Security Analysis of Anti-SAT

Muhammad Yasin[†], Bodhisatwa Mazumdar[‡], Jeyavijayan (JV)[ξ] Rajendran and Ozgur Sinanoglu[‡]

yasin@nyu.edu, bm105@nyu.edu, jv.ee@utdallas.edu, ozgursin@nyu.edu

† Electrical and Computer Engineering, NYU Tandon School of Engineering, NY, USA

ξ Erik Jonsson School of Engineering & Computer Science, The University of Texas at Dallas, TX, USA

‡ Electrical and Computer Engineering, New York University Abu Dhabi, Abu Dhabi, U.A.E.

Abstract— Logic encryption protects integrated circuits (ICs) against intellectual property (IP) piracy and overbuilding attacks by encrypting the IC with a key. A Boolean satisfiability (SAT) based attack breaks all existing logic encryption technique within few hours. Recently, a defense mechanism known as Anti-SAT was presented that protects against SAT attack, by rendering the SAT-attack effort exponential in terms of the number of key gates. In this paper, we highlight the vulnerabilities of Anti-SAT and propose signal probability skew (SPS) attack against Anti-SAT block. SPS attack leverages the structural traces in Anti-SAT block to identify and isolate Anti-SAT block. The attack is 100% successful on all variants of Anti-SAT block. SPS attack is scalable to large circuits, as it breaks circuits with up to 22K gates within two minutes.

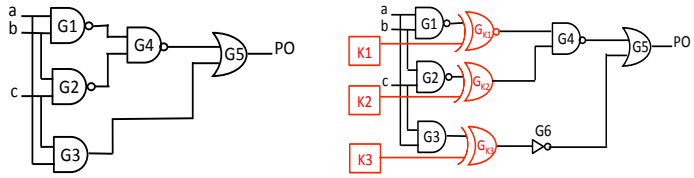
I. Introduction

In present-day semiconductor manufacturing, integrated circuits (ICs) are designed and fabricated in a globalized multi-vendor environment, leading to concerns such as IC piracy, overproduction and counterfeiting [1]. The malicious foundry can reverse-engineer a GDSII layout file to obtain its gate-level netlist, overbuild the IC and introduce illegal copies into the market, leading to serious economic loss to IC design companies [2], [3].

One of the techniques that thwarts such threats by an untrusted foundry is logic encryption [4]–[9]. In logic encryption[1], key-controlled logic gates called *key gates*, such as XOR gates, and an on-chip tamper proof memory are embedded into an IC to hide its original functionality. The key inputs are driven from

This paper has been accepted in ASP-DAC 2017 and its final version will appear in IEEE Explore.

[1]Logic encryption has also been referred to as logic locking, and logic obfuscation in literature [4]–[9].



(a) Example circuit: majority function [12].

(b) Circuit encrypted using XOR/XNOR key gates. The correct key $\{K1, K2, K3\}$ value is 101.

Fig. 1. Logic encryption using XOR/XNOR gates [4]. This technique is vulnerable to SAT attack [12].

the on-chip memory; the correct functionality of the IC is restored only when the correct key is driven from the on-chip memory. In existing literature, XOR/XNOR based logic encryption [4], [6], [7], MUX based logic encryption [7], [10], and look-up table based logic encryption [5], [11] have been proposed. The prime objective of these logic encryption techniques is to insert key gates such that the Hamming distance (HD) between the outputs on applying the correct key and a wrong key is 50%.

Figure 1(a) presents an example netlist and Figure 1(b) shows a logic encrypted netlist by inserting three key gates, $\mathrm{G_{K1}}, \mathrm{G_{K2}}$, and $\mathrm{G_{K3}}$. One of the inputs to a key gate is driven by a net in the original design while the other one, referred to as key input, is driven from an on-chip tamper proof memory. If correct key input to a key gate is 1, the corresponding XOR key gate implements an inverter whereas if the correct key input is 0, the XOR key gate implements a buffer. As an attacker does not know the correct key, he is unable to infer the (lack of) inversions in the netlist.

