Security Analysis of Industrial Test Compression Schemes

Amitabh Das, *Graduate Student Member, IEEE*, Barış Ege, Santosh Ghosh, Lejla Batina and Ingrid Verbauwhede, *Fellow, IEEE*

***Abstract*—Test compression is widely used for reducing test time and cost of a VLSI circuit. It is also claimed to provide security against scan based side-channel attacks. This paper pursues the legitimacy of this claim and presents scan attack vulnerabilities of test compression schemes used in commercial EDA tools. A publicly available AES design is used and test compression structures provided by Synopsys, Cadence and Mentor Graphics DfT tools are inserted into the design. Experimental results of the differential scan attacks employed in this paper suggest that tools using X-masking and X-tolerance are vulnerable and leak information about the secret key. Differential scan attacks on these schemes have been demonstrated to have a best case success rate of 94.22% and 74.94% respectively for a random scan design. On the other hand, time compaction seems to be the strongest choice with the best case success rate of 3.55%. In addition, similar attacks are also performed on existing scan attack countermeasures proposed in literature, thus experimentally evaluating their practical security. Finally, a suitable countermeasure is proposed and compared to the previously proposed countermeasures.**

***Index Terms*—Test compression, Security, Scan attack, Adaptive scan, OPMISR, Embedded deterministic test, countermeasures.**

I. Introduction

In VLSI industry, design for testability (DfT) infrastructure is included in most circuits for efficient testing of the final product. However when circuits with cryptographic algorithms are concerned, the testing functionality can be exploited by an attacker to recover the secret key used for encryption. These attacks can be categorized as a form of side channel attacks, which target the scan chain structure widely deployed as a DfT technique.

Scan chains were exploited in [1] to recover secret keys of Data Encryption Standard (DES) and Advanced Encryption Standard (AES) hardware implementations. Later on, these scan based attacks are extended to break public-key ciphers in [2]–[5]. Some of these works take into account the modern test compression schemes and the attacks are performed on a generic structure of the schemes. XOR-tree based test compression was attacked successfully using the signature attack in [6], [7], while advanced DfT structures such as X-Masking and partial scan were attacked by a new attack in [8].

In this paper, we have performed scan based side-channel attacks on popular DfT structures generated by the major test tools of leading EDA vendors: Synopsys, Cadence, and Mentor Graphics. The specific test structures are incorporated on an AES circuit using the DfT toolkits. Success rates of the attacks are reported in this paper for different DfT configurations. Though AES is taken as a case study, the scan attack principle outlined in this work is also applicable for symmetric-key ciphers which have similar diffusion properties. One of the purpose of this paper is to illustrate that the classical differential scan attack (DSA) principle [1] is still useful for attacking advanced DfT structures such as test compression with X-Tolerant and X-Masking by enhancing it further using techniques outlined in this paper and in [9].

Apart from performing DSA on industrial test structure, another contribution of the paper is to analyze the security provided by the scan attack countermeasures. Finally, we propose a new countermeasure along with experimental results which ensures security against DSA, albeit with increase in test cost. The preliminary scan attack results on generic test compression structures were presented in [9] along with a brief discussion on the effectiveness of scan attack countermeasures. This work employs the same attack principle as in [9]. However, substantial expansion has been made by providing comprehensive scan attack success results on AES in the presence of actual test compression and X-state handling schemes included in commercial test tools for a wide distribution of active scan chains, active slices and distributions of key-dependent flip-flops (KFFs) on active slices and scan chains. Moreover, an analysis based on the number of inputs required for a successful scan attack on AES designs with test compression is also provided. Attack successes on the scan attack countermeasures is given in detail. A noise injector countermeasure is proposed and its implementation and comparison with other countermeasures is provided.

This work was supported in part by the Research Council KU Leuven: TENSE (GOA/11/007), by iMinds, by the Flemish Government, FWO G.0550.12N and by the Hercules Foundation AKUL/11/19.

Amitabh Das is a PhD researcher in the Department of Electrical Engineering at KU Leuven, ESAT/COSIC and iMinds, Heverlee, Belgium, e-mail: Amitabh.Das@esat.kuleuven.be.

Barış Ege is a PhD researcher in the ICIS/Digital Security Group at Radboud University Nijmegen, Nijmegen, The Netherlands, e-mail: B.Ege@cs.ru.nl.

Santosh Ghosh is a Security Researcher at the Security Center of Excellence (SeCoE), Intel Corporation, 2111 NE 25th Avenue, Hillsboro, OR 97124, US. He was previously a Postdoctoral Fellow the Department of Electrical Engineering at KU Leuven, ESAT/COSIC and was funded by the IAP Programme P6/26 BCRYPT of Belgian Science Policy (Belspo) when part of the work was performed, e-mail: Santosh.Ghosh@intel.com.

Lejla Batina is an Assistant Professor in the ICIS/Digital Security Group at Radboud University Nijmegen, Nijmegen, The Netherlands, email: lejla@cs.ru.nl.

Ingrid Verbauwhede is a Professor in the Department of Electrical Engineering at KU Leuven, ESAT/COSIC and iMinds, Heverlee, Belgium, e-mail: Ingrid.Verbauwhede@esat.kuleuven.be.

The rest of the paper is organized in the following way. In Section II, we briefly describe the AES block cipher, scan attack and industrial DfT techniques. Previous work on DSA is described in Section III. We provide the motivation and objective for our work in Section IV. The overall DSA strategy used in this work is outlined in Section V. Sections VI, VII and VIII provide our scan attack results on test strategies provided by the main EDA vendors. In Section IX, we first present DSA on the existing scan attack countermeasures, followed by our new noise injector countermeasure. Finally, the paper is concluded in Section X.

II. Background

A. AES

AES [10] is one of the widely used industrial standard block ciphers which encrypts blocks of 128-bit messages and supports variable key sizes: 128 bits, 192 bits, or 256 bits [10]. The AES round function consists of four operations which are applied to the cipher state in the following order: SubBytes, ShiftRows, MixColumns and AddRoundKey. The SubBytes operation is a non-linear transformation which operates on each byte of the state. ShiftRows rotates the bytes in each row of the state. The MixColumns multiplies each column with a Maximum Distance Separable (MDS) matrix of branch number five. Therefore, each byte of the input will affect all four bytes of the MixColumns output which forms the basis of DSAs on AES. For instance, a nonzero byte difference in the first byte, as in Fig. 1, will transform into a non-zero difference after SubBytes and will not be affected by the ShiftRows operation. This difference will be transformed by the MixColumns operation and would lead to four non-zero differences on that column. Since the AddRoundKey operation simply XORs the round key to the state, it has no effect on difference propagation as long as the same key is used. Additionally, there is an initial key XOR step before the encryption starts, and this is the operation that is targeted in scan attacks to recover the encryption key of AES. In this work, we have used an open-source AES implementation from the Gezel hardware/software co-design website [11], as our attack target. This design contains table-lookup based S-box. The Gezel HDL code has been converted into VHDL using the `fdlvhd` converter tool and synthesized into gate-level Verilog netlist using Synopsys Design Compiler v2009.06 with a Faraday 130 nm library, on which the test compression structures are added.

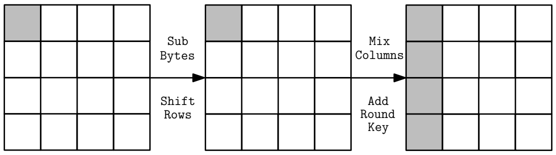

Fig. 1. Difference propagation of a byte difference through an AES round.

