IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 2021, date of current version 2021 00, 0000.

Digital Object Identifier 10.1109/ACCESS.20xx.DOI

From Cryptography to Logic Locking: A Survey on The Architecture Evolution of Secure Scan Chains

KIMIA Z AZAR[1], HADI M KAMALI[1], HOUMAN HOMAYOUN [2], and AVESTA SASAN[1]

[1]Electrical and Computer Engineering Department, George Mason University, Fairfax, VA 22030 USA
(e-mail: {kzamiria, hmardani, asasan}@gmu.edu)
[2]Electrical and Computer Engineering Department, University of California Davis, Davis, CA 95616 USA
(e-mail: {hhomayoun}@ucdavis.edu)

Corresponding author: Hadi Mardani Kamali (e-mail: hmardani@gmu.edu).

ABSTRACT

The availability of access to Integrated Circuits' scan chain is an inevitable requirement of modern ICs for testability/debugging purposes. However, leaving access to the scan chain OPEN resulted in numerous security threats on ICs. It raises challenging concerns particularly when the secret asset, like secret information, is placed within the chip, such as the keys of cryptographic algorithms, or similarly logic obfuscation key. So, to combat these threats, numerous secure scan chain architectures have been proposed in the literature to prevent any unauthorized access to the scan chain. They also keep the availability of the scan chain for testability/debugging. In this paper, we first show why a secure scan chain architecture is required when security primitives, like logic obfuscation, are in place. Then, we provide a holistic overview of all secure scan chain architectures starting from preliminary methods introduced when cryptography is in place and the adversary threat model is very limited. It is then followed by newer and more advanced methods introduced when logic obfuscation is in place and the adversary threat model is much stronger. Hence, we have more concentration on the architecture proposed more recently on logic obfuscation. We evaluate all secure scan chain architectures in terms of security and resiliency, testability/debugging time and complexity, and area/power/delay overhead.

INDEX TERMS Hardware Security, Cryptography, Logic obfuscation, Secure Scan Chain

I. INTRODUCTION

THE ever-increasing and huge cost of building and recurring maintenance of a new semiconductor foundry, has pushed many design houses to become fabless [1]. However, due to the lack of reliable monitoring and trustworthiness to offshore fabrication and testing processes, many security threats have emerged such as IP piracy, reverse engineering, counterfeiting, and IC overproduction [2]. To combat these hardware security threats, among various hardware *design-for-trust* (*DfTr*) techniques, logic obfuscation [3] engages a form of post-manufacturing programmability in the design, which ensures that the correct functionality will not be revealed without the programming value referred to as the *key*. In logic obfuscation, the correct key is either stored in a tamper-proof non-volatile memory (tpNVM) or generated using a physical unclonable function (PUF), and will be initiated after fabrication via a trusted party.

A. SAT ATTACK AS A TURNING POINT IN LOGIC OBFUSCATION

The introduction of primitive logic obfuscation solutions, such as RLL (random-based insertion) and SLL (no-sensitizable insertion) [3]–[6], was considering a reliable proactive solution against hardware security threats. However, in 2015, considering that the access to the scan chain of the circuit is *OPEN* for the test/debug purposes, a new and powerful attack based on Boolean satisfiability (SAT) was formulated that completely threatened the security of the existing logic obfuscation schemes [7], [8].

In the SAT attack, as an oracle-guided attack, the adversary has access to (1) one successfully reverse-engineered yet locked netlist, and (2) the activated/functional circuit with *OPEN* access to its scan chain architecture. By getting inspiration from the miter circuit in the formal verification process, in the SAT attack, a SAT solver is employed it-

eratively to rule out the incorrect keys. In each iteration of the SAT attack, the SAT solver finds a specific input, called discriminating (distinguishing) input patterns (DIP), that distinguish between two sets of keys, where at least one set of them is incorrect. By applying each DIP to the activated/functional circuit, the adversary is able to invalidate (rule out) the incorrect set(s) before the next iteration. This process continues until the SAT solver cannot find a new DIP. At this point, any key that generates the correct output for the set of found DIP is the correct key. In general, the SAT attack could eliminate all incorrect keys within a few iterations (a few minutes), leading to retrieving the correct functionality of the circuit.

B. POST-SAT OBFUSCATION COUNTERMEASURES

The main strength of the SAT attack comes from two important factors: (1) The pruning power of each DIP (each iteration of the SAT attack) is very high. In fact, the portion of incorrect keys that would be ruled out per each iteration is big leading to termination (successful de-obfuscation) within a few iterations (few minutes). (2) The access to the scan chain is *NOT* restricted, which helps the adversary to apply the SAT attack for each combinational logic part of the circuit separately (independently).

Considering these two factors, there are two main groups of countermeasures that have been introduced in the literature to show how a logic obfuscation technique could be built to defeat the SAT attack. One group tries to either weaken the pruning power of DIPs or introduce a solution that could not be formulated by the SAT attack (behavioral obfuscation). However, the main focus of the second group of countermeasures, on the other hand, is to restrict any unauthorized access to the scan chain to completely invalidate the possibility of engaging the SAT the attack.

C. WEAKENING/DISABLING THE SAT ATTACK

As mentioned previously, the first group of countermeasures tries to weaken/disable the SAT attack. Since having access to the scan chain does not provide any advantage for the adversary in this group of countermeasures, there is no concern for the designer to leave the scan chain architecture *OPEN* [15], [23]–[31], [37]–[41]. These countermeasures could be categorized into three main sub-groups: (1) point-function structure, (2) cyclic and behavioral obfuscation, and (3) routing obfuscation.

1) point-function (PF) structure

This sub-group exponentially increases the number of required SAT iterations, such as Anti-SAT, SARLock, and SFLL [23]–[25]. In such techniques, by using point-function structure, the SAT attack is able to rule out few incorrect keys (the best case is *ONE* incorrect key) per each iteration (per each DIP). Hence, similar to a brute force attack, the SAT attack faces an exponential runtime. However, these techniques suffer from various structural vulnerabilities that were eventually exploited to break them, such as SPS, removal, bypass, and FALL attack [9], [11], [42], [43]. Besides, these techniques suffer from very low output corruption. Due to the low output corruptibility, the error probability is exponentially small. Hence, the adversary could also rely on approximate key with an extremely low error rate, which could be found by approximate-based SAT attacks [10], [44].

2) Cyclic and Behavioral Obfuscation

Since the input format of the SAT solver is conjunctive normal form (CNF), which is a specific Boolean description of a problem, a SAT attack works perfectly fine if the logic obfuscation is of Boolean nature. Also, since the SAT solver works on directed acyclic graphs (DAG), it is only applicable to cycle-free circuits. Hence, in some recent obfuscation techniques, the behavioral properties of the circuit (such as timing) have been targeted that cannot be translated to CNF [29]. In some other techniques, the key-programmable cycles are added into the design, which traps the SAT solver in an infinite loop [26]–[28], [37], [38]. Although these solutions defeat the SAT attack, further investigation shows that this breed of obfuscation techniques is already broken using SMT attack [14], [45], timingSAT [46], and SAT-based attacks on cyclic obfuscation [12], [13], [47].

3) Routing Obfuscation

In the third sub-group, to weaken the SAT attack, the main aim of the obfuscation is to increase the complexity of the SAT problem, thereby the run-time of *each iteration* of the SAT solver would be increased substantially [15], [30], [31]. In such techniques, by exploiting the strength of symmetric routing structures, such as permutation networks or crossbars, the complexity of the SAT circuit per each iteration would be increased significantly. Some preliminary methods in this group have been broken already using newer attacks [15], [16], [48]. However, further studies on routing-based obfuscation techniques show that combination (twisting) of the routing obfuscation with logic obfuscation could help to reduce the vulnerability of routing-based obfuscation techniques against newer attacks [15], [48].

D. RESTRICTING UNAUTHORIZED SCAN ACCESS

As shown in Table 1, although all three previously discussed sub-groups explore the obfuscation solution space without the necessity of restricting access to the scan chain, all of them suffer from a newer attack or incur prohibited overhead. Hence, a few recent studies move towards the investigation of restricting scan access while the logic obfuscation is in place. In fact, since access to the scan chain is required to make the SAT attack applicable to obfuscated circuits, the second group of countermeasures tries to block any unauthorized access to the scan chain [19], [20], [32], [35], [49]–[51].

After restricting the access to the scan chain, the adversary has to rely on primary inputs/outputs (PI/PO) for de-obfuscation purpose. Also, since the SAT attack is no longer applicable when the scan chain access is restricted, such

TABLE 1: Comparison of State-of-the-art Logic Obfuscation Techniques: Why Securing Scan is Required?

	Defense	Corruptibility	Resilient against ⇓ SAT [7]	Removal [9]	Approx. [10]	FALL [11]	cycSAT [12]	icySAT [13]	SMT [14]	CP&SAT [15]	NNgSAT [16]	seq-SAT [17], [18]	Shift&Leak [19], [20]	Scan Unlock [21], [22]	Logic Overhead	Test/Implementation Issues: Limitation	Test Complexity	Test Time
point-func	SARLock [23]	low	✓	✗	✗	✓	✓	✓	✓	✓	✓	N/A	N/A	N/A	low	NO change	NO change	low
	Anti-SAT [24]		✓	✗	✗	✓	✓	✓	✓	✓	✓							
	SFLL [25]			✓		✗												
cyclic/ behavior	Cyclic [26]	high	✓	✓	✓	✓	✗	✗	✓	✓	✓	N/A	N/A	N/A	high	NO change	NO change	low
	SRCLock [27]						✓	✗	✓							NO change	NO change	low
	Mem-Cyclic [28]						✓	✗	✓							NO change	NO change	low
	DLL [29]						✓	✓	✗							NO change	NO change	low
routing	Cross-lock [30]	very high								✗	✗	N/A	N/A	N/A	very high	NO change	NO change	low
	Full-Lock [31]	very high	✓	✓	✓	✓	✓	✓	✓	✗	✗				very high			
	InterLock [15]	very high								✓	✓				high			
secure scan	EFF+RLL [32]	high	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✗	low	NO change	NO change	low
	FORTIS+SLL [33]											✗	✗	✓	moderate	test coverage	NO change	low
	R-DFS+SLL [34]											✗	✗	✓	moderate	1 extra test pins	NO change	low
	MSSD+RLL [19]											✓	✗	✓	moderate	1 extra test pins	key init	high
	DynScan+SLL [35]											✓	✓	✗	high	trusted tester*	NO change	low
	kt-DFS+SLL [20]											✓	✓	✓	moderate	3 extra test pins+	NO change	low
	DisORC+TRLL [36]											✓	✓	✓	moderate	trusted tester*	NO change	low

key init: In MSSD, since any form of shift operation has been blocked after key loading, a sys_rst is required for each test pattern, which results in extremely high test time.
*: For better efficacy in terms of functional test time/complexity, tester MUST be trusted to have access to the scan chain (Please refer to Section II)
+: kt-DFS requires THREE extra pins that extremely affects placement, routing, and die size of the chip obfuscated using this technique.

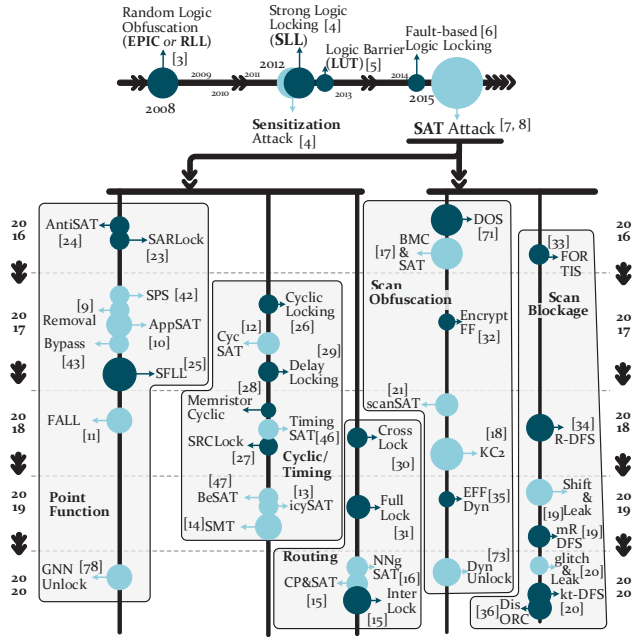


FIGURE 1: Attacks and Countermeasures in Logic obfuscation: Classification

techniques show that engaging a simple traditional logic obfuscation, such as SLL [4], significantly enhances the robustness against any form of de-obfuscation, particularly SAT-driven attacks. As shown in Table 1, the resiliency provided by this group of solutions is more reliable compared to that of the first group. As demonstrated in Fig. 1, this group of solutions could be categorized into two main sub-groups, called scan chain blockage and scan chain obfuscation.

E. THE SURVEY OVERVIEW

Design and implementation of secure scan chain architectures in the presence of logic obfuscation have received significant attention in recent years. In this survey paper, we will provide a holistic overview of this breed of countermeasures in terms of security, test time/complexity, and overhead. Since securing the scan chain was first originated in the presence of cryptographic engines, there exist some survey papers that review and evaluate such techniques [52]. However, unlike such literature review that focuses on the techniques that relied on an outdated and very limited threat model, in this survey paper, we first review and evaluate all secure scan chain architectures in crypto systems. Then, with much more concentration on security assets in logic obfuscation, we will step down further to review and evaluate secure scan chain architectures in the presence of logic obfuscation, which rely on a much stronger threat model. We will show that many of the primitive secure scan chain techniques in crypto systems fail to keep the resiliency against the adversary with the capability of the newer threat model.