Research efforts have been carried to exploit the weaknesses of different combinational logic encryption techniques and subsequently mount attacks to extract the secret key [6], [7], [12]. While most of these attacks target specific weakness of the target logic encryption technique, the strongest attack was mounted using Boolean satisfiability (SAT) [12]. SAT attack can effectively break logic encryption techniques proposed in [4]–[8] within a few hours even for large key sizes. SAT attack determines the correct key by using a small number of input patterns called *distinguishing input patterns* (DIPs). The outputs for these DIPs are compared with the correct outputs observed from the activated functional chip that can be obtained from IC market. These input/output pairs are determined by a sequence of SAT formulas that are solved by the state-of-the-art SAT solvers.

To mitigate SAT attack, a lightweight circuit block called Anti-SAT was proposed in [13]. The structure of Anti-SAT block is shown in Figure 2. In this architecture, a subset of the key inputs, $\mathrm{KI_A}$, locks the original circuit while the remaining subset, $\mathrm{KI_B}$, drives Anti-SAT block. The key inputs $\mathrm{KI_B}$ thwart the SAT attack as the number of required SAT attack iterations to recover the correct key is exponential in

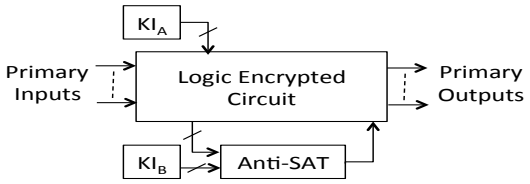


Fig. 2. Anti-SAT block for mitigating SAT attack [13].

the size of $\mathrm{KI_B}$. In this paper, we attempt to answer "Can Anti-SAT really protect logic encryption?". The contributions of the paper are as follows:

1) We perform security analysis of Anti-SAT and expose structural traces in the netlist that enables identification and removal of Anti-SAT from the logic encrypted circuit.
2) We develop *signal probability skew (SPS)* attack that breaks Anti-SAT within minutes. SPS attack is highly scalable to large circuits. The attack is more effective with increasing key size.
3) SPS attack is guaranteed to detect such traces even in the presence of structural or functional obfuscation; protection against SAT attack mandates existence of structural traces that can identify the output gate of Anti-SAT block.

II. Preliminaries

A. SAT Attack

The main idea of the SAT attack is to reveal the correct key by selectively applying the DIPs to a functional IC [12]. The attack rules out incorrect key values by using DIPs iteratively. A DIP is an input value for which at least two unique key values, k_1 and k_2, produce differing outputs, o_1 and o_2, respectively. Since o_1 and o_2 are different, at least one of the key values is incorrect. It is possible for a single DIP to rule out multiple incorrect key values.

Example. Let us consider an example SAT attack on the logic encrypted circuit shown in Figure 1(b). Figure 3 represents the output values of the encrypted circuit for different key input combinations. The values $(k0, \ldots, k7)$ represent all possible values for three key inputs $\{K1, K2, K3\}$. When SAT attack is launched, it takes four DIPs to obtain the correct key. The last column in the table lists the keys eliminated in each iteration. For example, in iteration 4, the pattern 100 is used that eliminates all incorrect keys, and thus identifies $k5$ as the correct key.

The efficiency of SAT attack depends on the order of choosing the DIPs. The total execution time for SAT attack comprising λ iterations with t_i as the execution time for the i-th iteration is $T = \sum_{i=1}^{\lambda} t_i$ [13]. SAT attack can be mitigated if either t_i or λ increases exponentially with the key size.

B. Anti-SAT Block

The SAT attack requires the maximum number of (X_d, I_d) pairs to eliminate all incorrect keys, if each such pair eliminates at most one incorrect key in an iteration. When this condition holds, the number of required SAT iterations is exponential in the number of key inputs, rendering SAT attack computationally infeasible for large key inputs. Anti-SAT block is constructed based on this condition, which exponentially increases the total execution time of the mounted SAT attack [13]. The block shown in Figure 4(a) comprises two blocks, $B_1 = g(X, K_{l1})$ and $B_2 = g(X, K_{l2}) = \overline{g(X, K_{l2})}$. These blocks share the same inputs X, but are encrypted with different keys K_{l1} and K_{l2}; the two blocks produce complementary outputs when correct keys are applied. The outputs of B_1 and B_2 drive an AND/OR gate to produce the output signal Y.