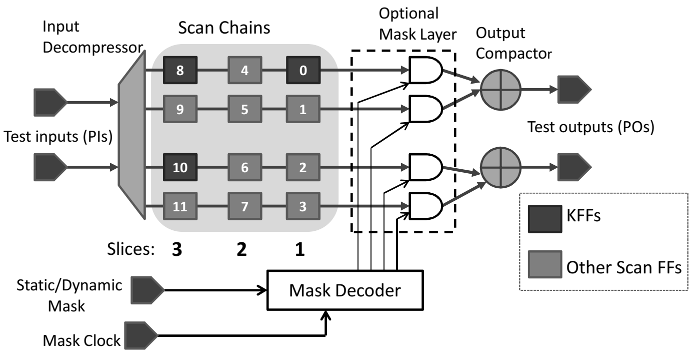

Fig. 2. Generic scan compression structure with X-state handling

B. Introduction to Scan Attack on AES

The target of scan attacks is generally the register storing the computation results of intermediate operations (for instance, the AES round register). Scan attacks are possible even if the secret key register is not included in the scan chains. The ability of the circuit to switch between normal and test mode is exploited in scan-based attacks. Following are the steps that are generally employed for mounting such an attack on an AES implementation:

- The AES circuit is run in functional mode for one round mode after feeding the desired plaintext input.
- The AES circuit is put in test mode by enabling the scan enable pin.
- The contents of the AES round register are scanned out and stored.
- The process is repeated several times for different pairs of inputs with a fixed Hamming distance, and an offline evaluation of the scan outputs is made to take a decision on the secret key.

Differential scan attack exploits the fact that two particular inputs to the round function of AES can transform into output vectors with a unique Hamming distance difference after one round of encryption.

C. Industrial Test Compression Schemes

Test compression is now widely deployed in the semiconductor industry for testing complex circuits in a short time without compromising on test quality. An example of a space compaction scheme using output XOR gates is shown in Fig. 2. Each key-dependent bit (KFF) in the figure is a part of the intermediate register. In the figure, each column of scan flip/flops represents a slice. A slice containing at least one KFF is called an active slice. Similarly, an active scan chain is defined as a scan chain containing at least one KFF. Fig. 2 shows a scan design with 3 KFFs which are distributed over 2 active scan chains and 2 active slices. In this work, attack success rates are presented for a scan structure containing a maximum of 32 scan chains and 32 active slices. The number 32 is chosen in this paper because it corresponds to the four bytes affected by the AES Mix Columns operation. When test vectors are generated for a circuit by an automatic test pattern generator (ATPG), most of the test vector bits are unspecified, or don't care states, which are randomly filled with 0s or 1s, to enable their use on an Automatic Test Equipment (ATE). These states can be removed from the test vectors in an efficient manner using test compression,

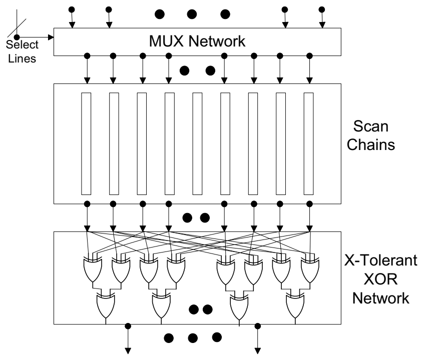

Fig. 3. X-Tolerant compressor logic structure.

allowing for substantial reduction in test time and cost. The three popular industrial test compression tools are Synopsys DFTMAX employing Adaptive Scan, Cadence Encounter Test using OPMISR, and Mentor Graphics Tessent TestKompress using Embedded Deterministic Test (EDT). In addition to providing space compaction, industrial DfT tools also have provisions such as X-tolerance and X-masking for dealing with unknown values (X-states).

Fig. 3 shows the structure of a X-Tolerant space compressor employed in Synopsys Adaptive Scan [12]. Due to the combination of different scan chain outputs at the space compactor, when DSA is considered, there is further loss of observability of the internal states of the scan chains compared to a simple XOR tree. The other technique, X-masking, as shown in Fig. 2, blocks certain parts of the scan chains which contain a higher probability of unknown states (X-states). This is achieved by inserting a mask layer between the test outputs and the test response compactor. The mask can be static as in OPMISR or dynamic as in EDT. As shown in the figure, in static masking a fixed mask value feeds the mask decoder, whereas in dynamic masking, part of the input vector is given to the decoder to generate a variable mask. This masking makes DSA even more difficult as some of the KFFs are not present in the compacted output anymore. Time compaction uses sequential logic to compact test responses. Here, multiple input signature registers (MISRs) are employed to reduce test time. In order to minimize the need to shift out test responses, the scan cell outputs are compressed into a signature with a MISR [13]. In Built-in Self-Test(BIST), this compressed signature is compared with a golden signature.

Since in test compression schemes, the externally observable values are only the compressed stimuli and compacted responses, one would expect them to have some security properties as well. The theoretical security analysis of EDT was presented in [14], while the security claims of Tessent TestKompress tool are investigated in a Mentor Graphics whitepaper [15]. Although no such security claims have been made to-date from Cadence and Synopsys, they offer similar test response compaction structures.

III. Previous Work

The target of scan attacks is generally the register storing the computation results of intermediate operations (for instance, the AES round register). Scan attacks are possible even if the secret key register is not included in the scan chains. The ability of the circuit to switch between normal and test mode is exploited in scan based attacks. DSA is based on the fact that two particular inputs to the round function of AES can transform into output vectors with a unique Hamming distance difference after one round of encryption.

Scan based attack on AES appears in [1], in which a two-step approach employing chosen plaintexts for attacking the round register of AES is presented. In the first step, the scan chain structure is determined and then the round key is recovered in the second step. This proceeds as follows: the chip is run in functional mode for one clock cycle. The result after the first round is stored in the round register. Then the chip is switched to test mode and the contents of the round register are scanned out. This step is repeated for another plaintext input differing in only one byte. Analyzing the distribution of the output Hamming distances for all possible 2^7 pairs generated with byte difference $(01)_x$ in their LSB, one can easily verify that there are four Hamming distance values (9, 12, 23 and 24) which can only be generated by a unique pair of inputs. Therefore, whenever such a Hamming distance is observed between the output vectors, one can XOR the corresponding plaintext byte with the pre-computed values to recover a byte of the encryption key.

Scan chains are usually combined with test compression in order to reduce test time and cost of a complex circuit without compromising test quality. Some recent works on scan attacks [6]–[8] consider test compression to some extent. An XOR tree space compactor has been targeted in [6], [7]. However, commercial EDA tools provide several space and time compaction options including X-state handling, which are much more complex than XOR trees. These test schemes reduce the amount of observability on the scan outputs and also the correspondence to the internal scan chain contents for DSA. Hence, in [8], a new attack is proposed which is able to deal with these advanced test compression infrastructures. In that paper, the authors search for a particular difference after the non-linear substitution operation of the AES round, thereby exploiting the widely used linear structures in the test compression schemes.

IV. Motivation

As described in Section II, commercial EDA tools provide several variants of space and time compaction techniques. The X-tolerant/ X-masking space compaction and MISR-based time compaction logic present in these tools are much more complex than simple XOR trees assumed in some of the previous works. These complex test structures are very often used in practical cryptographic circuits to meet current market demands of reducing test time, achieving lower product cost, and faster time-to-market. In this paper, these advanced test compression techniques are targeted depicting their vulnerability to DSA introduced in [1] and improved in [6]. This paper is aimed to show the applicability of the attack approach followed in these works to commercial test compression schemes and scan attack countermeasures.

Test compression structures are emulated in software on an AES circuit [11] followed by a statistical evaluation of attack success rates.

Scan attack countermeasures, such as partial scan [16] and scan chain scrambling [17] have been proposed to increase confusion on the scan out data. In this work, the vulnerability of these countermeasures to DSA is investigated. The countermeasures are emulated in software, and DSA is performed to illustrate their susceptibility to scan attacks. We enhance the scan attack principles outlined in [1], [6] by combining multiple attacks with distinct input differences to improve the attack success rate when applied to commercial test compression schemes and scan attack countermeasures. Moreover, we present comprehensive attack results for the popular space and time compaction techniques and countermeasures for different distributions of key information on the scan infrastructure, which was not extensively considered earlier and is addressed in detail in this work. Another unique feature of the current work is the investigation of the variation of success rates with respect to the number of test inputs.

V. Attack Strategy

A. General Scan Attack Strategy

Building upon the DSA basics given in Section III, the following assumptions are made in the DSA presented in this work.

- The scan enable pin can be controlled by the attacker.
- The cryptographic algorithm is known to the attacker.
- The time required to execute the target operation is known to the attacker.
- The test structure type (space compaction, time compaction, X-tolerance, X-masking) is known to the attacker. However, the attacker may not be aware of the exact details of the test scheme (such as compression ratio, number of scan chains).