The rest of this paper is organized as follows: In Section II, we first outline the necessity of having the scan chain architecture for test/debug purposes. In Section III, we then demonstrate the origin of scan chain restriction applied for the protection of sensitive data in cryptography, such as the protection of the key of symmetric/asymmetric encryption algorithms. Although the threat model/assumptions in this group of solutions are not valid (Not strong enough) at this time, in Section IV, we demonstrate the similarity of many of the current secure scan chain architectures while the threat model is relatively stronger. Secure scan chain architectures in the presence of the logic obfuscation could be categorized into two main groups. First, Section V shows the efficiency of *scan obfuscation* methods, while Section VI summarizes the *scan blockage* methods. In Section VII, we assess and compare all secure scan chain architectures either in crypto systems or obfuscated circuits in terms of security, test time/complexity, and overhead. We also draw a big picture regarding the future direction and opportunities in this topic. Finally, we conclude this survey in Section VIII.

II. IC TESTABILITY USING DFT-BASED TECHNIQUES

Having access to the internal parts of the circuit for test/debug purposes is fully inevitable in modern/complex ICs. Design-for-Testability (DFT)-based ASIC design provides this capability for the designers by adding scan (register) chain structures into the circuit. The DFT-based scan chain architecture has been widely used in most modern ICs. Even in cryptographic circuits, which have very sensitive information, such as encryption key, to have a high fault coverage, the test/debug step requires access to the scan chain to control and observe the internal states of the design-under-test (DUT). The full controllability and observability requirement in DFT-based (scan-based) testing, however, might pose security threats to ICs with security assets, such as obfuscated circuits that keep their own secret, i.e. the obfuscation key.

In modern ICs, different test capabilities must be provided to enhance the efficiency of validation/verification and test/debug flow. However, in the presence of security assets like logic obfuscation key, these capabilities require to be reconsidered from the security perspective as well as testability:

1) **Manufacturing (structural) test**: Almost all ICs require the manufacturing test (scan-based) prior to being released to the field. The scan-based manufacturing test is utilized for stuck-at faults. Manufacturing test patterns (and their corresponding outputs) could be generated using automated test pattern generators (ATPGs) at the design house (trusted side), and then could be sent out to the tester (could be outsourced and untrusted) for the evaluation of fault coverage. During manufacturing test, security assets like logic obfuscation key could be set to any arbitrary value (determined by ATPG). This option allows the designer to consider key values as extra primary inputs and set them as controlling values to maximize the fault coverage. Hence, when logic obfuscation (or any security asset) is in place, the access to the scan chain is not required to be limited for the structural test, and there is no need for having the correct key for detecting stuck-at faults.
2) **Full in-field (functional) test**: As its name implies, during the in-field functional test, a set of functional test vectors is required to check the correctness of the functionality of the design. It also could be done before releasing the chip into the field. Hence, when the logic obfuscation (or any security asset) is in place, the correct key is needed for the functional test. So, it is inevitably required that the design does not leak key information to its PO while it is in functional mode. Although the functional test could be done through PI/PO, and scan access is not necessarily required, many studies in the literature show that the efficiency of the in-field (functional) test could be significantly higher (in terms of performance and complexity) when the scan chains are available. For instance, during the functional test, it is required for each macro (IP) to be initialized to the desired state. Setting that state of a complex and huge circuit through PI becomes a major challenge and could potentially take millions of clock cycles to load the desired value in all internal FFs [34], [36], [53]. Thus, to avoid significantly increasing test time for functional testing and considering that tester time is a major contributor to the final cost of an IC, having access to the scan chain in massive-scale IC testing becomes an economically forced requirement. But we acknowledge that functional testing without having scan access and utilizing PI/PO is still possible and such an approach may be used for testing low-volume production yet sensitive hardware.

III. PRIMITIVE SECURE SCAN CHAIN ARCHITECTURES

In the last decade, there have been a number of scan-based attacks on various cryptosystems (such as DES [54], AES [55], [56], RSA [57] and etc.). Since scan chains directly reveal the internal state of the logic blocks, it is considered as a general threat to stream cipher, and attackers can use them to perform IP piracy. Hence, securing the scan chain architecture is required when a piece of secret information is placed within the DUT. Many primitive secure scan chain architectures are introduced for crypto engines with secret assets, like encryption keys. In this section, we evaluate these primitive methods which all relied on a weak and outdated threat model.

A. THREAT MODEL AND ASSUMPTION

The threat model and assumption in cryptosystems could be enumerated as follows:

1) The adversary is not able to de-package the chip and probe the internal signals.
2) The adversary is familiar with the crypto algorithm.
3) The adversary is able to run the circuit in normal (functional) mode and in scan (test) mode, and switch between the two modes at any clock cycle.
4) The adversary could access to the scan chain inputs/outputs (SI/SO). Hence, (s)he can shift data into internal scan chains through SI and receive the updated internal data for observation through SO.
5) The scan chain structure is unknown to the adversary.

Considering this threat model, various countermeasures have been proposed to combat scan-based attacks on crypto systems. At first, many of the countermeasures are limited to just manipulating the structure of the scan chain with relying on the assumption that the internal assessment through reverse-engineering or/and probing is not possible. However, more recent countermeasures engage scan key obfuscation to avoid this confined assumption.

B. FLIPPING THE SCAN CHAIN USING STATIC INVERTER NETWORKS

One of the early attempts to manipulate the scan chain is a flipped scan [58]. As shown in Fig. 2, a certain number of

inverters are statically inserted between randomly selected scan flip-flops (SFF), which change the scan data (initial state) during shifting in/out. However, the work in [59] proved that the positions of the statically inserted inverters could be determined easily by resetting all SFFs. Specifically, if the chip is reset, all SFFs are initialized to zero. Then, the adversary obtains the SO pattern by running the chip under test (shift) mode. The SO pattern would look like a series of 0's interleaved with a series of 1's because of the reversion of the inverters. By analyzing the SO pattern, the locations of the inserted inverters could be pinpointed.

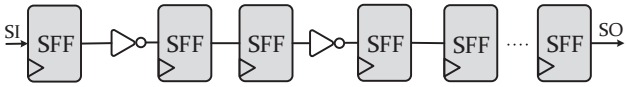


FIGURE 2: Manipulating the Scan Chain Structure using Statically Inserted Inverter Gates (Flipped Scan) [58].

C. FEEDBACK XOR IN SCAN CHAIN

To remove the possibility of pinpointing the location of manipulation in the scan chain structure, the work in [59] inserted certain feedback XORed signals within the scan chain. As shown in Fig. 3 (left part of the scan chain), one of the inputs to each XOR gate is the present input of the SFF from the one preceding it in the chain while the other input is the current output of the SFF. In this technique, the inversion would be data-dependent, and conditionally inverts the present input based on the past input. This configuration passes the reset-based attack on the flipped scan architecture, as in case of reset, SO pattern will be all zeros, i.e., the XOR gates become transparent.

After the introduction of the feedback XOR scan, the work in [60] showed that this scheme is still vulnerable because the adversary is able to determine the number and positions of the XOR gates in the scan chain. So, Double feedback XOR scan is proposed in [60] to overcome this weakness. As shown in Fig. 3 (right part of the scan chain), this scheme is very similar to the previous feedback XOR scan and the only difference is that the outputs of two downstream SFFs are fed back to the XOR gate.

The weakness of the double Feedback XOR scan is shown in [61], which introduces a new technique that deduces the locations of the inserted XOR gate in the double feedback XOR scan chain architecture. As a countermeasure, the author in [61] proposed random-based XOR scan architecture (rXOR) that is shown in Fig. 4. In this countermeasure, a MUX, whose selector signal is generated by a physical unclonable function (PUF), is inserted at the end of every double feedback XOR gate. Based on the PUF response, the MUX would select the input from the preceding SFF or double feedback XORed signal. Unlike all previous techniques that statically manipulate the structure of the scan chain, the introduction of MUX and PUF improves the randomness and security of this design. Since the PUF response is known to either the designer, or a valid end-user, or the tester, as expected

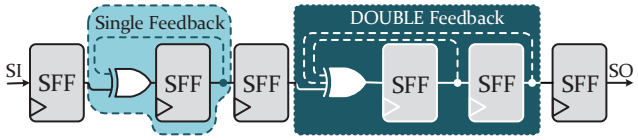


FIGURE 3: Manipulating the Scan Chain Structure using Single/*DOUBLE* feedback XORed Signals [59], [60].

responses to their challenges, the structure of the scan chain would be known. However, there is still a big limitation on the assumption that the tester must be a trusted party.

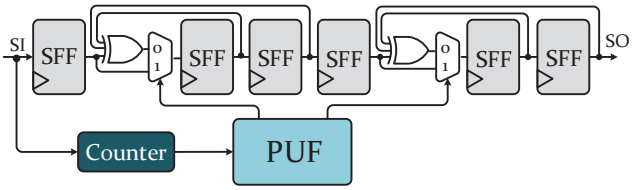


FIGURE 4: Manipulating the Scan Chain Structure using PUF-based feedback XORed Signals with Randomness [61].

D. STATE DEPENDENT SCAN FLIP-FLOP

Although PUF-based rXOR adds randomness into the manipulation mechanism, it still behaves statically, and the changes are only based on the PUF responses to its challenges. Hence, the work in [62] proposed the state dependent SFF (SDSFF), which is a secure scan chain architecture that is able to change the structure of the scan chain dynamically using temporal dependency and the position where inverters or XORs are inserted is changed virtually and dynamically. In SDSFF as shown in Fig. 5, based on *load* signal, the SFF value would be loaded into the latch. During the scan (shift) mode, the output of the latch would be XORed with the previous SFF before being shifted to the next SFF. The SDSFF can change the value of scan output using latch memorizing a past state of the SFF dynamically. To attack this SDSFF-based design, the adversary has to know the number of SDSFFs, the positions of SDSFFs, and the time of data change in latches. To conceal the timing of latches update, and to eliminate the external controllability of data loading into the latches, an improved SDSFF has been introduced, which updates each latch for every N clock cycles.

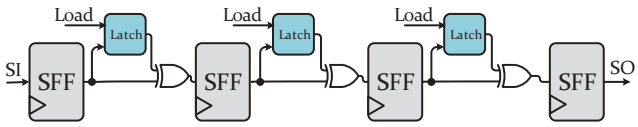
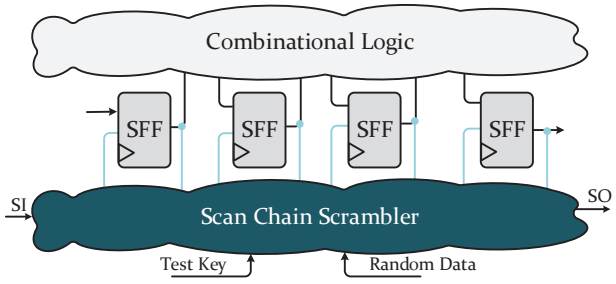


FIGURE 5: Dynamically Manipulating the Scan Chain Structure using State Dependent Scan Flip-Flop (SDSFF) [62].

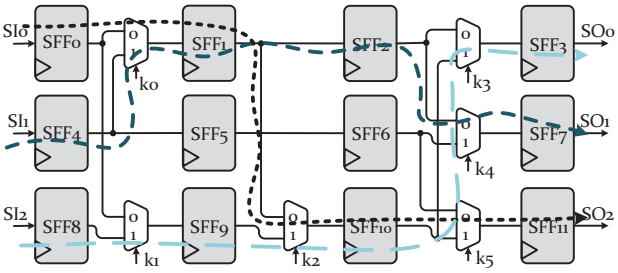
E. SCRAMBLED SECURE SCAN

The key-based manipulation of the scan chain structure (It could be called scan chain obfuscation) was first introduced in [63], where a key-based scrambling mechanism determines the order of SFFs. In key-based scan chain scrambling,

the SI/SO of the selected SFFs is permuted. To scramble the scan chains, as shown in Fig. 6, MUXes are added between selected SFFs, where the SI of the selected SFF_{ij} (located on scan chain j, scan slice i) is fed by the output of a MUX. The inputs to the MUX come from any two (or more) SFFs from the $(i-1)^{th}$ scan slice. The MUX selector is a key bit that controls the ordering of scan path fragments. A scrambling controller generates the control signals of the MUXes. During the test mode, a test key allows certifying the validity of the mode of operation. The scrambler controller reads this key and generates adequate control signals to connect the scan chain segment based on the correct sequence. In any other mode of operation, or when the test key is not valid, the scrambling controller sends random values to the multiplexer control inputs. However, since scan segments are connected, as the sub-chain length decreases, this technique will create significant logic and routing overhead.



(a) The Overview of Scrambled Scan Architecture



(b) Scrambling Implementation.

FIGURE 6: Key-based Scan Chain Scrambling. Correct paths: in green, Red, and Purple [63].