An instance of Anti-SAT block is shown in Figure 4(b); this running example is used in [13] as well. At the inputs of B_1 and B_2, a set of XOR/XNOR key gates are inserted. The number of key inputs are the same as the number of signals tapped from the logic encrypted circuit, i.e., $|K_{l1}| = |K_{l2}| = |X| = n$. The resulting key size is thus $2n$. The output Y is implemented as $Y = g(X \oplus K_{l1}) \wedge \overline{g(X \oplus K_{l2})}$. The output Y is 0 in this example when the correct keys K_{l1} and K_{l2} are applied. For incorrect keys, Y may take on the value 1 , flipping an internal net in the netlist. Inversions may be added to produce a value of 1 as the correct value of Y.

The SAT attack complexity on Anti-SAT block to decode the $2n$ key bits is defined in terms of the number of input vectors that make the function g equal to 1, i.e., the on-set of g [13]. For an n-bit input

abc	Y	k0	k1	k2	k3	k4	k5	k6	k7	Incorrect keys detected
000	0	1	1	1	1	1	0	1	1	
001	0	1	1	1	1	1	0	1	1	
010	0	1	1	1	1	1	0	1	1	
011	1	1	1	1	1	0	1	1	1	iter 1: k4
100	0	1	1	1	1	1	0	1	1	iter 4: rest incorrect keys
101	1	1	0	1	1	1	1	1	1	
110	1	1	1	1	1	1	1	1	0	iter 3:k7
111	1	1	1	0	1	1	1	1	1	iter 2: k2

Fig. 3. Analysis of SAT attack against logic encryption [12], [14].

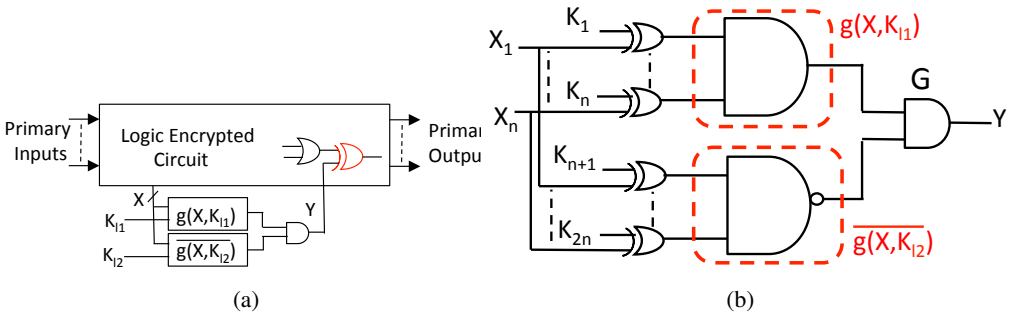


Fig. 4. a) Integrating Anti-SAT block into the logic encrypted circuit [13], b) An instance of Anti-SAT block to resist SAT attack [13].

vector $\mathbf{L} \in \{0,1\}^n$, such input vectors are elements of the set,

$$L^T = \{\mathbf{L} | g(\mathbf{L}) = 1\}, |L^T| = p \tag{1}$$

Anti-SAT constructs g in such a way that the number of such input vectors p is close to either 1 or $2^n - 1$. For the block in Figure 4(b), $p = 1$. The lower bound on the number of SAT attack iterations to recover the $2n$ key bits of Anti-SAT block is $\lambda_l = \frac{2^{2n}-2^n}{p(2^n-p)}$ [13]. For $p \in \{1, 2^n - 1\}$, the number of required iterations λ_l is 2^n, i.e., exponential in half the number of key bits in Anti-SAT block. So, the SAT attack resilience of Anti-SAT hinges on p being very small or very large. As Anti-SAT block provides a provable measure to increase the SAT attack complexity exponentially, the conventional logic encryption techniques need to be combined with Anti-SAT block to obtain foolproof logic encryption.

C. Secure and Random Integration of Anti-SAT

The SAT attack resilience of Anti-SAT attack also depends on the internal nets that drive the inputs of Anti-SAT block. Two integrations of Anti-SAT with original logic encrypted circuit are considered in [13]: *secure integration* and *random integration*.

Secure Integration. In this scheme:

- the n inputs of Anti-SAT are driven by n primary inputs of the logic encrypted circuit, and
- the output Y is connected to a wire in the original logic encrypted circuit that is among those within the top 30% observability.