The first paper that discusses the model for the attacker is [18], whereas the attacker model specific to AES is discussed in [19]. In addition to this, an implicit assumption in DSA is that all FFs (except the KFFs) have the same value after one encryption round. Thus the differential process eliminates the effect of these FFs.

However, different from the previously published works, evaluating test compression schemes from different EDA vendors requires modifications to the attack implementation applicable for generic test compression algorithms. In fact, when X-masking or X-tolerant logic is considered, one has to use a modified version of the key guessing strategy proposed in the previous works. The main differences in the attack strategy used in this work and the previous works is that multiple byte differences are used for attacking designs, and the attacks are repeated for random test inputs depending on the testing scheme under security analysis.

The DSA outlined in this paper is performed on the software emulation of the DfT structures. The approach taken in our attack is presented in Fig. 4. It is divided into two phases: an online phase and an offline phase. In the online phase, testing structure of industrial DfT solutions is derived by inserting DfT functionality to an AES design using the DfT tools. Also, possible inputs to the round function of AES are derived for corresponding input differences in one byte. In the offline phase, the scan attack is performed by emulating the DfT structures in a C program, making use of the XOR differences and possible inputs derived in the online phase. An attack is deemed to be successful whenever the correct key byte is suggested as the most likely key byte after the attack. Success rates are computed over 10000 random permutations of KFFs.

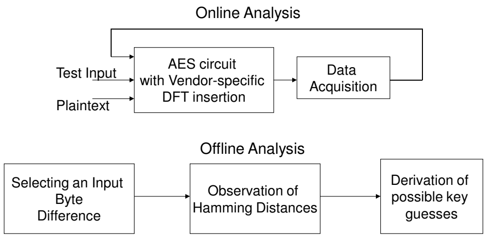
Online Analysis
Test Input
Plaintext
AES circuit with Vendor-specific DFT insertion
Data Acquisition
Offline Analysis
Selecting an Input Byte Difference
Observation of Hamming Distances
Derivation of possible key guesses

Fig. 4. Combined hardware/software scan attack approach.

All the attack successes presented in the following sections are obtained employing attack codes written in C customized for the specific test compression structure or scan attack countermeasure. Simulations are performed on a 64-bit x86-64 Intel Core i7-2600 CPU running at 3.4 GHz having 7 virtual processors and 8GB of RAM.

B. Distributions Considered

The AES round register can be spread in the scan structure in several ways unknown to the attacker. This work considers the following distributions of active slices and active scan chains to cover the most typical scenarios:

- 32 active scan chains with active slices varying from 16 to 32.
- 32 active slices with active scan chains varying from 16 to 32.
- 24 active scan chains with 24 active slices.

In AES, one byte of input difference can affect up to 32 bits of the round register due to the Mix Column operation after one round. Therefore, there can be at most 32 KFFs would be in the scan structure. This indicates that there will be at most 32 active scan chains when each scan chain contains one KFF. Similarly, there will be at most 32 active slices when each slice contains one KFF. These represent two extreme cases. To consider other practical scenarios, the number of active scan chains is varied from 32 to 16 keeping active slices fixed at 32, and vice-versa. An intermediate scenario of 24 active scan chains and 24 active slices is also considered to show the effect of different KFF distributions when the number of active slices and active scan chains is fixed. Table I shows the distributions used in the paper. Even if AES forms a small part of an industrial SoC consisting of millions of gates, the distributions considered in this paper are still applicable. Independent of the size of the design, the maximum number of active slices or active scan chains is always equal to 32 (as only 32 FFs would be key-dependent and the rest unrelated when subjected to DSAs).

TABLE I
DISTRIBUTIONS FOR 24 ACTIVE SLICES / SCAN CHAINS

#	Distribution
1	$\{2, 2, 2, 2, 2, 2, 2, 2, 1, \ldots, 1\}$
2	$\{3, 2, 2, 2, 2, 2, 2, 1, \ldots, 1\}$
3	$\{3, 3, 2, 2, 2, 2, 1, \ldots, 1\}$
4	$\{3, 3, 3, 2, 2, 1, \ldots, 1\}$
5	$\{3, 3, 3, 3, 1, \ldots, 1\}$
6	$\{4, 2, 2, 2, 2, 2, 1, \ldots, 1\}$
7	$\{4, 3, 2, 2, 2, 1, \ldots, 1\}$
8	$\{4, 3, 3, 2, 1, \ldots, 1\}$
9	$\{4, 4, 2, 2, 1, \ldots, 1\}$
10	$\{4, 4, 3, 1, \ldots, 1\}$
11	$\{5, 2, 2, 2, 2, 1, \ldots, 1\}$
12	$\{5, 3, 2, 2, 1, \ldots, 1\}$
13	$\{5, 3, 3, 1, \ldots, 1\}$
14	$\{5, 4, 2, 1, \ldots, 1\}$
15	$\{5, 5, 1, \ldots, 1\}$
16	$\{6, 2, 2, 2, 1, \ldots, 1\}$
17	$\{6, 3, 2, 1, \ldots, 1\}$
18	$\{6, 4, 1, \ldots, 1\}$
19	$\{7, 2, 2, 1, \ldots, 1\}$
20	$\{7, 3, 1, \ldots, 1\}$
21	$\{8, 2, 1, \ldots, 1\}$
22	$\{9, 1, \ldots, 1\}$

In the table, each digit represents the number of KFFs in one slice. There are 24 KFFs in total in each distribution. For instance, the first distribution has two KFFs on 8 slices, and one KFF each on another 8 slices. Similarly, for the 22nd distribution, there are 9 KFFs on one slice and one KFF each on 15 slices. Since there are 22 possible choices for distributing active slices, there are a total of $22^2 = 484$ distributions with 24 active slices and 24 active scan chains. These distributions are in fact all possible choices to distribute 32 KFFs over 24 active slices / scan chains. Since in these distributions, there are multiple FFs on each active slice, we consider cases where one KFF would mask another on the same slice, as is possible in a realistic scenario.

VI. SCAN ATTACK ON SYNOPSYS ADAPTIVE SCAN

A. Introduction to Adaptive Scan

Adaptive scan is the test compression architecture used in the Synopsys DFTMAX test tool. At the input side, there are multiplexers to enable testing of multiple scan chains using a reduced number of scan inputs. At the output side, there is an XOR network to connect multiple scan chains to reduce the number of test outputs. This way compression is achieved without compromising on testability. The output side XOR compactor network is also known as the unload compressor which provides an X-tolerant design. This helps in diagnosing high volume of scan pattern failures which can be observed on the tester [13]. The X-tolerant compressor [20] is based on an algorithm derived from Steiner systems, and provides a good balance between scan compression, and silicon area. To prevent aliasing, cancellation of simultaneous faults on two or more chains, a unique combination of sub-scan chains connects to each compactor output. This combination depends on the compaction structure employed. Sometimes, the outputs are computed by XORing disjoint subsets of scan outputs.

Since the X-tolerant XOR compressor used in Adaptive Scan has a different structure than an XOR-tree, DSA success rates are expected to be different than the success rates for the attacks available in the literature. Observed Hamming distances vary depending on the structure of the XOR network, therefore providing some security through obscurity as long as the structure of the XOR connections is not known. However, this compressor also leaks information on the Hamming distance (HD) between outputs as it consists of linear operations.