F. INTEGRATION OF TEST KEY WITH SCAN CHAIN

Another key-based low-cost secure scan (LCSS) solution was proposed in [64], in which the state of the scan chain is dependent on a test key that is integrated into all test vectors. As shown in Fig. 7, this design composes of three components: scan chain integrated with dummy flip-flops (DFF), key checking logic module (KCL), and random bit generator (RBG). DFFs are added allowing the tester/designer to use the same key for every test vector. DFFs are designed similarly to SFFs except that there is no connection to the combinational logic. DFFs must be checked concurrently by the combinational KCL, and if the test key fails to be checked by KCL (incorrect test key), the RBG will make the scan chain output unpredictable using randomness. The similar idea is also used in [65] and [66].

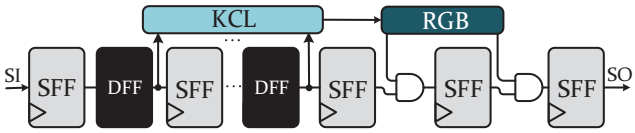


FIGURE 7: Integration of Test Key with The Scan Chain Structure [64].

G. DECOUPLING SENSITIVE DATA USING MIRROR KEY REGISTER

Decoupling the sensitive (critical) data from regular data was first introduced in [55], which is implemented using duplicated sensitive registers, called mirror key register (MKR). MKR prevents entering any sensitive data to the scan chain during test mode. By isolating the sensitive data using a finite state machine, this architecture distinguishes between the insecure state and the secure state of the scan chain. When the circuit is in the insecure mode, the MKRs operate as regular SFF, and test vectors would be scanned in and the test result would be scanned out. However, when the circuit is in secure mode, the MKRs load the secret-key information, and the contents of MKRs cannot be scanned out until being reset. As shown in Fig. 8, when Load_Key signal is 1, the input of the MKR is locked to the secret key, and any operation that writes to or scans the MKR is disabled.

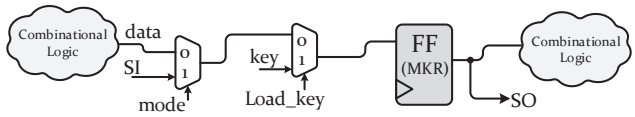


FIGURE 8: Decoupling the Sensitive Data using the MKR structure [55].

H. DIVISION OF SCAN CHAIN TO SUB-CHAINS WITH RANDOMNESS

In some secure scan chain architectures, the division of scan chains into several sub-chains has been introduced, in which the filling of the sub-chains would be controlled based on a random source. In sub-chain based scan architecture in [67], the scan chain is divided into smaller sub-chains of equal length. The important point in this architecture is that the test vectors are not sequentially shifted into each sub-chain but rather a linear feedback shift register (LFSR) performs a pseudo-random selection of a sub-chain to be filled.

Fig. 9 shows a general architecture for this technique. As shown, the LFSR value would be used as the controllers of the sub-chains, and as the selector of the MUX. When the circuit is in secure mode, the sub-chains would be selected based on a predictable and non-sequential order. However, when the circuit is in insecure mode, the user attempting to access the scan chain is considered untrustworthy until deemed otherwise with a correct test key. Unless the test key is entered and confirmed to be correct, the LFSR will

unpredictably select sub-chains, presenting the user with false information about the scan chain.

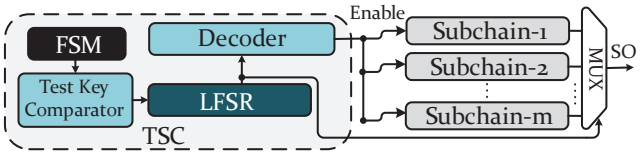


FIGURE 9: Sub-chains based secure scan [67].

Similarly, the work in [68] proposed a random order scan (ROS), which change the connection order of sub-chains virtually/dynamically. Fig. 10 shows the ROS architecture, in which a long scan chain is divided into several short sub-chains. All the sub-chains would receive the scan input from the same external SI. In ROS, only the selected sub-chain is activated, while the others are clock gates using enable signal. Hence, a controller is designed to generate enable signal allowing only one sub-chain to be activated at a time.

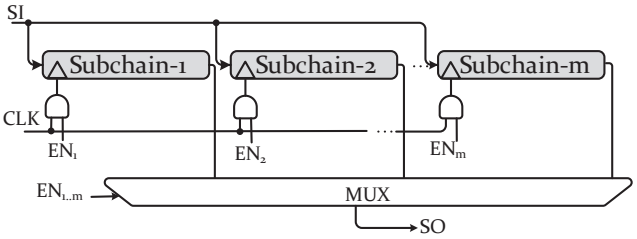


FIGURE 10: Random Order Scan [68].

However, the work in [69] has proven that designs by obfuscating scan chain order cannot provide sufficient security as predicted, and it is demonstrated that such countermeasures are vulnerable to signature attacks [57]. Signature attacks do not rely on the information of the scan chain order, and they show that as long as complete scan chain states are obtained, these designs may be cracked by attackers.

I. PARTIAL SECURE SCAN CHAIN ARCHITECTURE

By getting the benefit of combinational ATPG, a partial secure scan chain architecture has been introduced in [70], in which a particular set of SFFs are removed from the scan chain such that the resulting kernel, which refers to the sequential circuit without the scan chain, belongs to balanced-structures. Hence, the sequential kernel with delayless wires could be tested using combinational ATPG. However, this approach is only applicable to pipelined circuits with limited feedback and feedforward connections.

Although, this partial secure scan chain architecture limits test engineers from controlling and observing the internal states of the circuits, which reduces the testability coverage, the work in [71] proposed another partial secure scan design that addresses this issue. In this architecture, those SFFs that storing sensitive information would be removed from the full scan chain. Hence, the adversary is not able to get access to such sensitive information. Fig. 11 depicts the architecture of the partial scan proposed in [71], which consists of four major components: (1) The partial scan chain contains SFFs with no sensitive data, (2) the SFFs removed from the scan chain with sensitive data, which are connected to the DUT but not to the SIs/SOs, (3) a FSM that controls the value of the unchained SFFs, which provides the tester full controllability to the unchained SFFs, and (4) a LFSR that stores a backup copy of the removed SFFs to ensure the observability of them. Meanwhile, the output of the LFSR is XORed with the partial scan chain output. Without knowing the state of the FSM and the config of the LFSR, the adversary cannot control and observe the value in the unchained flip-flops.

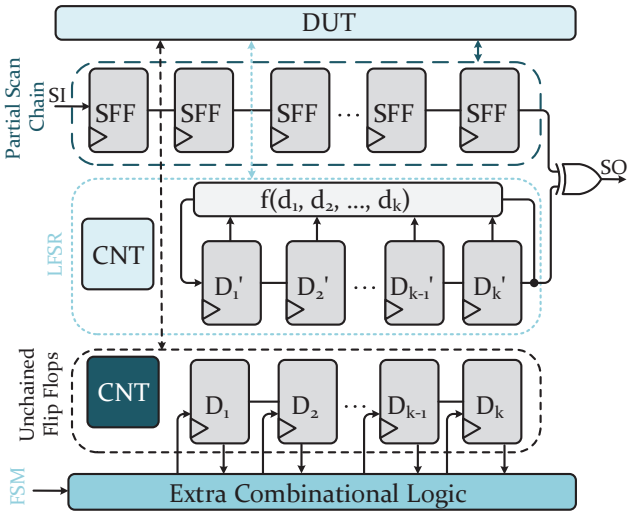


FIGURE 11: Partial Secure Scan [71].

IV. MORE ADVANCED SECURE SCAN CHAIN ARCHITECTURES IN THE PRESENCE OF LOGIC OBFUSCATION

Many of the secure scan chain architectures, particularly those were introduced for crypto systems, which are elaborated in the section III, rely on an outdated and very limited threat model described in Section III-A. One big assumption of this threat model is that the scan chain structure is unknown to the adversary. However, many recent studies demonstrate and validate the possibility of successfully reverse-engineering an IC and a PCB via delayering, imaging, annotation, and netlist extraction [72], [73]. Thus, since the adversary has access to the successfully reverse-engineered netlist, many of the previously discussed secure scan architectures would be invalid, particularly those architectures that manipulated the scan chain structure without the insertion of the key, such as flipped scan, feedback XORed scan, and state dependent scan.

Similar to crypto systems that require a protection technique for the sensitive data during the test/debug phase, such as the encryption key, having a secure scan chain architecture in the presence of logic obfuscation is inevitable. Moreover, it gets worse in the logic obfuscation when one requirement of the SAT attack is having access to the scan chain [7]. Hence,

not only for protecting the obfuscation key but also to resist against the SAT attack, having a secure/restricted scan chain architecture in the presence of logic obfuscation techniques is mandatory. Since de-obfuscation using the SAT attack relies on the fact that the adversary has access to a successfully yet locked reverse-engineered netlist, hence, compared to the threat model for secure scan chain architecture in crypto systems, the threat model and assumptions in this category would be much stronger.

A. THREAT MODEL AND ASSUMPTION

The most prevalent threat model in the presence of the logic obfuscation could be enumerated as follows:

1) The designer is trusted, i.e., the personnel and the tools used in the design house are trustworthy.
2) The foundry and the end-user are untrusted.
3) The adversary has access to the successfully reversed-engineered but locked netlist.
4) The adversary is able to purchase the unlocked/licensed functional chip from the market, so (s)he could apply any desired inputs to an unlocked/licensed chip and monitor the correct outputs.
5) The adversary knows the logic obfuscation technique, as well as the location of the key gates. The obfuscation key is the only unknown value to the adversary.

B. SCAN CHAIN BLOCKAGE VS. SCAN CHAIN OBFUSCATION

Securing the scan chain architecture in the presence of the logic obfuscation could be categorized into two main groups: (1) Blocking the scan chain after activation process, and (2) independently obfuscating the scan chain. It is worth mentioning that in both cases, an obfuscation technique has already been applied to the circuit. Hence, as shown in Fig. 12 and Fig. 13, in both cases, the combinational logic is already obfuscated, and the obfuscation key (functional key) is initiated and stored into a tamper-proof non-volatile memory (tpNVM) after fabrication via a trusted party.

Similar to the scan-based attacks on crypto systems, by getting the benefit of the access to the internal states using

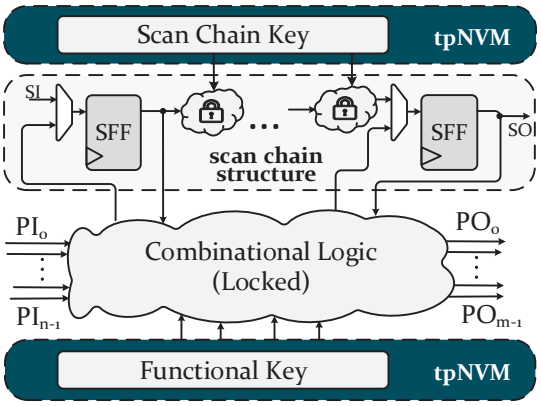


FIGURE 12: Scan Chain Obfuscation Mechanism.

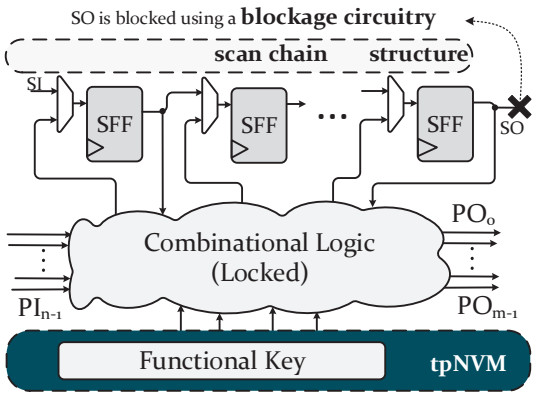


FIGURE 13: Scan Chain Blockage Mechanism.

the scan chain, the adversary is able to retrieve the functional key. The SAT attack is an obvious example of this point of vulnerability/threat. Hence, many recent studies on logic obfuscation started to explore the solution space, including scan chain obfuscation as well as scan chain blockage, for restricting the access to the scan chain while the logic obfuscation is in place. Fig. 12 demonstrates the architectural overview of the scan chain obfuscation. Similarly, Fig. 13 shows the architectural overview of the scan chain blockage. As shown in Fig. 12, the scan chain paths are locked in the scan chain obfuscation, and similar to functional key, the scan chain key must be initiated/stored in tpNVM. To lock the scan chain paths, different mechanisms have been studied in the literature. For instance, similar to scrambled secure scan architecture described in Section III-E, the order/sequence of the scan registers could be obfuscated, and the correct key value establishes the correct order/sequence. In the next section, we will review some of the mechanisms in this breed of obfuscation, such as encrypt-FF and DOS [32], [74].

In scan chain blocked architecture, on the other hand, as shown in Fig. 13, the SO is blocked using a blockage circuitry. By using blockage architecture, the access to the scan chain pins become limited after activation/unlocking of the obfuscated circuit. In the following, we comprehensively review and compare these two groups of solutions in terms of security, overhead, and test complexity.

V. OBFUSCATING THE SCAN CHAIN IN THE PRESENCE OF LOGIC OBFUSCATION

To obfuscate the scan chain, similar to the logic obfuscation flow, the desired key-based gates (key gates) must be added within the scan chain path. However, unlike logic obfuscation, which could be done either during the design step or post-synthesis step, scan chain obfuscation must be done after design-for-testability (DFT) synthesis, where the scan chain structure has been inserted into the design.

