors to Test Responses with Many Unknowns. *Proc. VLSI Test Symp.*, pp. 107-112, 2003.
- [13] J. Rajski, J. Tyszer, C. Wang and S. M. Reddy, Convolutional Compaction of Test Responses. *Proc. Int. Test Conf.*, pp. 745-754, 2003.
- [14] J. Rajski, et al., Embedded Deterministic Test for Low Cost Manufacturing Test. *Proc. Int. Test Conf.*, pp. 301-310, 2002.
- [15] K. Tiri and I. Verbauwhede, A VLSI Design Flow for Secure Side-Channel Attack Resistant ICs. *Proc. Design, Automation and Test in Europe*, pp. 58-63, 2005.
- [16] B. Yang, K. Wu and R. Karri, Secure Scan: A Design-for-Test Architecture for Crypto Chips. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 2287-2293, 2006.
- [17] B. Yang, K. Wu and R. Karri, Scan Based Side Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard. *Proc. Int. Test Conf.*, pp. 339-344, 2004.