B. Description of the Attack

Similar to the initial attack in [1], our scan attack consists of two main steps. First, all 256 possible values are given to the first byte of the plaintext and corresponding first round outputs are collected. In the second step, these outputs are paired depending on the selected input difference and the Hamming distance between them is computed. Unlike previous works, we use five different XOR differences (namely `0xD1,0x01,0x89,0x4A,0x69`) to amplify the visibility of the correct key among other key guesses. Note that all test outputs are XORed together to obtain a single value for the Hamming distance. This is important since the structure of the X-tolerant logic used can vary depending on the number of scan chains and the number of test outputs in the design. The X-tolerant logic used in this work has been derived from an actual test compression DfT insertion with a 32:8 compressor on the AES design by Synopsys DFT Compiler. It combines the scan chain outputs in the following way:
$out_0 = s_2 \oplus s_5 \oplus s_{24} \oplus s_{26} \oplus s_{19} \oplus s_{13} \oplus s_8 \oplus s_{29} \oplus s_{16} \odot s_{22} \oplus s_0 \oplus s_{11}$
$out_1 = s_3 \oplus s_8 \oplus s_5 \oplus s_{27} \oplus s_{16} \oplus s_{18} \oplus s_{13} \oplus s_{21} \oplus s_{23} \oplus s_{29} \oplus s_0 \oplus s_{10}$
$out_2 = s_{28} \oplus s_{25} \oplus s_{17} \oplus s_{20} \oplus s_{22} \oplus s_{14} \oplus s_9 \oplus s_3 \oplus s_{31} \odot s_6 \oplus s_0 \odot s_{11}$
$out_3 = \neg s_{30} \oplus s_{27} \oplus s_{14} \oplus s_{19} \oplus s_{21} \oplus s_1 \oplus s_6 \oplus s_{22} \oplus s_3 \oplus s_8 \oplus s_{11} \oplus s_{17}$
$out_4 = s_1 \odot s_9 \oplus s_{12} \oplus s_{15} \oplus s_{18} \oplus s_4 \oplus s_{26} \oplus s_{20} \oplus s_{31} \odot s_6 \oplus s_{23} \oplus s_{29}$
$out_5 = \neg s_{30} \oplus s_{27} \oplus s_{14} \oplus s_{19} \oplus s_{25} \oplus s_{12} \oplus s_7 \oplus s_4 \oplus s_1 \oplus s_9 \oplus s_{16} \odot s_{22}$
$out_6 = s_{10} \oplus s_7 \oplus s_{13} \oplus s_{21} \oplus s_{28} \oplus s_{15} \oplus s_{31} \oplus s_2 \oplus s_{24} \oplus s_{26} \oplus s_{18} \oplus s_4$
$out_7 = s_{28} \oplus s_{25} \oplus s_{17} \oplus s_{20} \oplus \neg s_{30} \odot s_{23} \odot s_{12} \oplus s_{15} \oplus s_2 \oplus s_5 \oplus s_{10} \oplus s_7$
where $s_i,\ i \in \{0, \ldots, 31\}$ are the scan chain outputs.

In the compactor structure above, each scan chain occurs at least twice at the compactor outputs to satisfy the X-Tolerant requirement of canceling X-states occurring on the scan chains. Since all the scan chains are included more than once, evaluating compactor outputs separately can result in an incorrect estimation of the actual Hamming distance between the corresponding scan designs. Therefore, the test outputs are XORed to make sure that the observed Hamming distance is smaller than or equal to the actual Hamming distance.This is due to the fact that without the knowledge of the exact structure of the output compactor, it would not be possible to tell which scan chains contribute to the differences observed in particular compactor outputs. In some cases that we observed for different X-tolerant logic designs generated by Synopsys, a scan chain is included in an even number of compactor outputs. In this case, XORing the compactor outputs will cancel out these scan chain outputs, therefore degrading the observability of the actual Hamming distance between two scan designs. When the case in the example above is considered, one can easily see that s_{22} and s_{24} are the only ones that are included in an even number of times in the compactor output. Therefore, when performing the analysis, they will be canceled out and therefore information about two particular scan chains s_{22} and s_{24} will be lost. Hence, a decrease in the success rate for a random distribution of KFFs is to be expected when compared to an XOR tree.

A key guess is performed if the observed Hamming distance is one of the extreme cases. For example for the XOR difference `0xD1`, we make a key guess if the observed Hamming distance is less than 5 or 9, and if it is exactly equal to 23 or 24. Our experiments show that this approach improves the overall success rate of the attack.

C. Attack results

Table II summarizes the attack success for different number of active slices and active scan chains. The results suggest that the attack success decreases with decrease in the number of active slices. However, it seems that the success rate is affected to a much smaller extent with variations in the number of active scan chains.

TABLE II
DSA Success rates for X-Tolerant Logic for different distributions

#Active Scan Chains = 32		#Active Slices = 32	
#Active Slices	Success Rate	#Active Scan Chains	Success Rate
32	74.94%	32	74.94%
31	71.22%	31	74.83%
30	70.24%	30	74.24%
29	66.35%	29	74.11%
28	62.65%	28	74.28%
27	60.60%	27	74.45%
26	59.16%	26	74.22%
25	57.93%	25	74.13%
24	57.26%	24	74.06%
23	55.90%	23	74.19%
22	54.38%	22	74.56%
21	49.65%	21	73.58%
20	50.66%	20	61.98%
19	49.79%	19	56.65%
18	44.62%	18	56.78%
17	37.59%	17	57.34%
16	30.26%	16	56.09%

Fig. 5 tells another interesting story. Although the number of active slices affects the success rate quite significantly (see Table II), it seems the effect on the distribution over 24 active slices is minimal. When the number of active slices and active scan chains are both fixed at 24, the average success rate of the attack is around 76.01% for a random permutation of KFFs.As stated in Section VI-B, for the 32:8 X-tolerant structure used in this work, the information on only two scan chains (namely s_{22} and s_{24}) are lost after XORing the two test outputs. Although the loss of information is minimal, some distributions of scan chains lead to further reductions in the observed Hamming distance value. Hence, the success rate is inevitably degraded. Experiments are repeated for 10000 random distributions of KFFs to have reliable statistics on the attack success. It takes an average of 1.28 milliseconds to perform the attack once, using the configuration mentioned in Section V.

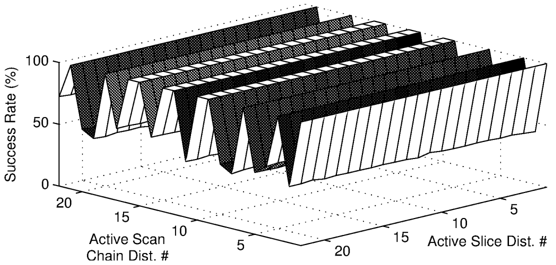

Fig. 5. Success rate of the attack on Adaptive Scan for 24 active scan chains and 24 active slices (Distributions from Table I)

VII. Scan Attack on Cadence OPMISR

A. Introduction to Cadence OPMISR

OPMISR (On-Product MISR) is one of the main features included in the Cadence Encounter Test toolkit [13]. The essential part of OPMISR is space compaction employing XOR trees against which DSA in [6] is effective. OPMISR has an optional feature of X-state handling using static X-masking. Although X-masking is used for testing purposes, it can enhance security by reducing the observability of the internal registers in a design. Time compaction with Multiple Input Signature Registers (MISRs) is another optional feature provided by OPMISR. We also provide results of scan attack on combined XOR-tree, static X-masking, and MISR structures.

B. Description of the Attack

As in Section VI, the attack consists of two stages. First, all 256 possible values are given to the first byte of the plaintext and the corresponding test outputs are collected. In the second stage, the test outputs are paired depending on the chosen XOR difference and a key guess is made depending on the Hamming distance between output pairs. This two-stage attack is repeated multiple times for different mask values, so that the correct key guess eventually overwhelms the incorrect key guesses, and becomes the top key candidate. Therefore, an attack is deemed successful only if the top key candidate is the correct one. A final note on the attack is that since it is repeated for a number of random test inputs, only 3 different XOR differences are used in analysis to reduce the execution time of the attack.

In Fig. 6, the trend in success rate in comparison to the number of test inputs used is presented. It can be observed from the figure that the attacks on masking schemes are more successful if a larger collection of random test inputs are used to mount an attack.