Considering that the scan chain is obfuscated, the adversary has no longer the capability of loading the initial state into SFFs without having the correct scan chain key. (S)he is also no longer able to monitor the updated value of the SFFs

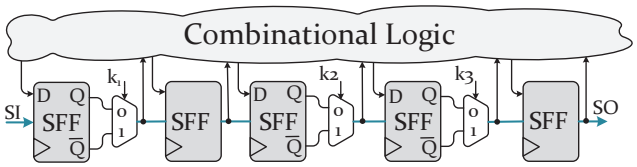


FIGURE 14: Encrypt Flip-Flop strategy [32].

after one-cycle capture mode. Hence, the SAT attack and all other attacks that require access to the scan chain would fail after applying the scan obfuscation on a circuit.

A. ENCRYPT FLIP-FLOP

The primitive solution for obfuscating the scan chain was introduced in [32], called *encrypt flip-flop* (EFF), which obfuscates the outputs of a list of selected SFFs. As shown in Fig. 14, the output of SFFs are obfuscated using MUX21 controlled by the key. The inputs of the MUX are connected to Q and $\bar{Q}$, and its output is connected to the next logic level. A wrong key-input propagates an inverted input to the next logic level, which leads to an erroneous functionality of the circuit. Also, a placement strategy has been introduced in EFF to be used instead of the random placement that may expose a design to the adversary.

Considering that the adversary (especially an end-user) can handle/manipulate the key, it could be set to the desired value that has no toggle. The key determines which connections have the inverters, and in this case, the overall architecture would be similar to the flipped scan in the cryptosystem. However, it is worth mentioning that handling/manipulating the key is relatively impractical in case of logic/scan obfuscation. The key values come from a tpNVM that would be initiated after fabrication via a trusted party, and after the reverse engineering, the content of these memories would be wiped out, and there is no way to reduce this countermeasure to a simpler approach like the flipped scan.

B. DYNAMICALLY OBFUSCATED SCAN

Unlike EFF that obfuscates the scan chain statically, a dynamically obfuscated scan (DOS) architecture is first proposed in [74]. Also, unlike EFF that engages MUX-based key gates, in DOS, the scan data is obfuscated using XOR gates, and one input of the XOR gates is controlled dynamically by an obfuscation key generated by a linear feedback shift register (LFSR). The overall structure of DOS is very similar to the PUF-based feedback XOR structure in crypto systems. However, unlike the PUF-based feedback XOR structure that engages the output of a counter as the input challenge to the PUF, in DOS, a control unit has been used to determine the update frequency of the obfuscation key that is generated by the dynamic module (the LFSR).

The dynamically obfuscated scan architecture is composed of a LFSR, a shadow chain with XOR gates, and a control unit as illustrated in Fig. 15. In this architecture, based on these three major components, the obfuscation flow of the DOS follows these steps:

1) During the initialization, a control vector is loaded to both the LFSR and the control unit (a seed for the LFSR and a vector to determine the obfuscation key update frequency).
2) The obfuscation key is generated at the output of the LFSR.
3) During the first λ scan clock cycles after the reset (λ is the length of scan chains), the protected obfuscation key is generated bit by bit based on the shadow chain and the obfuscation key. During this period, the scan chains are not obfuscated but their outputs are blocked with AND gates (masked to $ZERO$).
4) At the λ^{th} scan clock cycle, the protected obfuscation key settles down.

After these steps, all the test patterns and responses will be scrambled based on the protected obfuscation key. The DOS promises all the security guarantees (resilience to scan flush and mode-reset attempts, no key inference from XOR locations, etc.) of static scan obfuscation. In addition, resetting attacks will not work due to the utilization of the shadow scan chain which protects the obfuscation key during a global reset. The shadow scan chain also protects from differential attacks [75], as the first SO pattern is all 0's. Furthermore, as the key is updated periodically, there is a limited time window for any attack to retrieve a key before it is updated.

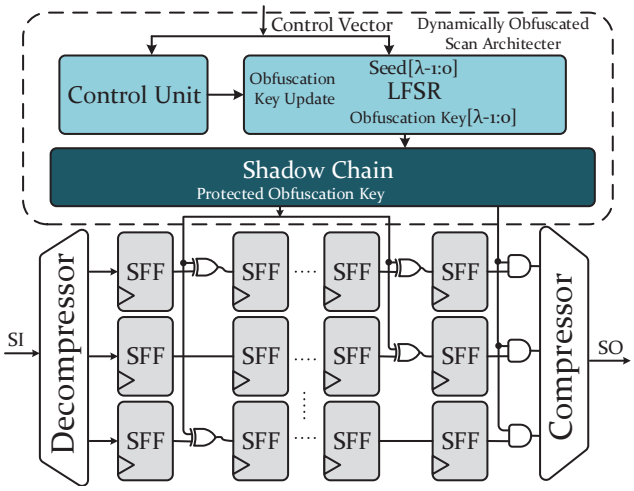


FIGURE 15: Dynamically Obfuscated Scan [74].

C. ATTACKS ON OBFUSCATED CIRCUITS WITH RESTRICTED SCAN

Shortly after the introduction of the first studies on a secure scan chain in the presence of the logic obfuscation, new studies demonstrate the feasibility of engaging the SAT attack on circuits even while the access to the scan chain is restricted. When the access to the scan chain is blocked/obfuscated, the adversary access would be limited to PI/PO. Hence, s(he) has to deal with a fully sequential circuit. Since the SAT attack is only applicable to the combinational circuits, the attacks in this category, which are referred to as *sequential SAT*

attacks, try to convert each circuit to its combinational counterpart, and then applies the SAT attack. This preprocessing step could be done using unfolding/unrolling with a specific depth. In fact, the adversary would unroll the obfuscated (sequential) circuit by up to u times. The u times unrolled circuit takes in u input patterns and producing u outputs. Also, there are formal verification methods, such as model checking techniques that would be able to formulate the miter circuit (for finding the distinguishing input) for the sequential circuit with a specific depth.

1) Primitive Sequential SAT Attack

The sequential SAT was first introduced in [17], which only requires access to the PI/PO. The overall procedure of this attack is shown in Algorithm 1. Similar to the SAT attack, it has an iterative process for pruning the search space. However, due to the restricted access to the internal registers, rather than finding a discriminating input pattern in each iteration, it finds a sequence of inputs X denoted as discriminating input *sequence* (X_{DIS}) that can generate two different outputs for two different keys.

In the sequential SAT attack described in Algorithm 1, $C(X, K, Y)$ refers to the obfuscated circuit producing output sequence Y using input sequence X and key vector K, and $C_{BlackBox}(X)$ refers to the output sequence of the activated/unlocked circuit for the same input sequence. After transforming the obfuscated circuit to a circuit SAT (Model) problem, the attack instantiates a bounded model checker (BMC) to find the X_{DIS}. After the discovery of each X_{DIS}, the model is updated with a new condition to make sure that the next pair of keys, that will be discovered in the subsequent attack iterations, produce the same output for previously discovered X_{DIS}. This process continues until no further X_{DIS} is found within the boundary of b. After reaching the boundary, the algorithm checks three criteria to determine if the attack can be terminated:

Algorithm 1 Sequential Attack on Obfuscated Circuits [17]

$b = initial\_boundary, Terminated = False$;
$Model = C(X, K_1, Y_1) \wedge C(X, K_2, Y_2) \wedge (Y_1 \neq Y_2)$;
while not $Terminated$ **do**
  while $(X_{DIS}, K_1, K_2) \leftarrow BMC(Model, b) = T$ **do**
    $Y_f \leftarrow C_{BlackBox}(X_{DI})$;
    $Model = \wedge\, C(X_{DIS}, K_1, Y_f) \wedge C(X_{DIS}, K_2, Y_f)$;
  if $\mathbf{U}C(Model, b) \vee \mathbf{C}E(Model, b) \vee \mathbf{U}MC(Model)$ **then**
    $Terminated$;
  $b = b + boundary\_step$;

Unique Completion (UC): This criterion checks for the uniqueness of the key found by the algorithm. If there is only a single key that satisfying all previous DISes, the attack is successfully terminated, and the key is the correct one.

Combinational Equivalence (CE): If there is more than one key that agrees with all previously found X_{DIS}, the attack checks the combinational equivalency of the remaining keys. In this step, the input/output of FFs are considered as pseudo primary outputs/inputs allowing the attacker to treat the circuit as combinational. The resulting circuit is subjected to a SAT attack, and if the SAT solver fails to find a different output or next state for two different keys, it concludes that all remaining keys are correct and the attack terminates successfully.

Unbounded Model Check (UMC): If both UC and CE fail, the attack checks the existence of a DIS for the remaining keys using an unbounded model checker. This is an exhaustive search with no limitation on bound (or the number of unrolls). If no DIS is discovered, the existing set of DIS is a complete set, and the attack terminates. Otherwise, the bound is increased and previous steps are repeated.

Although this primitive sequential SAT attack is able to break the obfuscated circuit locked by encrypt flip-flop (EFF), it is only applicable to static scan chain obfuscation. Hence, it is not applicable on LFSR-based DOS architecture that periodically updates the key using a LFSR. Also, this sequential SAT attack runs into the scalability issues as it relies on two sub-routines which are in **PSPACE** and **NP**, thereby, failing to terminate for even moderately small circuits, which contain only a few thousand gates.

2) KC2

The work in [18], improved and accelerated the primitive sequential SAT attack [17] via implementing several tweaks in the attack procedure. KC2 combines a traditional sequential SAT attack [17] with several dynamic optimization techniques to implement a faster deobfuscation method. This technique uses incremental SAT solving along with BDD/SAT-based key condition sweeping, conversion of key-conditions to BDDs, and negative key-condition crunching while avoiding unnecessary clause inflation to boost a speedup in decryption times of up to two orders of magnitude in comparison to sequential SAT attack [17].

Although KC2 demonstrates the run-time speedup by two orders of magnitude, further investigation on still small circuits shows that this technique also runs into the scalability issue, particularly while the number of DISes increases.

3) ScanSAT on Both Static and Dynamic Obfuscation

Similar to the primitive sequential SAT attack, ScanSAT [21] is based on the insight that the complex u-cycle transformation of scan obfuscation could be formulated by generating an unfolded/unrolled combinational (1-cycle transformation) equivalent counterpart of the scan-obfuscated circuit. The obfuscation modules on the scan path become part of the resultant combinational circuit, which effectively is an obfuscated circuit with key gates at the pseudo-primary inputs/outputs of the circuit. The obfuscated circuit equivalent of a generic scan-obfuscated circuit in Fig. 16(a) is provided in Fig. 16(b), where the obfuscation on the stimulus and the response are modeled separately as combinational blocks driven by the same scan obfuscation key. In general, ScanSAT models the obfuscated scan chains as a logic-locked combinational circuit, paving the way for the application of the powerful

SAT attack to reveal the key (sequence), unlocking the scan chains, and thus, restoring access to the oracle.

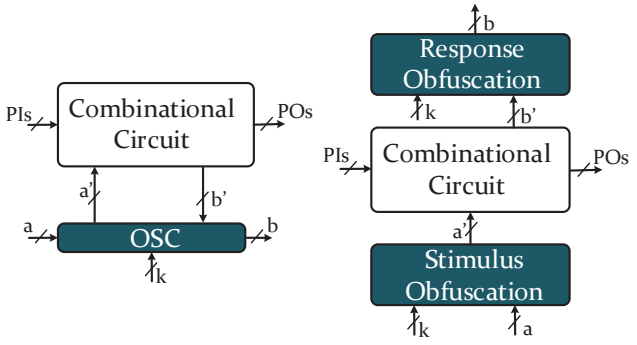
In addition to de-obfuscating the statically scan-obfuscated circuit, ScanSAT is also able to be applied on dynamically scan-obfuscated circuit that is locked by DOS architecture. Since after successfully reverse engineering, the LFSR structure, and its polynomial are known to the adversary, finding the seed and update frequency parameter (p), that is the only secret in DOS architecture, would lead to deriving all the keys that are dynamically generated on the chip.

A simple method to identify p is to apply the same stimulus pattern repeatedly from the SI, and observe the response through the SO. The point is that after p capture operations, by repeatedly applying the same stimulus, the response would be different because of the updated key; thus, most likely, there will be a noticeable change in the observed response, helping detect the update operation on the key.

After finding p, the same approach that were used for static scan obfuscation would be used in this case. The difference now is that the SAT attack could be executed for at most p iterations (after p iterations, the key is updated). If more than p DIPs are required to identify a dynamic key, the SAT attack needs to be terminated prematurely upon p DIPs. Another SAT attack must be executed subsequently to identify the next dynamic key in the sequence still within p iteration. Since the updated key is generated by the LFSR whose polynomial is known for the adversary, independent SAT attack runs on each dynamic key reveals partial information of the seed; thus, the information from independent SAT attack runs by gradually gathering information about the seed in every run, and finally by incorporating into the ScanSAT model, the relationship between the seed and the keys would be revealed.

D. YET ANOTHER DYNAMIC SCAN OBFUSCATION: DYNAMIC ENCRYPT FLIP-FLOP

As a countermeasure against ScanSAT attack, dynamic encrypt flip-flop (EFF-Dyn) [76] combines scan obfuscation approach from EFF [32] and a PRNG, to introduce dynamicity in the design. In EFF-Dyn, during either functional mode or the capture operation in test mode (scan enable (SE) signal is low), the scan obfuscation key that is stored in the tpNVM controls the key gates. During testing, an externally provided test key is expected. When this test key matches the scan obfuscation key, the key gates receive this correct key during the shift operations (SE signal is high) as well; and in case of a mismatch, however, the PRNG that updates the key in every clock cycle controls the key gates dynamically. The overall structure of EFF-Dyn has been illustrated in Fig. 17.



(a) Obfuscated Scan Chain **(b)** Combinational Equivalent Model

FIGURE 16: Converting an Obfuscated Scan Chain to its Combinational Counterpart [21].

Although EFF-Dyn introduces a new countermeasure against scanSAT as the state-of-the-art attack on logic obfuscation with restricted scan access, the biggest issue in this model of defense is that it is assumed the tester must be trusted party to have the actual key for testing. However, in many cases for many high-tech companies, this assumption is not valid. Hence, it could not be counted as a reliable solution for them.

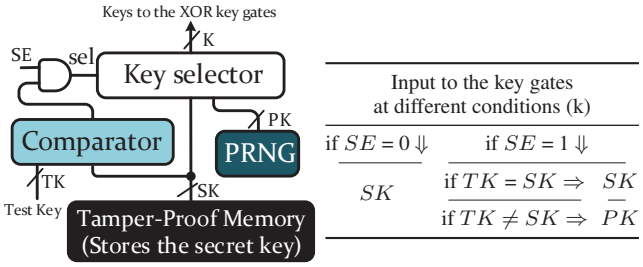


FIGURE 17: Dynamic Encrypt Flip-Flop (EFF-Dyn) [32].

E. DYNUNLOCK ATTACK: BREAKING DYNAMIC ENCRYPT FLIP-FLOP

Similar to LFSR, the structure of PRNG and its polynomial would be known for the adversary after successfully reverse engineering. So, if the adversary could recover the seed of the PRNG, the generated key sequence would be revealed. Hence, similar to the ScanSAT, a new attack called DynUnlock [77] introduces a similar approach to show how it is possible to find the seed of the PRNG in EFF-Dyn. With the secret seed known, the adversary can gain scan access without the knowledge of the scan obfuscation key; an arbitrary test key can be used to leave the scan access control to the PRNG, which can be easily modeled by the adversary as long as its seed is known.

With assuming that the structure of PRNG is similar to an LFSR in DynUnlock, as shown in Fig. 18, it first starts by reverse-engineering the LFSR circuit and obtaining the equations corresponding to each clock cycle. Next, it determines the location of key gates inserted between the SFFs. Then it models this sequential logic circuit into a combinational circuit with SFFs replaced with inputs and outputs. Once modeling is complete the combinational obfuscated counterpart circuit, with seed bits acting as primary key inputs, is fed to a SAT solver, which provides a DIP and its corresponding output pattern. In [77], the authors carry out the attack for just one capture cycle. To recover more bits, they restart the LFSR circuit and obtain a new DIP and its corresponding output

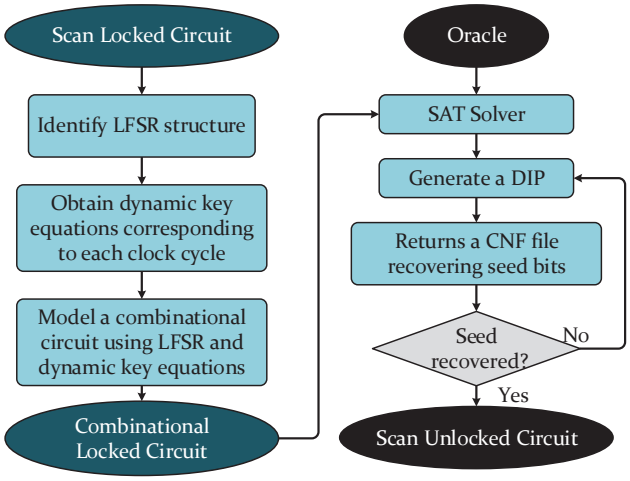


FIGURE 18: Flowchart for the DynUnlock Attack [77].

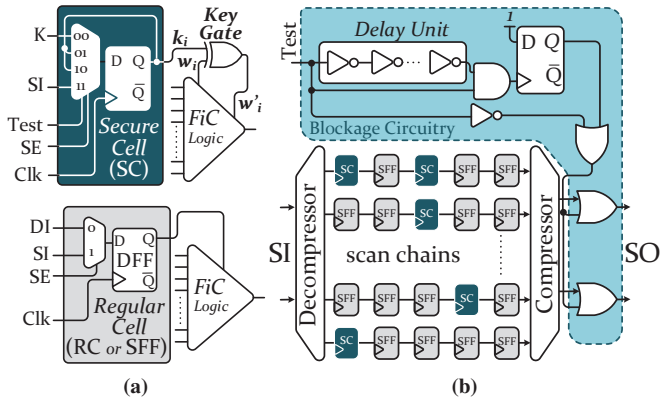


FIGURE 19: (a) Secure Cell (SC) vs. Regular Cell (SFF), (b) Restricted Unauthorized Scan Access using Blockage Circuitry [34].

pattern from the SAT solver, and recover more seed bits. they repeat the restart step until all the seed bits have been recovered, or the remaining seed bits can be brute-forced.

F. ENGAGING CRYPTOGRAPHIC MACROS IN SCAN OBFUSCATION

ScanSAT and DynUnlock demonstrate that adding dynamicity into the design by means of predictable modules, such as LFSR and PRNG, does not add any advantages to guarantee the security of the scan chain. Hence, a few recent studies rely on encrypting the test communication to ensures the confidentiality of the exchanged messages between the circuit and the tester. Unlike predictable modules, such as LFSR and PRNG, that could be analyzed and learned by the attack, encryption algorithms, and modules like PUF and TRNG, is not either predictable or breakable. By using the encryption, any unauthorized communication with the protected circuit would be failed, and the adversary is no longer able to intercept and handle the communication [78], [79]. Both block ciphers and stream ciphers have been engaged and evaluated for scan encryption, and it is shown that the preferred mechanism for encrypting the scan chain is the usage of stream ciphers [80], because block ciphers incur a larger area overhead and have to be adapted to cope with the serial nature of the exchanged data with the automatic test equipment (ATE). Nevertheless, more investigation on stream ciphers shows that this group of ciphers may introduce vulnerabilities, and for this reason, the implementation of dedicated countermeasure introduce a larger cost.

VI. BLOCKING THE SCAN CHAIN IN THE PRESENCE OF LOGIC OBFUSCATION

Recent studies have evaluated the possibility of blocking the scan chain after activation of the obfuscated circuit. They show how this mechanism could provide more benefits compared to both scan chain obfuscation and scan chain encryption techniques. Scan chain blockage techniques limit access to the scan chain. They also have no impact on the structural test and negligible impact on the functional test while their overhead is considerably lower than both scan chain obfuscation and scan chain encryption. In this Section, we evaluate the state-of-the-art scan blockage techniques when logic obfuscation is in place to show why moving towards scan blockage techniques are more promising when secret assets are stored in the chip.

A. R-DFS: ROBUST DESIGN FOR ASSURING THE TRUST IN LOGIC OBFUSCATION

Scan chain blocking in the presence of logic obfuscation was first introduced in [34], called **r**obust **d**esign-**f**or-**s**ecurity (R-DFS). R-DFS introduces and proposes a custom scan (storage) cell, which is denoted as secure cell (SC). The main aim of SCs is to store the obfuscation key securely. Fig. 19(a) shows the differences between a regular scan (storage) cell (RC or SFF) and the proposed SC. SCs are equipped with a 4-to-1 multiplexer ($MUX41$); thus each SC could operate in four different modes. As depicted in Table 2, SCs' operation mode depends on the SE pin and the new pin called $Test$. Each SC is added to store one bit of the obfuscation key. The key values could be loaded into SCs either directly from tp-NVM (actual key, $\{Test, SE\} = \{0, 0\}$, mode M_0) or the SI (dummy/actual key, $\{Test, SE\} = \{1, 1\}$, mode M_2). The overall structure of scan chains in R-DFS is shown in Fig. 19(b). The scan chains are constructed by stitching the SCs with regular SFFs (RCs). The SCs can keep their values in modes M_{1a} and M_{1b}. The only difference between the M_{1a} and M_{1b} mode is the value of the SE pin that determines the shift/capture mode in RCs/SFFs. Both of the M_{1a} and M_{1b} modes allow the SCs to be bypassed (preserving their values) when the SFFs are in shift/capture mode.

Stitching SCs and RCs allows accomplishing the structural (a.k.a. manufacturing fault) test with no issue. To do that in R-DFS, the $Test$ pin must be 1, allowing the shift and capture operations to be carried in modes M_2 and M_{1b} respectively, giving unrestricted access to the scan. Hence, the tester could

TABLE 2: Modes of Operation in Secure Cell (SC) [34].

Test	SE	Mode	Description
0	0	M_0	The circuit is in functional mode. Actual keys from tpNVM applies to the Logic (Correct Functionality).
0	1	M_{1a}	The SCs hold their previous value. Based on the value of SE, SFFs are in capture/shift mode.
1	0	M_{1b}	
1	1	M_2	The SCs become part of the scan chain. Actual/Dummy keys from SI for structural testing.

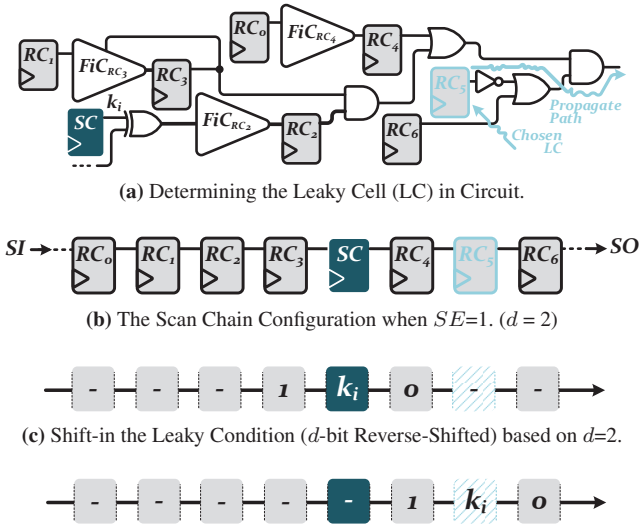
load any key (e.g. generated by ATPG tool), to initiate all storage units (SCs and RCs/SFFs) to the initial value, and carry out the structural test. For functional test on the other hand, the correct key (actual key) is loaded from tpNVM into SCs using the mode M_0 ($\{Test, SE\} = \{0, 0\}$). Then, the initial state is loaded into the SFF in mode M_{1a}, with no change on the key value in SCs. But, in R-DFS, after switching to mode M_0, since the actual key is loaded into the DUT, to avoid any form of leakage, or break scan-based attack, all scan out (SO) pins will be blocked. So, the tester has to observe the response through the PO in mode M_0.

To block SO pins after loading the actual key (after switching to mode M_0), a blockage circuitry has been introduced in R-DFS. Fig. 19(b) shows the blockage circuitry, which blocks/masks the SOs upon a switch from functional mode (mode M_0 that loads the actual key into SCs) to test mode (mode M_2 that supports the shift operation). Hence, after loading the actual key in mode M_0, SO will no longer be accessible. It will also remove the possibility of SAT attack, in which access to the SO is required. Additionally, it limits the adversary's attack option to the far weaker and non-scalable sequential attacks [17], [18].

B. SHIFT-AND-LEAK ATTACK ON R-DFS

Although SO pins will be blocked in R-DFS after switching to mode M_0, the availability of shift operation still reveals a leakage possibility in this countermeasure. The introduction of shift-and-leak attack [19] shows that there is a valid key leakage possibility in R-DFS that allows the adversary to observe and extract the logic obfuscation key using PO. This attack exploits (1) the availability of the shift operation through the scan, and (2) the capability of reading out the PO through chip pin-outs in the functional mode. Fig. 20 illustrates a simple example of how the new shift-and-leak attack could retrieve the key when R-DFS is in place. The steps of a shift-and-leak attack are as follows:

1) Identify leaky cells (LCs) that can leak info onto a PO (propagateable and sensitizable).
2) Insert a stuck-at-fault at the chosen LC candidate.
3) Propagate the fault onto a PO (SCs set to unknown $X's$). If it fails to propagate, it rules out this LC and repeats steps 1 and 2.
4) Power up the chip in mode M_0 to load the correct key into SCs.
5) Switch to mode M_{1a} (SCs hold value) and shift in d-bit reverse-shifted of the leak condition into the scan. The value of d is the scan distance between the targeted SC and the chosen LC.
6) Switch to mode M_2 (SCs are in the scan), and perform d-bit shift to have the leak condition in place and the key in chosen LC.
7) Clocklessly switch to mode M_0 and observe the PO, to leak the content of the LC, i.e., the target key bit.