C. DSA on XOR Compaction with Static X-Masking

The attack success not only depends on the number of active slices, but also on the number of active scan chains. This

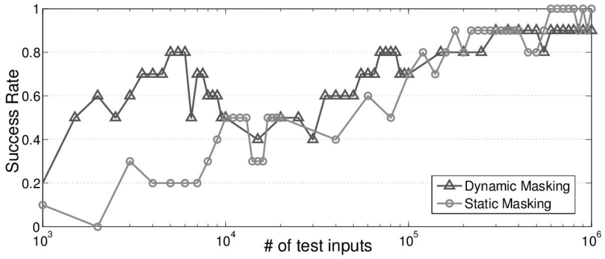

Fig. 6. Change in success rate with respect to # of test inputs (or masks) used for the attack.

is due to the fact that a smaller number of scan chains can be covered more frequently since a static mask with a lower Hamming weight would be sufficient to include all of them. For instance, let there be c active scan chains in the design. Statistically, once in 2^c different masks, all KFFs will affect the test outputs, therefore giving a better chance of mounting a successful attack. Hence, the smaller number of active scan chains present in the design, the better the success rates will be for the attack. Also it should be noted that even when the number of active scan chains and active slices are fixed, the attack success will be affected by the distribution of KFFs over the scan structure.

Table III shows DSA success rates in percentages for varying active scan chains and active slices. The results illustrate that there is a substantial fall in the success rates with decreasing number of active slices when the number of active scan chains is kept fixed. However, when the number of active slices is fixed, there is a small increase in success rates as the number of active scan chains is reduced. This shows that the dependency of DSA is more on active slices than on active scan chains. This observed behaviour is due to KFFs on the same slice being XORed together and therefore reducing the observed Hamming distance value in the test output.

TABLE III
DSA Success rates for static masking for different distributions

#Active Scan Chains = 32		**#Active Slices = 32**	
#Active Slices	**Success Rate**	**#Active Scan Chains**	**Success Rate**
32	81.91%	32	81.91%
31	77.48%	31	83.26%
30	72.02%	30	83.76%
29	66.79%	29	86.07%
28	63.21%	28	86.14%
27	56.88%	27	88.19%
26	53.24%	26	88.03%
25	49.21%	25	88.30%
24	44.39%	24	89.82%
23	41.25%	23	91.09%
22	37.81%	22	91.77%
21	33.19%	21	92.28%
20	30.39%	20	92.60%
19	27.94%	19	93.13%
18	25.29%	18	93.78%
17	22.63%	17	94.49%
16	20.75%	16	94.22%

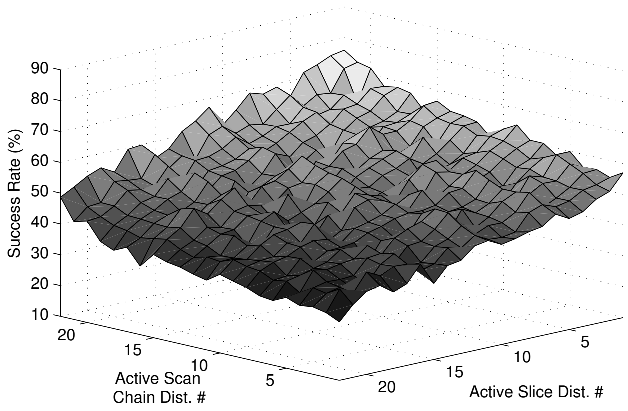

Fig. 7. Success rate of the attack on OPMISR (space compaction only) for 24 active scan chains and 24 active slices (Distributions from Table I)

Fig. 7 shows the change in the success rate of the attack with different distributions having the same number of active scan chains and active slices. There are two important observations which can be made from Fig. 7. Firstly, distribution of KFFs over active slices will affect the success rate as it determines how fast the information is processed by the compactor. For instance, when the distribution 22 in Table I is considered for the active slices, it is clear that 9 KFFs will be processed in one test clock. However, for distribution 4, it would take 3 test clocks to process the same amount of information, which means less information is lost and eventually the observed Hamming distance after the compactor is more likely to be successfully attacked. Hence, it is realistic to expect a change in success rates inversely proportional to the increasing number of KFFs on a single slice. Similarly, when 9 KFFs are grouped on a single scan chain, information on those KFFs will be included with probability $\frac{1}{2}$. If we again compare it to distribution 4 for the active scan chains, the same KFFs would be included in the output with probability $\frac{1}{8}$. Therefore, the wider the distribution of KFFs over the active scan chains, the lower success rates one will get when mounting the attack.

Although the above arguments can provide a designer with options towards designing more secure scan chain structures with the same tools that he is using, Fig. 6 should always be kept in mind when claiming security. It should be noted that, it is possible to make the attacker's job harder, but it is only a matter of resources for the attacker to recover the key when masking schemes are considered. Applying the attack once with 1000 inputs on a design, with a random distribution of KFFs, takes around 0.96 seconds using the configuration mentioned in Section V.

D. DSA on XOR Compaction, Static X-Masking and OPMISR

For the sake of completeness, the same attack is applied for a design which uses both static X-Masking and MISRs, therefore having both time and space compaction at the same time. When the attack is applied using the distribution (32 active slices and 16 active scan chains) on which the attack described in Section VII-B is most successful, success rate is reduced from 94.22% to 3.55%. Although this gives an idea about the security of combined time and space compaction, it should be noted that the attack does not include any method to exploit the MISR structure.

Though the attack in [8] is claimed to be successful against AES in the presence of MISR-based time compaction, it relies on the assumption that the MISR register is observable after each scan clock. In a real-life case, it may be possible to make the parallel outputs of a MISR visible during testing, however it would raise two major issues. Firstly, the gain in using MISRs after the scan structure diminishes as they are supposed to compress the golden value to make the testing procedure more efficient. Secondly, if MISR content is available at all times, then this implies that the complete scan chain contents are available to the attacker. Therefore using the method proposed in [1] would suffice to recover the key. In this work, we focus only on the output signatures of MISRs, which we believe to be a more realistic assumption, and observe that attacking such a system would not be possible by using simple DSA techniques.

VIII. Scan Attack on Mentor Graphics Embedded Deterministic Test(EDT)

A. Introduction to Embedded Deterministic Test

Mentor Graphics test compression tool, Tessent TestKompress employs Embedded Deterministic Test (EDT) [21] [22]. Similar to test compression tools from Synopsys and Cadence, it uses XOR trees for space compaction. However, it deals with X-states in a different manner through the use of a dynamically changing mask, which provides more flexibility and easy of applicability, since knowledge of scan chains which have a higher probability of occurrence of X-states (as in static X-masking) is not required. This method makes use of a ring generator (similar to a circular linear feedback shift register) together with a phase shifter to produce independent inputs for each scan chain. The scan outputs are compacted together with an XOR tree which is used right after an X-masking operation. X-masking is done with AND gates where the enabling inputs are generated on-the-fly through a pattern mask decoder. The masking logic varies based on a special EDT clock and test inputs.

In [15], it is claimed that Tessent TestKompress provides inherent security as the scan inputs and outputs are compressed and rendered useless for an attacker. A theoretical security analysis of EDT is provided in [14]. However, the security claim in that work is based on the assumption that a scan attack requires knowledge of the internal test structure and secret registers, which may not always be valid.

B. DSA on Dynamic X-Masking

Similar to the attack described in Section VII, the scan attack principle remains the same. However, the only difference is that the mask used in the emulation of the attack is dynamically changing depending on the test input. First all 256 possible values are given to the first byte of the plaintext and the first round outputs are collected. Then, the outputs are paired depending on the selected input XOR difference and a key guess is made. The attack is again repeated for a number of times to be able to distinguish the correct key candidate from others.

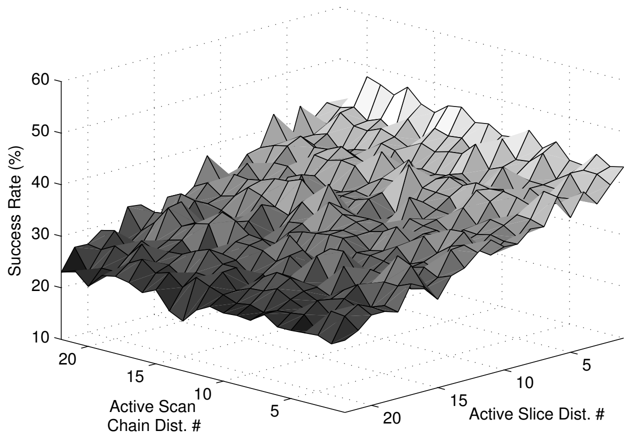

Fig. 8. Success rate of the attack on EDT for 24 active scan chains and 24 active slices (Distributions from Table I)

Table IV shows DSA success rates in percentages for varying active scan chains and active slices. Similar to the static X-masking case presented in Section VII, the results illustrate that there is a substantial fall in the success rates with decreasing number of active slices when the number of active scan chains is kept fixed. However, when the number of active slices is fixed, the success rate stays almost the same. This is so since the mask is assumed to be updated at each clock, and therefore the probability of all KFFs affecting the output only depends on the total number of KFFs, as each KFF in a different slice should be selected at each clock individually. This shows that the dependency of DSA for dynamic X-Masking is much more pronounced for varying active slices, and there is hardly any dependency on active scan chains. This can act as a guideline for the security design engineer to place the KFFs on specific slices during DfT insertion.

TABLE IV
DSA Success rates for dynamic masking for different distributions

#Active Scan Chains = 32		**#Active Slices = 32**	
#Active Slices	**Success Rate**	**#Active Scan Chains**	**Success Rate**
32	82.18%	32	82.18%
31	77.00%	31	81.87%
30	71.73%	30	81.28%
29	66.66%	29	81.37%
28	62.37%	28	82.52%
27	57.49%	27	81.77%
26	54.10%	26	81.60%
25	49.99%	25	81.74%
24	44.14%	24	81.75%
23	40.00%	23	81.48%
22	37.62%	22	81.45%
21	32.49%	21	82.58%
20	29.88%	20	81.64%
19	27.97%	19	81.19%
18	24.78%	18	80.85%
17	22.19%	17	81.88%
16	20.53%	16	81.90%

From Fig. 8, it can be inferred that the argument in Section VII regarding the distribution of KFFs over active slices is also valid for the case of dynamic masking. However, the KFF distribution over the active scan chains will have no effect since the mask is assumed to be updated at each test clock. Therefore, even if the KFFs are grouped in the same active scan chain, it is equally difficult for all of them to be picked as in distribution 1 in Table I. However if the mask update clock is slower than the test clock, then we would see a slight increase in success rate while more and more KFFs are grouped at the same scan chain (as in distribution 22 in Table I). Therefore, the cases presented in this work cover the two extreme possibilities when the mask is not updated and when mask update clock is the same as the test clock, giving an idea of the cases in between. Applying this attack once with 1000 inputs on a design with a random distribution of KFFs takes around 0.88 seconds using the configuration mentioned in Section V.

IX. Countermeasures

In Section VII-D, we show that combined space and time compaction can act as a suitable scan attack countermeasure implicitly achieved through the DfT structure. To the best of our knowledge, latest version of Cadence Encounter Test consisting of OPMISR+ is the only DfT tool which provides a combination of time compaction (MISRs) and space compaction (XOR-trees). Other EDA companies (Mentor, Synopsys, SynTest) have the capability to add a MISR to a design by means of scripting to get the OPMISR+ effect. However, in our paper, we have only considered the standard features provided in the commercial test tools, and not any custom extensions that can be added.

An explicit scan attack countermeasure is necessary to be integrated with the DfT structure generated by these tools. In this section, some of the existing explicit scan attack countermeasures are evaluated, and a new countermeasure is also proposed. To have a fair basis for comparing the countermeasures considered in this section, all simulations are run without X-masking or X-tolerant logic. Experiments are repeated 10000 times with three distinct XOR differences to get good statistics and only one test input is used per attack since there are no test input dependent elements in the proposed countermeasures.

A. Insertion of Inverters in the Scan Path

The technique is also known as the flipped scan tree architecture [23]. It involves dividing the scan chains into a number of sub-chains in the form of a scan tree. Sub-chains are parts of complete scan chains which may be connected in a random order in order to make extraction of useful information from the scan outputs difficult for an attacker. Inverters are inserted in front of the scan flip-flops in some secret locations. These locations are known only to the designer and the tester, but not to an attacker. However, as the position of the inverters in the scan path is fixed, DSAs are immune against this countermeasure and the same attack principle is applicable.

B. Partial Scan

This approach (also referred to as balanced secure scan) aims to protect non-scan registers by employing a test controller that enables the test mode only when an authentication succeeds [16]. Only a few flip-flops belonging to the secret registers are included in the scan chains. Further confusion is added to the kernel wherever a secret register is inserted in the scan chain.

To emulate the effect of this countermeasure in our software implementation of the attack, we removed some of the KFFs from the two-dimensional array used to represent the scan chains. Then DSA is performed together with the test compression structures. The distributions in our experiments have 25%, 50% and 75% of the KFFs removed from the scan chains. The average scan attack success results that we obtained for these distributions are 24.46%, 3.01% and 0.82% respectively. The attack is repeated 10000 times, with different KFFs being blocked at each try. Therefore, the success rates presented here shows the effect of partial scan countermeasure when a randomly selected group of KFFs are blocked in a design.

Although the results suggest that the partial scan countermeasure is a good way to mitigate DSA on an AES design, the cost of the countermeasure should also be taken into account. We implemented the partial scan countermeasure following the structure of Figure 3 in [16]. The test controller along with the other modules were implemented in HDL and a 32-bit LFSR is used to derive the signals which are used for blocking a group of KFFs. Implemented this way, the partial scan countermeasure takes 341.5 GE when synthesized using a Faraday 130 nm technology library. This corresponds to an area overhead of 1.95% for the partial scan countermeasure when added to the AES design in [11], which occupies an area of 17484.25 GEs.

C. Scan Chain Scrambling

This countermeasure proposes to divide each scan chain into multiple scan elements and the order of connections of the scan elements is controlled through the scan chain scrambler. When the scan mode has been reached securely, the scan chain elements are arranged in a predetermined order [17]. However, in insecure mode, the order of the scan chain elements keeps changing at a certain frequency.

As the resulting KFF distribution in insecure mode is expected to be randomised by this countermeasure, 10000 random distributions of KFFs are used to simulate the behaviour of the countermeasure. The same scan attack principle is applicable. Full information is still visible as all KFFs contribute to the test outputs even after randomising the KFF distributions. Hence, for each distribution, the attack successfully recovers the correct key.

D. Masking

In [7], two masking countermeasures for protecting AES against scan attacks were proposed. These masking schemes are similar to the countermeasures used to protect against differential power analysis (DPA) side-channel attacks. The first method is based on masking the round-register data. The mask can be added to all the 128 round-register FFs, and then removed from the encrypted value before executing the next AES round. The masking is effective only during testing and is completely transparent in functional mode. Alternatively, to reduce area requirements, the mask can be applied on a single FF per slice to be protected. In this manner, the parity of the whole slice is affected by the mask and its effects cannot be eliminated by the attacker. The second method works by modifying the response compactor. This is implemented by masking the parity bitstream on the output of the test response compactor instead of the data before being captured in the scan chain. It makes use of an enhanced LFSR (eLFSR) that can function either as a simple register or as an LFSR. In functional mode, the eLFSR is loaded with a value matching with the round-register (a 128 bit value unknown to the attacker), or with any other value depending on the input message and part of the secret key. In test mode, the eLFSR provides the mask to the observed stream.

The mask register countermeasure has an area overhead of 5.9%, while the countermeasure of modifying the response compactor has an overhead of 4.5% for the AES design used in the paper [7] and involves storing a secret key. The masking countermeasure can also be extended to ciphers other than AES, for instance RSA or ECC, by masking the intermediate registers storing the results of modular square and multiply operations or point doubling and point addition results respectively.

There are also other scan attack countermeasures such as the Lock and Key Technique [24], Design for Secure Test [25] and resetting crypto chip in test mode [17] that were not evaluated experimentally in this work. These schemes are briefly discussed qualitatively here. The Lock and Key technique [24] uses a plaintext key for comparison to unlock the finite state machine used for randomizing the order in which the scan elements are connected. In case the communication link is not secure, an attacker can observe this key. The Design for Secure Test [25] which checks the parity of AES rounds is an ad-hoc solution for AES designs with a completely unrolled structure (having high area requirements), limiting its applicability to other designs. Though the scheme involving resetting the crypto chip and removing all traces of cryptographic execution in test mode [17] might provide a high level of security, it is not applicable for implementations where some secret data needs to be stored on-chip.