(a) Determining the Leaky Cell (LC) in Circuit.

(b) The Scan Chain Configuration when SE=1. (d = 2)

(c) Shift-in the Leaky Condition (d-bit Reverse-Shifted) based on d=2.

(d) Shift all FFs, Including SCs and SFFs, in Mode M_2, to Put the k_i into the LC.

FIGURE 20: Example of Shift-and-Leak Attack on R-DFS [19].

To make the attack scalable, especially when the number of SCs increases, ATPG may fail to find a leak condition for the chosen LC. This challenge is also addressed in shift-and-leak attack by exploiting the conventional SAT attack [7]. So, a pre-processing step was added to the shift-and-leak attack, in which the logic cone was treated as a locked combinational circuit considering SFFs as the primary inputs and SCs as the key inputs. The pre-processing phase (which resembles the steps of the conventional SAT attack) is launched as follows:

1) Extract the combinational fan-in cones of the PO.
2) Obtain a Discriminating Input (DIP) from the SAT tool on the extracted circuit.
3) Power on the IC in mode M_0 (SCs capture the actual key).
4) Switch to M_{1a} (SCs hold their values), and shift in the obtained DIP from the SAT tool to the SFFs.
5) Clocklessly switch to mode M_0 and observe the PO ($eval$ of the SAT attack). Then, go to step 2 until no more DIP found.

By using this mechanism, the shift-and-leak attack is able to reveal the logic obfuscation key through the POs.

C. MR-DFS: MODIFIED ROBUST DESIGN FOR SECURITY TO RESIST AGAINST SHIFT-AND-LEAK

As a countermeasure, to defeat the shift-and-leak attack, and to re-assure the trust in logic obfuscation, the work in [19] proposes a modified version of robust design-for-security

architecture (denoted as mR-DFS in this paper) with a slight modification to the R-DFS. As discussed previously, the availability of shift operation still reveals a leakage possibility in the R-DFS architecture. The shift operation is provided using mode M_{1a}. Hemce, in mR-DFS, this mode is blocked. Also, to avoid any other form of leakage, after switching to mode M_0, it is not possible to re-enable any shift mode in the scan chain. To do that, as shown in Fig. 21, they build a shift disable (SD) signal, such that when $Test$ = 1, SD follows SE. But, after the first capture of the actual key, i.e. when there is a positive transition on the $Test$ or when the $Test$ is low, SD becomes *ALWAYS ZERO*. It helps to block the shift operation after that. So, there is no longer a mode where SCs can be bypassed, retaining their values, while SFFs can be loaded/shifted.

D. MR-DFS ARCHITECTURAL DRAWBACKS

The mR-DFS can address the leakage possibility in R-DFS using shift disable (SD) signal. However, the work in [20] illustrates that the proposed shift disable (SD) signal in mR-DFS poses some new challenges for design and implementation flow, as well as test and debug process, including (1) high functional test time, (2) necessity of duplicating SCs, and (3) the possibility of re-enabling shift using leaky glitches. For instance, regarding the high functional test time, since shift operation is no longer available after activation, to do functional test on a DUT with mR-DFS architecture, each test pattern requires a separate key load before observing the POs. This makes the functional test time significantly higher than usual. Also, with re-enabling the shift operation using the glitches, the work in [20] shows that the leakage possibility in mR-DFS is not resolved yet.

E. KT-DFS: KEY-TRAPPED DESIGN FOR SECURITY

Due to the architectural drawbacks in mR-DFS, the work in [20] introduces a new scan blockage mechanism which is called key-trapped DFS (kt-DFS). In kt-DFS as illustrated in Fig. 22(a), the scan chain(s) of the SCs are completely decoupled from the scan chain(s) of the SFFs/RCs. Also, the SCs' chain outputs are permanently blocked to avoid any form of the leakage.

To guarantee the security of SCs against any form of leakage, a new secure cell has been re-designed and introduced in kt-DFS, called 1-way secure cell (1wSC). Fig. 22(b) depicts the details of 1wSC. Each 1wSC has two internal storage elements: a scan-connected storage (denoted as FF_1), and a trap storage (denoted as FF_2). The scan-connected storage could be used to shift values in and out of the 1wSC or into the trap storage. However, the value of the trap storage cannot be shifted out, and is only connected to its corresponded key gate. The transfer of key value from FF_1 to FF_2 takes place after setting $REG = 1$ and $SE = 0$, which is called *register mode*. Registration of the key into trap storage takes place on the rising edge of the clock input of the FF_2, which is a function of REG and SE. Also, this condition is used as the $RESET$ condition of all FF_1s to clear their values.

In kt-DFS, the keys could be loaded into 1wSC from either tpNVM or scan-in (SI). Hence, the tester would be able to carry out the structural test by loading the desired key using SI. But, since the scan chain(s) of 1wSCs are decoupled in kt-DFS, two extra dedicated scan-enable and scan-in pins are used for the scan chain(s) of the SCs, called KSE and KSI respectively.

The behavior of 1wSC is controlled using two pins, called REG and SE. As captured in Table 3, based on these two pins, a 1wSC can be operated in three main modes. Similar to R-DFS and mR-DFS, a blockage circuitry is required to block the SO after the first attempt of key loading from the tpNVM. To support the operational modes in the kt-DFS, a new blockage circuitry is designed. In kt-DFS, the SO must be blocked after loading the actual keys into FF_2s. When KSE is low, the FF_1 is fed using tpNVM. Hence, $\overline{KSE}$ is used to mask the SO. Note that the actual key would be loaded into FF_2 when REG = 1 and SE = 0 (register mode). However, before this condition, the tester has to load the actual key into FF_1s while the KSE is low. Hence, by only considering KSE = 0 as the blocking condition, the register-mode is also covered. Accordingly, the SO would be no longer available when KSE becomes low. Although using these extra pins allows the designers to avoid any forms of leakage in kt-DFS architecture, engaging these extra pins will incur a significant impact on the PnR stage and die size of the chip.

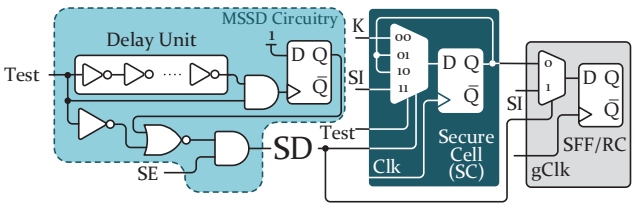
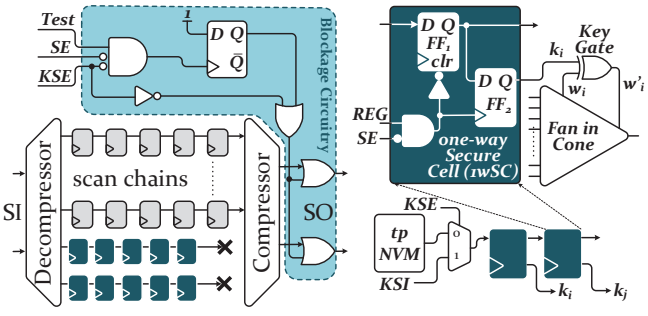


FIGURE 21: Mode Switch Shift Disable (MSSD) in mR-DFS [19].



(a) Decoupling SCs from SFFs **(b)** One-way Secure Cell (1wSC)

FIGURE 22: (a) kt-DFS Architecture with New Re-designed Blockage Circuitry, (b) Using 1wSC for Logic obfuscation Key. KSE determines the source of the key (tpNVM or KSI) [20].

TABLE 3: Modes of Operation in kt-DFS [20]

REG	SE	Mode	Description
0	0	M_0 (Functional Mode)	FF_2 must have the key*. FF_1 could capture the key*.
0	1	M_1 (Shift Mode)	FF_1 could capture the key*.
1	0	M_2 (Register Mode)	FF_2 are fed from FF_1. FF_1 will be reset to *ZERO*. Chain is erased.
1	1	M_3 (Shift Mode)	FF_1 could capture the key*.

* Based on KSE, actual/dummy key could be loaded from tpNVM/KSI

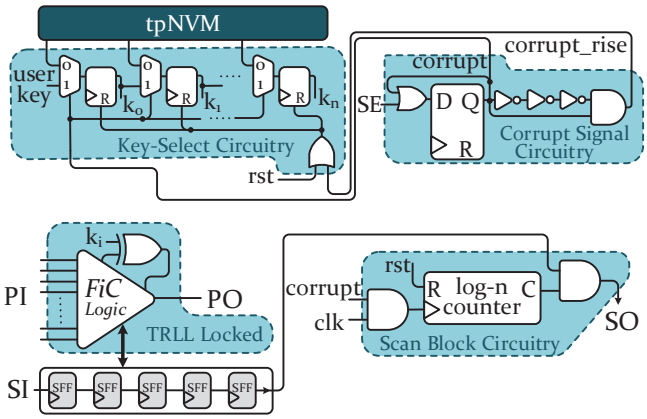


FIGURE 23: The Dishonest Oracle with Scan Blockage Circuitry [36]

F. DISORC: ORACLE DISHONESTY WITH SCAN BLOCKAGE: KEY-TRAPPED DESIGN FOR SECURITY

In the most recent study, DisORC [36], the scan chain blockage is combined with a new concept called dishonesty of the oracle. The overall structure of DisORC is illustrated in Fig. 23. Similar to kt-DFS, DisORC follows decoupling of SCs' chains and that of RCs/SFFs. The main aim of DisORC is to block the SO pins immediately after the first positive transition in *SE* pin. It is accomplished by using *corrupt* signal generated by *corrupt signal circuitry*. When *corrupt* is 1, *key-select circuitry* loads the user key (from the JTAG), as the new key, into the key registers, guaranteeing that there is no actual key in the DUT when the shift operation is enabled. So, the new key can corrupt the functionality (the oracle). Also, a counter-based scan block circuitry is designed, to guarantee that there is no leakage after switching to shift mode. DisORC can accomplish the in-field functional test using PI/PO. However, as discussed in Section II, carrying out the functional test through PI/PO might significantly affect test time and complexity. So, to have a high performance and efficient functional testing, a trusted party must have the correct key to perform the functional test.

G. COMPARISON OF SCAN BLOCKAGE TECHNIQUES

Table 4 provides a top view comparison between different scan blockage techniques in the presence of logic obfuscation. The overall structure of these techniques is almost the same, which relying on new secure cell and new blockage circuitry. However, DisORC opens a new direction with the introduction of oracle dishonesty, which makes the adversary capabilities more limited compared to other blockage techniques.

We compare the overhead in terms of logic and physical overhead. Regarding the logic overhead, logic count-based overhead has been provided in Table 4. Based on the overall architecture of these techniques demonstrated in Figs. 19, 21, 22, and 23, it seems that the 1wSC in the kt-DFS has two storage units and has a larger footprint compared to the other techniques. However, all other techniques also require two FFs per each key bit. For instance, in DisORC as demonstrated in Fig. 23, key registers in key-select circuitry are directly connected to tpNVM (through MUXes). But, connecting all key registers directly to tpNVM requires an ultra-wide memory that provides all bits (key bits) at once (one clock cycle) using only a single-clock read operation. This assumption is almost impossible particularly while the size of the key is in order of hundreds to thousands of key bits. So, engaging a wrapper built using temporary registers (FFs) to load (shift in) the key into them (from tpNVM) at power ON, then connecting each key register in key-select circuitry to its corresponding temporary register (FFs), is required. These temporary registers are scan-connected storage (denoted as FF_1) in 1wSC. Hence, the number/structure of key storage in all techniques is also similar.

Based on the gate count demonstrated in Table 4, kt-DFS requires less number of gates than other techniques. However, as listed in physical overhead, kt-DFS requires more extra pins, which significantly enhances the complexity of PnR, and it also affects the die size. Also, R-DFS and mR-DFS stitch all register in the same scan chains. However, kt-DFS and DisORC decouple scan chains of the key register that might have an impact on the PnR and optimization stages.

VII. DISCUSSION AND OPPORTUNITIES

Although the scan chain architecture is an integral part of almost all ICs that would be used for test/debug purposes, the state-of-the-art attacks on crypto systems and particularly obfuscated circuits, such as scan-based side-channel attacks, the SAT attack, and leaking-based attacks, show that having a secure scan chain architecture is indispensable while the sensitive data is stored in the chip. As of now, as discussed previously, three main categories, i.e. scan chain obfuscation, scan chain encryption, and scan chain blockage, are available to be engaged to resist against the existing threats. Table 5 compares the most well-known solutions of each category in terms of security (resilience against the existing attacks), the overhead and design/implementation complexity, and testability overhead/complexity.

As shown in Table 5, many of the existing approaches suffer from a big shortcoming. All manipulation mechanisms have been implemented based on the fact that reverse-engineered netlist could not be extracted by the adversary; however, recent achievements on successfully reverse engineering invalidates this assumption, and make them al-

TABLE 4: Major Differences between Different Scan Blockage Techniques in terms of Overhead / Test / Security.