E. Proposed Countermeasure

The existing scan attack countermeasures aim at securing uncompacted scan chain structures. However, as described in the previous sections, practical scan structures consist of compaction techniques which lead to a loss of observability of internal scan chains. Hence, they may be extended into a countermeasure. In this regard, we propose a new scan attack countermeasure based on randomization of the compactor outputs.

Fig. 9 shows the proposed noise injector countermeasure. It consists of a Linear Feedback Shift Register (LFSR), a True Random Number Generator (TRNG) and some basic logic gates. For the particular case of Noise Injector depicted in the figure, two OR gates, one NOR gate, one AND gate, and one XOR gate are required. The LFSR makes a pseudo-random selection of the test cycles when random noise is injected into the compactor outputs. More specifically, the compactor output is flipped if ‘A’ in Fig. 9 is 1. The signal ‘A’ is generated through a combination of TRNG and the LFSR outputs, making it unpredictable for an attacker. Hence, the compactor outputs becomes random and cannot be exploited by DSA.

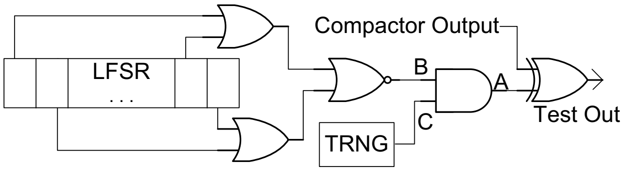

Fig. 9. Noise Injector countermeasure for Injection Freq. Factor of 16

Injection frequency factor (IFF) refers to the rate at which noise is injected into the compactor outputs, calculated as a factor by which the test clock frequency is divided. As an example, let us consider the LFSR structure in Fig. 9 having a IFF of 16. Let us assume that the state bits of the LFSR are completely random, with equal probability of occurrence of 1s and 0s ($\frac{1}{2}$). An OR gate will give an output of 0 with probability of $\frac{1}{4}$. Hence the probability that both inputs of the NOR gate are 0 (to give an output of 1) is $\frac{1}{16}$ (as the events are independent of each other). Only when the output ‘B’ of the NOR gate is 1, random noise is injected into the compactor output, as the output of the TRNG and ‘B’ are connected through the AND gate. Similarly, to obtain an injection frequency factor of 4, one of the OR gates may be removed and the other input of the NOR gate connected to ground (0 logic), resulting in a probability of $\frac{1}{4}$ for obtaining a 0 at both inputs of the NOR gate. Similar simple structures can be designed for all the other possible cases. The gate combination to obtain an injection frequency factor which is not a power of 2 is also straightforward, though involving more number of gates. For instance, to obtain an injection factor of 5, we can construct a 4-input truth table which gives five 1s and eleven 0s at its output. One such possible expression would be $(\neg I_1 \wedge \neg I_2 \wedge I_3) \vee (I_1 \wedge I_2 \wedge \neg I_3) \vee (I_0 \wedge \neg I_1 \wedge I_2 \wedge I_3)$, where I_0, I_1, I_2 and I_3 represent the states at the four tap points of the LFSR.

We have also analyzed two ways of attacking this system to determine its security. The first attack does not use any information about the countermeasure, therefore taking a black-box approach. The attack is applied just as it would be applied to any other countermeasure. Results for this scan attack on the noise injector is provided in the second column (Success Rate^1) of Table V. However, there is another way of attacking this countermeasure. An attacker can first give the same input to the crypto algorithm and the same test input to the test circuit. After collecting the outputs by repeating this procedure enough number of times, the attacker can figure out the points where TRNG corrupts the output. Later, DSA can be applied to the circuit with the same test input and these bits can be removed from the test output as they have the potential to corrupt the test output. We simulated the attack in software and the probability of such an attack to be successful for a random design is presented in the third column (Success Rate^2)

TABLE V
CHANGE IN SUCCESS RATE VS HOW FREQUENTLY A RANDOM BIT IS XORED TO THE COMPACTOR OUTPUT

Injection Freq. Factor	Success Rate[1]	Success Rate[2]
16	63.33%	81.37%
15	59.17%	78.70%
14	54.37%	75.53%
13	49.34%	72.79%
12	43.78%	68.93%
11	37.71%	66.31%
10	32.76%	60.57%
9	28.52%	56.99%
8	23.55%	52.53%
7	18.77%	47.70%
6	15.05%	40.62%
5	11.91%	33.02%
4	8.32%	23.04%
3	5.51%	12.85%
2	2.93%	3.38%
1	0.90%	0.00%

of Table V. As is evident from the table, success rate of the attack decreases drastically when noise is injected at a higher frequency. Attack success reduces to only 0.9% when noise injected at the same frequency as the test clock.

Test coverage of the circuit is not affected by the proposed countermeasure with the following conditions:

- The LFSR structure (feedback polynomial, output points and seed value) should be known to the tester.
- The test cycles should be ignored when signal ‘B’ is 1 (Fig. 9).
- For achieving complete test coverage, the test patterns for ignored test cycles should be repeated until signal ‘B’ becomes 0 (sustained vector technique).

The knowledge of the TRNG outputs is not required by the tester, as the test cycles when signal ‘B’ = 1 are always ignored irrespective of signal ‘C’. We suggest to have dedicated test outputs while incorporating our proposed countermeasure. In case of pin constrained applications, the test outputs may be multiplexed with some of the primary outputs. To make the proposed scheme compatible for such applications, an additional control circuit is required which configures the whole cryptographic circuit based on its mode of operation. In normal mode, the noise injector is not connected to the compactor outputs, while in test mode it is connected. This could be realized by multiplexers controlled by the mode selection input pin.

A TRNG is used instead of a LFSR for generating the random input ‘C’ as a LFSR has a linear structure which is prone to cryptanalysis if the seed and feedback polynomial is known or if the LFSR does not have sufficient length (incurring high area overhead). A TRNG has much higher unpredictability property. An implementation based on Fibonacci and Galois Ring Oscillators is presented in [26].

Compared to some of the countermeasures mentioned in the previous section, our proposed scheme has lower area overhead, as we are utilizing the existing test compression infrastructure. Our noise injector countermeasure requires an area of 106.75 Gate Equivalents (GEs) incurring an overhead of 0.61% over the table-lookup based S-box AES implementation in [11] (which needs 17484.25 GEs) with implemented DfT, using a Faraday 130nm library and synthesized using Synopsys Design Compiler version C-2009.06-SP3. The area requirement is less than one-third of that required for the partial scan countermeasure presented in Section IX-B. A representative area requirement for Ring Oscillator based TRNGs is 0.0016 sqmm (or 200 GEs) as presented in [27] (where for UMC 180 nm employed in the paper, 125 KGEs is generally contained in one sqmm). Hence, the overhead of the noise injector countermeasure with an actual TRNG will be around 1%. The TRNG may also be part of a Cryptographic SoC as an on-chip random number generator. In such a case, it would not require any additional hardware resources. Test application time will however increase by a factor of 1/IFF.

We present here a brief comparison with the weighted (biased) pseudo-random (WPR) test pattern generation schemes [28] employed in older Logic BIST solutions before the advent of test data compression. WPR approach is based on two key steps. Firstly, bit-fixing is used in which certain idler register bit positions in the scan chains are assigned to hold fixed values for certain portions of the test time. This is followed by Biased pseudo-random (PR) testing, consisting of applying biased PR test patterns in the variable idler register bits. Our noise injection technique is different from this approach as we do not have fixed bit positions in the LFSR deciding the test cycles where random noise from the TRNG is injected. We only use certain portions of the LFSR state to derive the noise injection point.

X. CONCLUSIONS

In this work, security of industrial test compression schemes against differential scan attacks has been evaluated in detail for the first time. Scan attack results for all the three major DfT tools from Synopsys, Cadence and Mentor Graphics are presented. It is demonstrated that space compression with X-handling logic is vulnerable to scan attacks, whereas time compaction acts as a strong countermeasure against scan attacks. The most well-known scan attack countermeasures are investigated for their security vulnerability and attack success results are presented. A new noise injector countermeasure is proposed and its security properties are analyzed. Future work would be investigate the scan attack susceptibility of other time compression techniques, such as Syntest Virtual scan and Ultra Scan which uses Time Division Demultiplexer(TDDM) and Multiplexer(TDM).