Circuit	R-DFS [34]	mR-DFS [19]	kt-DFS [20]	DisORC [36]
Mechanism	New Secure Cell + New Blockage Circuitry + Stitching Secure Cells with SFFs	New Secure Cell + Blockage Circuitry + Disabling Shift + Stitching Secure Cells with SFFs	New Secure Cell + New Blockage Circuitry + Decoupling Secure Cells from SFFs + Key from Extra pin	New Blockage Circuitry + Oracle Dishonesty + Key Selection + Disabling Shift + JTAG Key
Suggested Obfuscation	SLL [4]	RLL [3]	SLL [4]	*new* TRLL [36]
Logic (Gate-Level) Overhead	$(2K+1)\times$FFs $(K)\times$MUX41 $(4)\times$INV1 $(M+1)\times$OR2 $(1)\times$AND2	$(2K+4)\times$FFs $(K)\times$MUX41 $(11)\times$INV1 $(M+2)\times$OR2 & $(2)\times$NOR2 $(4)\times$AND2	$(2K+1)\times$FFs $(1)\times$MUX21 $(3)\times$INV1 $(M+1)\times$OR2 $(1)\times$AND2	$(2K+log(M)+1)\times$FFs $(K)\times$MUX21 $(4)\times$INV1 $(M+2)\times$OR2 $(log(M)+4)\times$AND2 & $(1)\times$NAND2
Physical Overhead	*ONE* extra pin Efficient PnR^{+}	*ONE* extra pin Efficient PnR^{+}	*THREE* extra pin Non-Efficient PnR^{o}	*ONE* extra pin (JTAG key) Non-Efficient PnR^{o}
Manufacturing Test	Supported ✓	Supported ✓	Supported ✓	Supported ✓
Functional Test	Through POs ✓	High-Performance at Trusted *	Through POs ✓	High-Performance at Trusted *
Attacked by	Shift and Leak [19]	Glitch-based Shift and Leak [20]	None	None

K: Size of the Key Input M: Number of Scan Chains $^{+}$: Key scan chain is not decoupled ^{o}: Key scan chain is decoupled
*: Scan chain access is required to load the initial state of different macros (IP) (Section II)

TABLE 5: Comparison of Secure Scan Chain Architectures in Terms of Security, Testability Time/Complexity, and Overhead.

	Defense	Attacked by	Logic Overhead	Test/Implementation Issues	
				Limitation	Time
Manipulation	Flipped [58]	Reseting/Flushing SFFs [59]	Very Low	Vulnerable to RE	Low
	XOR-based [59]	XOR Positioning [60]	Very Low	Vulnerable to RE	Low
	double-XOR [60]	XOR Positioning [61]	Very Low	Vulnerable to RE	Low
	rXOR [61]	No Guarenteed Security of the PUF	High	Vulnerable to RE	Low
	SDSFF [62]	Sequential (Unrolling+SAT) and BMC-based [17]	Low	Vulnerable to RE	Low
Obfuscation	Scramble [63]	ScanSAT Attack [21]	Moderate	Trusted Tester w/ Correct Key^{+}	Low
	Test Key Integration [64], [66]	Sequential (Unrolling+SAT) and BMC-based [17]	Moderate	Trusted Tester w/ Correct Key^{+}	Low
	Mirror key [55]	Partial Protection on Scan Chain	Moderate	Trusted Tester w/ Correct Key^{+} Low Coverage	Low
	Sub-chain [67], [68]	Differential Attack [81]	Moderate	Trusted Tester w/ Correct Key^{+}	Low
	Partial Scan [70], [71]	Partial Protection on Scan Chain	High	Trusted Tester w/ Correct Key^{+} Low Coverage	Low
	Encrypt-FF [32]	Sequential SAT Attack, ScanSAT Attack [21]	Low	NO limit	Low
	Dynamically Obfuscated Scan [74]	ScanSAT Attack [21]	High	NO limit	Low
	Dynamic-EFF [35]	DynUnlock Attack [77]	Moderate	Trusted Tester w/ Correct Key^{+}	Low
Encryption	SEBC [79]	NONE	Very High	NO limit	High
	SESC [78]	NONE	High	NO limit	Very High
Blockage	Robust-DFS [34]	Shift & Leak Attack [19]	Moderate	ONE extra Test Pin	Low
	modifed Robust-DFS [19]	Glitch & Leak Attack [20]	Moderate	Key Re-Init + ONE extra Test Pin	Very High
	key-trapped-DFS [20]	NONE	Moderate	THREE extra Test Pins	Low
	DisORC [36]	NONE	Moderate	Trusted Tester* + ONE extra Test Pin (JTAG)	Low

Trusted Tester w/ Correct Key^{+}: the test facility must be trusted. To accomplish the test phase, the test facility requires to know the correct key value.
* For better efficacy in terms of functional test time/complexity, the scan access is required (to load initial (internal) state of different macros (IPs)).

ready broken after reverse engineering. Hence, all of them are not resilient against those attacks that assumed that the adversary has access to the successfully reverse-engineered netlist. Amongst a wide variety of logic obfuscation solutions, dynamic-based obfuscation, i.e. DOS [74] and DynEFF [35], are in the better condition. However, recent studies on the possibility of recovering PRNG/LFSR structure describe in both scanSAT [21] and DynUnlock [77] reveals the vulnerability of these solutions.

As shown in Table 5, although encryption-based scan chain architecture provides guaranteed security against the state-of-the-art attacks, this breed of architectures suffer from ultra-high overhead incurred by the encryption/decryption module. Also, since for test/debug purposes, all communications to/from the chip must be secured, the test time would be increased considerably. As of today, the existing blockage mechanisms provide more advantages compared to other categories, in terms of overhead and robustness. Also, the structural (manufacturing fault) test could be handled with full scan access, and only the functional test must be

done through PO after activation. Hence, the test limitation is minimized in this group of solutions. Also, fully blocking the scan chain after activation eliminates the possibility of any form of attack on this group of solutions which makes them more reliable compared to other categories.

The introduction of oracle dishonesty in DisORC [36] has opened a new direction in secure scan chain architectures. DisORC shows how an oracle could be turned into a dishonest reference whenever a potential attack is detected. Turning the oracle into a dishonest one with an advanced architecture like DisORC could be an open direction for further studies, which provides enhanced security at lower overhead and less test compromising.

Having direct access and reading the electrical signals on a chip is a big challenge against any security countermeasures, which recently received a lot of attention [82]. One approach to combat these threats is the exploit of randomness, such as randomizing initialization and using register with random initial values within the test/scan components. Hence, the design and implementation of test/scan components that support such randomization to combat physical accesses require more evaluation. Similarly, hardware Trojan insertion could undermine logic obfuscation techniques. Few recent studies have investigated hardware Trojan attacks on logic obfuscation [83]. Designing a new test/scan structure that could detect unauthorized test access (such as test access by activated hardware Trojan) is mandatory in this case. This is an open research area that requires more attention to combat this breed of threats.

Limiting access to the scan chain results in emerging more advanced attacks on logic locking that only require access to PI/PO, such as unrolling-based SAT and BMC-based sequential de-obfuscation [17], [18]. However, these attacks rely on the fact that the system clock is synchronized, and all registers will be updated at the same time. So, few recent studies investigate the possibility of invalidating this synchronicity using latch-based logic obfuscation or asynchronous circuits [84], [85]. However, these clock-gating and asynchronous architectures for obfuscation purposes have opened a new direction that still requires more investigation on their capabilities and limitations, particularly testability and their robustness against different threat models.

VIII. CONCLUSION

Although the access to the scan chain of ICs is mandatory for testability/debugging, it raises big questions about the security of the chips, particularly when there exist security information in it, such as the symmetric/asymmetric key of cryptographic algorithms and logic obfuscation key. In this paper, we reviewed all solutions and countermeasures introduced to build a secure scan chain architecture. We first demonstrated that some preliminary mechanisms have been introduced to secure the scan chain when the cryptographic modules are in place. Then, we showed that many of these approaches relied on a very limited threat model. We then described and summarized solutions and countermeasures introduced when logic obfuscation is in place and the adversary threat model is much stronger. In general, all solutions could be categorized as a manipulation, obfuscation, encryption, or blockage technique targeting the scan chain to make it invulnerable against the state-of-the-art attacks. We evaluated all secure scan chain architectures in terms of security and the resiliency, testability/debugging time and complexity, and area/power/delay overhead.

REFERENCES

[1] A. Yeh, "Trends in the Global IC Design Service Market," *DIGITIMES research*, 2012.

[2] M. Rostami, F. Koushanfar, and R. Karri, "A Primer on Hardware Security: Models, Methods, and Metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, 2014.

[3] J. A. Roy *et al.*, "EPIC: Ending Piracy of Integrated Circuits," in *Design, Automation and Test in Europe (DATE)*, 2008, pp. 1069–1074.

[4] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Design Automation Conference (DAC)*, 2012, pp. 83–89.

[5] A. Baumgarten, A. Tyagi, J. Zambreno, "Preventing IC piracy using reconfigurable logic barriers," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 66–75, 2010.

[6] J. Rajendran *et al.*, "Fault Analysis-based Logic Encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2013.

[7] P. Subramanyan, S. Sayak, and S. Malik, "Evaluating the Security of Logic Encryption Algorithms," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2015, pp. 137–143.

[8] M. El Massad *et al.*, "Integrated Circuit (IC) Decamouflaging: Reverse Engineering Camouflaged ICs within Minutes," in *Network and Distributed System Security Symposium (NDSS)*, 2015, pp. 1–14.

[9] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Removal Attacks on Logic Locking and Camouflaging Techniques," *IEEE Transactions on Emerging Topics in Computing*, 2017.

[10] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: Approximately Deobfuscating Integrated Circuits," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2017, pp. 95–100.

[11] D. Sirone and P. Subramanayan, "Functional Analysis Attacks on Logic Locking," *IEEE Transactions on Information Forensics and Security*, 2020.

[12] H. Zhou, R. Jiang, and S. Kong, "CycSAT: SAT-based Attack on Cyclic Logic Encryptions," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 49–56.

[13] K. Shamsi, D. Z. Pan, and Y. Jin, "IcySAT: Improved SAT-based Attacks on Cyclic Locked Circuits," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–7.

[14] K. Z. Azar, H. M. Kamali, H. Homayoun, and A. Sasan, "SMT Attack: Next Generation Attack on Obfuscated Circuits with Capabilities and Performance Beyond the SAT Attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, pp. 97–122, 2019.

[15] H. M. Kamali, K. Z. Azar, H. Homayoun, A. Sasan, "InterLock: An Intercorrelated Logic and Routing Locking," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020, pp. 1–9.

[16] K. Z. Azar *et al.*, "NNgSAT: Neural Network guided SAT Attackon Logic Locked Complex Structures," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020, pp. 1–9.

[17] M. El Massad, S. Garg, and M. Tripunitara, "Reverse Engineering Camouflaged Sequential Circuits without Scan Access," in *IEEE/ACM Int'l Conf. on Computer-Aided Design*, 2017, pp. 33–40.

[18] K. Shamsi M. Li, D. Z. Pan, and Y. Jin, "KC2: Key-Condition Crunching for Fast Sequential Circuit Deobfuscation," pp. 1–6, 2019.

[19] N. Limaye, A. Sengupta, M. Nabeel, and O. Sinanoglu, "Is Robust Design-for-Security Robust Enough? Attack on Locked Circuits with Restricted Scan Chain Access," *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, 2019.

[20] H. M. Kamali, K. Z. Azar, H. Homayoun, A. Sasan, "On Designing Secure and Robust Scan Chain for Protecting Obfuscated Logic," in *Great Lakes Symposium on VLSI (GLSVLSI)*, 2020, pp. 1–6.

[21] L. Alrahis *et al.*, "ScanSAT: Unlocking Obfuscated Scan Chains," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2019, pp. 352–357.

[22] N. Limaye and O. Sinanoglu, "DynUnlock: Unlocking Scan Chains Obfuscated using Dynamic Keys," in *Design, Automation and Test in Europe (DATE) Conference*, 2020, pp. 1–6.

[23] M. Yasin, B. Mazumdar, J. Rajendran, and O. Sinanoglu, "SARLock: SAT Attack Resistant Logic Locking," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 236–241.

[24] Y. Xie and A. Srivastava, "Mitigating SAT Attack on Logic Locking," in *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2016, pp. 127–146.

[25] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. Rajendran, and O. Sinanoglu, "Provably-Secure Logic Locking: From Theory to Practice," in *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1601–1618.

[26] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, Y. Jin, "Cyclic obfuscation for creating sat-unresolvable circuits," in *Proceedings of the on Great Lakes Symposium on VLSI (GLSVLSI)*, 2017, pp. 173–178.

[27] S. Roshanisefat, H. M. Kamali, A. Sasan, "SRCLock: SAT-resistant cyclic logic locking for protecting the hardware," in *Proceedings of the on Great Lakes Symposium on VLSI (GLSVLSI)*, 2018, pp. 153–158.

[28] A. Rezaei, Y. Shen, S. Kong, J. Gu, and H. Zhou, "Cyclic Locking and Memristor-based Obfuscation against CycSAT and inside Foundry Attacks," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, 2018, pp. 85–90.

[29] Y. Xie, and A. Srivastava, "Delay Locking: Security Enhancement of Logic Locking against IC Counterfeiting and Overproduction," in *Proceedings of Design Automation Conference (DAC)*, 2017, p. 9.