REFERENCES

[1] B. Yang, K. Wu, and R. Karri, “Secure scan: A design-for-test architecture for crypto chips,” *IEEE Trans. Comput.-Aided Des.*, vol. 25, no. 10, pp. 2287–2293, Oct. 2006.

[2] R. Nara, K. Satoh, M. Yanagisawa, T. Ohtsuki, and N. Togawa, “Side-channel attack against rsa cryptosystems using scan signatures,” *IEICE Trans. Fund. Elec. Comm. and Comp. Sc.*, no. E 93A(12), pp. 2481–2489, 2010.

[3] R. Nara, K. Satoh, M. Yanagisawa, T. Ohtsuki, and N. Togawa, “Scan-based attack against elliptic curve cryptosystems.” *in Proc. ASPDAC*, pp. 407–412, 2010.

[4] J. Da Rolt, A. Das, G. Di Natale, M.-L. Flottes, B. Rouzeyre, and I. Verbauwhede, “A new scan-attack on RSA in presence of industrial countermeasures,” *in Proc. COSADE*, vol. 7275, pp. 89–104, 2012.

[5] J. Da Rolt, A. Das, G. Di Natale, M.-L. Flottes, B. Rouzeyre, and I. Verbauwhede, “A new scan attack on elliptic curve cryptosystems in presence of industrial design-for-testability structures,” *in Proc. DFT*, pp. 43–48, 2012.

[6] J. Da Rolt, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, “New Security Threats Against Chips Containing Scan Chain Structures,” in *in Proc. HOST*, 2011, pp. 105–110.

[7] J. Da Rolt, G. Di Natale, M.-L. Flottes, B. Rouzeyre, “Scan attacks and countermeasures in presence of scan response compactors,” in *in Proc. IEEE ETS*, 2011, pp. 19–24.

[8] J. Da Rolt, G. Di Natale, and B. Flottes, M-L. Rouzeyre, “Are advanced dft structures sufficient for preventing scan-attacks?” *in Proc. IEEE VTS*, pp. 325–336, 2012.

[9] B. Ege, A. Das, S. Ghosh, and I. Verbauwhede, “Differential scan attack on aes with x-tolerant and x-masked test response compactor,” *in Proc. IEEE Euromicro Conf. DSD*, pp. 545–552, 2012.

[10] J. Daemen and V. Rijmen, *The Design of Rijndael*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2002.

[11] “http://rijndael.ece.vt.edu/gezel2/examples.html#aes,” *Gezel Hardware/Software codesign Environment - Advanced Encryption Standard GEZEL Code*.

[12] P. Wohl, J. A. Waicukauski, S. Patel, and M. B. Amin, “X-tolerant compression and application of scan-atpg patterns in a bist architecture,” *in Proc. IEEE ITC*, pp. 727–736, 2003.

[13] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability (Systems on Silicon)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

[14] C. Liu and Y. Huang, “Effects of embedded decompression and compaction architectures on side-channel attack resistance,” in *in Proc. IEEE VTS*, 2007, pp. 461–468.

[15] Mentor Graphics, “Silicon test and yield analysis whitepaper - high quality test solutions for secure applications,” Apr. 2010.

[16] M. Inoue, T. Yoneda, M. Hasegawa, and H. Fujiwara, “Partial scan approach for secret information protection,” *in Proc. IEEE ETS*, pp. 143–148, 2009.

[17] D. Hely, M.-L. Flottes, F. Bancel, B. Rouzeyre, N. Berard, and M. Renovell, “Scan design and secure chip [secure ic testing],” in *in Proc. IEEE IOLTS*, 2004, pp. 219–224.

[18] B. Yang, K. Wu, and R. Karri, “Scan based side channel attack on dedicated hardware implementations of data encryption standard,” *in Proc. IEEE ITC*, pp. 339–344, 2004.

[19] B. Yang, K. Wu, and R. Karri, “Secure scan: A design-for-test architecture for crypto chips,” in *in Proc. IEEE/ACM DAC*, 2005, pp. 135–140.

[20] P. Wohl, J. A. Waicukauski, R. Kapur, S. Ramnath, E. Gizdarski, T. W. Williams, and P. Jaini, “Minimizing the impact of scan compression,” in *in Proc. IEEE VTS*, 2007, pp. 67–74.

[21] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, K.-H. Tsai, A. Hertwig, N. Tamarapalli, and G. Mrugalski, “Embedded deterministic test for low cost manufacturing test,” in *in Proc. ITC*, 2002, pp. 301–310.

[22] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, “Embedded deterministic test,” *IEEE Trans. Comput.-Aided Des.*, vol. 23, no. 5, pp. 776–792, May 2004.

[23] G. Sengar, D. Mukhopadhayay, and D. Roy Chowdhury, “An efficient approach to develop secure scan tree for crypto-hardware,” in *in Proc. ADCOM*, 2007, pp. 21–26.

[24] J. Lee, M. Tehranipoor, and J. Plusquellic, “Securing designs against scan-based side-channel attacks,” *IEEE Trans. Depend. and Secure Comput.*, vol. 4, no. 4, pp. 325–336, Oct.-Dec. 2007.

[25] Y. Shi, N. Togawa, M. Yanagisawa, and T. Ohtsuki, “Design for secure test - a case study on pipelined advanced encryption standard,” in *in Proc. IEEE ISCAS*, 2007, pp. 149–152.

[26] J. D. Golic, “New methods for digital generation and postprocessing of random data,” *IEEE Trans. Comput.*, vol. 55, no. 10, pp. 1217–1229, Oct. 2006.

[27] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, “A high-speed oscillator-based truly random number source for cryptographic applications on a smart card ic,” *IEEE Trans. Comput.*, vol. 52, no. 4, pp. 403–409, Apr. 2003.

[28] M. AlShaibi and C. R. Kime, “Fixed-biased pseudorandom built-in self-test for random pattern resistant circuits,” *in Proc. IEEE ITC*, pp. 929–938, 1994.

Amitabh Das received the B.Tech degree in Electronics and Communication Engineering from the University of Kalyani, India, in 2003, and the M.Tech degree in Instrumentation and Electronics Engineering from Jadavpur University, Kolkata, India, in 2009. He is currently a Ph.D. candidate at ESAT/COSIC, KU Leuven, Belgium. His research interests include secure DfT, electronic design automation, hardware cryptography, embedded systems, and hardware/software co-design.

Barış Ege received his BSc degree in Mathematics and MSc degree in applied mathematics and cryptography from the Middle East Technical University (METU), Turkey, in 2007 and 2010 respectively. Currently, he is working as a PhD student in Digital Security Group - Institute for Computing and Information Sciences at Radboud Universiteit Nijmegen, The Netherlands. His research interests are in lightweight cryptography, secure testing and side-channel analysis.

Santosh Ghosh He obtained his M.S. and Ph.D. in 2008 and 2011 respectively from the Department of Computer Sc and Engg, Indian Institute of Technology Kharagpur, India. He was a Post-doctoral Researcher at ESAT/COSIC, KU Leuven, Belgium, during 2011-2012. He is presently working at the Security Center of Excellence (SeCoE), Intel Corporation, United States. His research interests include cryptography and network security, VLSI design of cryptosystems, side-channel analysis, and secure testing.

Lejla Batina is an assistant professor at Radboud University Nijmegen in The Netherlands and a post-doctoral researcher at ESAT/COSIC, KU Leuven. She received her M.Sc. degree in Mathematics from the University of Zagreb, Croatia in 1995 and Ph.D. degree in engineering from the KU Leuven in 2005. Her research interests include efficient arithmetic for cryptographic algorithms, secure implementations of cryptographic algorithms, side-channel security, lightweight cryptography, sensor networks, etc.

Ingrid Verbauwhede received the electrical engineering degree and PhD degree from the KU Leuven, Belgium, in 1991. She is currently a professor at the KU Leuven and an adjunct professor at UCLA. At KU Leuven, she is a co-director of the Computer Security and Industrial Cryptography (COSIC) Laboratory. Her research interests include circuits, processor architectures and design methodologies for real-time embedded systems for security and cryptography.