[30] K. Shamsi, M. Li, D. Z. Pan, Y. Jin, "Cross-lock: Dense layout-level interconnect locking using cross-bar architectures," in *Proceedings of the on Great Lakes Symposium on VLSI (GLSVLSI)*, 2018, pp. 147–152.

[31] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, "Full-Lock: Hard Distributions of SAT Instances for Obfuscating Circuits using Fully Configurable Logic and Routing Blocks," in *Proceedings of Design Automation Conference (DAC)*, 2019, p. 89.

[32] R. Karmakar, S. Chatopadhyay, R. Kapur, "Encrypt flip-flop: A novel logic encryption technique for sequential circuits," *arXiv preprint arXiv:1801.04961*, 2018.

[33] U. Guin, Q. Shi, D. Forte, and M. Tehranipoor, "FORTIS: A Comprehensive Solution for Establishing Forward Trust for Protecting IPs and ICs," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 21, no. 4, pp. 1–20, 2016.

[34] U. Guin *et al.*, "Robust Design-for-Security Architecture for Enabling trust in IC Manufacturing and Test," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 5, pp. 818–830, 2018.

[35] R. Karmakar, H. Kumar, and S. Chattopadhyay, "Efficient Key-gate Placement And Dynamic Scan Obfuscation Towards Robust Logic Encryption," *IEEE Transactions on Emerging Topics in Computing*, 2019.

[36] N. Limaye, E. Kalligeros, N. Karousos, I. G. Karybali, and O. Sinanoglu, "Thwarting All Logic Locking Attacks: Dishonest Oracle with Truly Random Logic Locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

[37] A. Rezaei *et al.*, "CycSAT-unresolvable cyclic logic encryption using unreachable states," in *Proceedings of Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2019, pp. 358–363.

[38] S. Roshanisefat, H. M. Kamali, H. Homayoun, and A. Sasan, "SAT-Hard Cyclic Logic Obfuscation for Protecting the IP in the Manufacturing Supply Chain," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 4, pp. 954–967, 2020.

[39] H. M. Kamali, K. Z. Azar, K. Gaj, H. Homayoun, and A. Sasan, "LUT-lock: A Novel LUT-based Logic Obfuscation for FPGA-bitstream and ASIC-hardware Protection," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2018, pp. 405–410.

[40] G. Kolhe *et al.*, "Security and Complexity Analysis of LUT-based Obfuscation: From Blueprint to Reality," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–8.

[41] H. M. Kamali, K. Z. Azar, H.Homayoun, and A. Sasan, "ChaoLock: Yet Another SAT-hard Logic Locking using Chaos Computing," in *International Symposium on Quality Electronic Design (ISQED)*, 2019, pp. 1–8.

[42] M. Yasin, B. Mazumdar, O. Sinanoglu, J. Rajendran, "Security Analysis of Anti-SAT," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2017, pp. 342–347.

[43] X. Xu, B. Shakya, M. Tehranipoor, D. Forte, "Novel Bypass Attack and BDD-based Tradeoff Analysis against all Known Logic Locking Attacks," in *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2017, pp. 189–210.

[44] Y. Shen and H. Zhou, "Double-Dip: Re-evaluating Security of Logic Encryption Algorithms," in *Proceedings of the on Great Lakes Symposium on VLSI (GLSVLSI)*, 2017, pp. 179–184.

[45] K. Zamiri Azar, H. Mardani Kamali, H. Homayoun, and A. Sasan, "Threats on logic locking: A decade later," in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*. ACM, 2019, pp. 471–476.

[46] A. Chakraborty, Y. Liu, and A. Srivastava, "TimingSAT: Timing Profile Embedded SAT Attack," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–6.

[47] Y. Shen *et al.*, "BeSAT: behavioral SAT-based attack on cyclic logic encryption," in *Proceedings of Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2019, pp. 657–662.

[48] J. Sweeney, M. Heule, and L. Pileggi, "Modeling Techniques for Logic Locking," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–9.

[49] X. Wang *et al.*, "Secure scan and test using obfuscation throughout supply chain," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 9, pp. 1867–1880, 2017.

[50] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, "SCRAMBLE: The State, Connectivity and Routing Augmentation Model for Building Logic Encryption," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2020, pp. 153–159.

[51] S. Roshanisefat *et al.*, "DFSSD: Deep Faults and Shallow State Duality, A Provably Strong Obfuscation Solution for Circuits with Restricted Access to Scan Chain," in *IEEE VLSI Test Symposium (VTS)*, 2020, pp. 1–6.

[52] X. Li, W. Li, J. Ye, H. Li, and Y. Hu, "Scan Chain based Attacks and Countermeasures: A Survey," *IEEE Access*, vol. 7, pp. 85 055–85 065, 2019.

[53] R. Kuppuswamy, P. DesRosier, D. Feltham, R. Sheikh, and P. Thadikaran, "Full Hold-Scan Systems in Microprocessors: Cost/Benefit Analysis." *Intel Technology Journal*, vol. 8, no. 1, 2004.

[54] B. Yang, K. Wu, and R. Karri, "Scan based Side Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard," in *International Conference on Test*, 2004, pp. 339–344.

[55] B. Yang, K. Wu, and R. Karri, "Secure scan: A design-for-test Architecture for Crypto Chips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 10, pp. 2287–2293, 2006.

[56] R. Nara, N. Togawa, M. Yanagisawa, and T. Ohtsuki, "A Scan-based Attack based on Discriminators for AES Cryptosystems," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 92, no. 12, pp. 3229–3237, 2009.

[57] R. Nara, K. Satoh, M. Yanagisawa, T. Ohtsuki, and N. Togawa, "Scan-based Side-channel Attack against RSA Cryptosystems using Scan Signatures," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 93, no. 12, pp. 2481–2489, 2010.

[58] G. Sengar *et al.*, "Secured flipped scan-chain model for crypto-architecture," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 11, pp. 2080–2084, 2007.

[59] M. Agrawal, S. Karmakar, D. Saha, and D. Mukhopadhyay, "Scan based Side Channel Attacks on Stream Ciphers and their Counter-measures," in *International Conference on Cryptology in India*, 2008, pp. 226–238.

[60] S. Banik and A. Chowdhury, "Improved Scan-chain based Attacks and Related Countermeasures," in *International Conference on Cryptology in India*, 2013, pp. 78–97.

[61] S. Banik, A. Chattopadhyay, and A. Chowdhury, "Cryptanalysis of the Double-Feedback XOR-chain Scheme Proposed in Indocrypt 2013," in *International Conference on Cryptology in India*, 2014, pp. 179–196.

[62] Y. Atobe *et al.*, "Dynamically Changeable Secure Scan Architecture against Scan-based Side Channel Attack," in *International SoC Design Conference (ISOCC)*, 2012, pp. 155–158.

[63] D. Hely, M. Flottes, F. Bancel, B. Rouzeyre, N. Berard, and M. Renovell, "Scan Design and Secure Chip," in *International On-Line Testing Symposium (IOLTS)*, 2004, pp. 219–224.

[64] J. Lee, M. Tebranipoor, and J. Plusquellic, "A Low-cost Solution for Protecting IPs against Scan-based Side-channel Attacks," in *IEEE VLSI Test Symposium (VTS)*, 2006, pp. 6–pp.

[65] U. Chandran and D. Zhao, "SS-KTC: A High-Testability Low-Overhead Scan Architecture with Multi-level Security Integration," in *IEEE VLSI Test Symposium (VTS)*, 2009, pp. 321–326.

[66] M. A. Razzaq, V. Singh, and A. Singh, "SSTKR: Secure and Testable Scan Design through Test Key Randomization," in *Asian Test Symposium*, 2011, pp. 60–65.
[67] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing Designs against Scan-based Side-Channel Attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 4, pp. 325–336, 2007.
[68] Y. Atobe, Y. Shi, M. Yanagisawa, and N. Togawa, "Secure Scan Design with Dynamically Configurable Connection," in *IEEE Pacific Rim International Symposium on Dependable Computing*, 2013, pp. 256–262.
[69] A. Cui, Y. Luo, H. Li, and G. Qu, "Why Current Secure Scan Designs Fail and How to Fix Them?" *Integration*, vol. 56, pp. 105–114, 2017.
[70] M. Inoue, T. Yoneda, M. Hasegawa, and H. Fujiwara, "Partial Scan Approach for Secret Information Protection," in *IEEE European Test Symposium*, 2009, pp. 143–148.
[71] X. Chen, Z. Lu, G. Qu, and A. Cui, "Partial Scan Design against Scan-based Side Channel Attacks," in *IEEE International Conference On Trust, Security And Privacy In Computing And Communications (Trustcom)*, 2018, pp. 1484–1489.
[72] S. Keshavarz, C. Yu, S. Ghandali, X. Xu, D. Holcomb, "Survey on Applications of Formal Methods in Reverse Engineering and Intellectual Property Protection," *Journal of Hardware and Systems Security*, vol. 2, no. 3, pp. 214–224, 2018.
[73] U. Botero *et al.*, "Hardware Trust and Assurance through Reverse Engineering: A Survey and Outlook from Image Analysis and Machine Learning Perspectives," *arXiv preprint arXiv:2002.04210*, 2020.
[74] D. Zhang, M. He, X. Wang, M. Tehranipoor, "Dynamically Obfuscated Scan for Protecting IPs against Scan-based Attacks throughout Supply Chain," in *IEEE VLSI Test Symposium (VTS)*, 2017, pp. 1–6.
[75] J. Rolt *et al.*, "A novel Differential Scan Attack on Advanced DFT Structures," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 18, no. 4, pp. 1–22, 2013.
[76] R. Karmakar, S. Chattopadhyay, R. Kapur, "A scan obfuscation guided design-for-security approach for sequential circuits," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2019.
[77] N. Limaye and O. Sinanoglu, "DynUnlock: unlocking scan chains obfuscated using dynamic keys," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020, pp. 270–273.
[78] M. Da Silva *et al.*, "A New Secure Stream Cipher for Scan Chain Encryption," in *IEEE International Verification and Security Workshop (IVSW)*, 2018, pp. 68–73.
[79] M. Da Silva, M. Flottes, G. Di Natale, and B. Rouzeyre, "Preventing Scan Attacks on Secure Circuits through Scan Chain Encryption," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 3, pp. 538–550, 2018.
[80] K. Z. Azar, F. Farahmand, H. M. Kamali, S. Roshanisefat, H. Homayoun, W. Diehl, K. Gaj, and A. Sasan, "COMA: Communication and Obfuscation Management Architecture," in *International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2019, pp. 181–195.
[81] S. M. Saeed, S. S. Ali, O. Sinanoglu, and R. Karri, "Test-mode-only scan attack and countermeasure for contemporary scan architectures," in *International Test Conference*, 2014, pp. 1–8.
[82] M. T. Rahman, S. Tajik, M. S. Rahman, M. Tehranipoor, and N. Asadizanjani, "The key is Left under the Mat: On the Inappropriate Security Assumption of Logic Locking Schemes," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2020, pp. 262–272.
[83] A. Jain, Z. Zhou, and U. Guin, "TAAL: Tampering Attack on Any Key-based Logic Locked Circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 4, pp. 1–22, 2021.
[84] J. Sweeney, V. M. Zackriya, S. Pagliarini, and L. Pileggi, "Latch-Based Logic Locking," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2020, pp. 132–141.
[85] K. Z. Azar, H. M. Kamali, S. Roshanisefat, H. Homayoun, C. P. Sotiriou, and A. Sasan, "Data Flow Obfuscation: A New Paradigm for Obfuscating Circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, no. 01, pp. 1–14, 2021.

KIMIA ZAMIRI AZAR received her B.Sc. degree in computer engineering from Khajeh Nasir University in Tehran, Iran. She received her M.Sc. degree in computer engineering from the Shahid Beheshti University, Tehran, Iran in 2015. She is currently a Ph.D. candidate at the Electrical and Computer Engineering department at George Mason University. Her research interests are in the areas of hardware security and trust, security for supply chain, and VLSI design and test.

HADI MARDANI KAMALI received his B.Sc. degree in computer engineering from Khajeh Nasir University in Tehran, Iran in 2011, and his M.Sc. degree in computer engineering from Sharif University of Technology, Tehran, Iran in 2013. He is currently a Ph.D. student at the Electrical and Computer Engineering department at George Mason University. His research focuses on the security evaluation of logic obfuscation, design for trust, and supply chain security.

HOUMAN HOMAYOUN received the Ph.D. degree from the Department of Computer Science, University of California at Irvine, Irvine, CA, USA, in 2010. He is currently an Associate Professor at the Electrical and Computer Engineering Department, University of California Davis, CA, USA. Since 2017 he has been serving as an Associate Editor of IEEE Transactions on VLSI. He was the program co-chair of GLSVLSI'18 and the general chair of the GLSVLSI'19 conference.

AVESTA SASAN received his B.Sc. in Computer Engineering from the University of California Irvine in 2005 with the highest honor (Summa Cum Laude). He then received his M.Sc. and his Ph.D. in Electrical and Computer Engineering from the University of California Irvine in 2006 and 2010 respectively. Dr. Sasan then worked at the industry in Broadcom and Qualcomm Co. till 2016. He joined George Mason University in 2016, where he is currently serving as an Associate Professor in the Department of Electrical and Computer Engineering.