Random Integration. In this scheme, the inputs as well as the output of Anti-SAT block are connected to random wires in the logic encrypted circuit. The SAT attack results show that secure integration provides more resilience than random integration as it requires more iterations, resulting in a larger execution time to reveal the secret key [13].

D. Logic Obfuscation in Anti-SAT Block

A trivial attack could simulate the circuit and find the complementary pair of signal outputs of g and $\bar{g}$ leading to the identification and removal of Anti-SAT block. Key gates can be inserted at the inputs of Anti-SAT block to break complementary relations between signals, thereby, providing *functional obfuscation*.

Another simple attack could be in the form of circuit partitioning to identify the isolated Anti-SAT block and remove it from the netlist. To thwart such attacks, *structural obfuscation* based on MUX-based logic encryption was proposed to increase the inter-connectivity between the logic encrypted circuit and Anti-SAT block [13]. The obfuscated Anti-SAT block (OA) will have $4n$ key gates.

Example. An instance of functional and structural obfuscation for the logic encrypted circuit in Figure 1(b) is shown in Figure 5. The outputs of gates $G8$ and $G10$ form the output signals of the functions

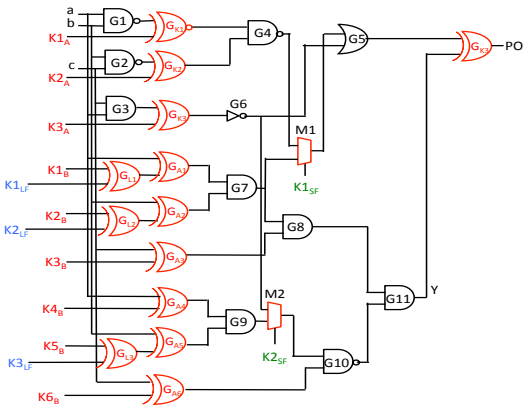


Fig. 5. Functional and structural obfuscation between Anti-SAT and logic encrypted circuit of Figure 1(b). $\{K1_A, \ldots, K3_A\}$ are the key inputs to the logic encrypted circuit, $\{K1_B, \ldots, K6_B\}$ are the key inputs to Anti-SAT circuit, $\{K1_{LF}, \ldots, K3_{LF}\}$ (shown in blue color) are the key inputs for functional obfuscation, and $\{K1_{SF}, K2_{SF}\}$ (shown in green color) are the key inputs for structural obfuscation. $M1$ and $M2$ are used for MUX-based logic encryption.

g and $\bar{g}$, and hence are complementary signals; an attacker can attempt to find potential complementary pair of signals leading to identification of Anti-SAT block. Anti-SAT block, comprising an additional set of three key gates $\{G_{L1}, G_{L2}, G_{L3}\}$, obfuscates the pair of complementary signal outputs. Further, the MUXes $M1$ and $M2$ in the figure are used to increase the inter-connectivity of the logic encrypted circuit and Anti-SAT block. This structural obfuscation of Anti-SAT renders the identification of Anti-SAT block difficult for the attacker, as the boundary between the two blocks are obscured.

III. Security Analysis of Anti-SAT and SPS Attack

The main vulnerability of Anti-SAT is that it is incorporated into the netlist at a single point, where its output Y is XORed with an internal net. Therefore, Anti-SAT defense relies on various obfuscations that make the identification of the block and its output difficult. At the same time, SAT attack resilience is ensured by having a skewed p value irrespective of all the structural and functional obfuscations. This basic construction principle inevitably leads to structural traces that help identify Anti-SAT block output in a given netlist.

As the p value has to be very small or very large to attain resilience against SAT attacks, a signal probability skew attack can point out potential candidates in a netlist for an Anti-SAT block output. The two complementary blocks of Anti-SAT produce oppositely skewed signals that converge at a gate, whose output is Anti-SAT output Y that is integrated into the netlist. The functional and structural obfuscation techniques utilized in [13] may help hide Anti-SAT block structure and make it seem a part of the original netlist; unfortunately, the signal skews remain as traces of Anti-SAT even upon applying the obfuscation techniques.

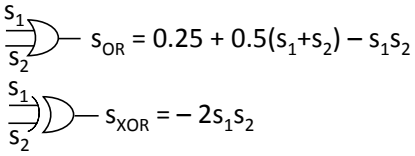


Fig. 6. SPS of OR and XOR gate outputs where s_1 and s_2 are the SPS of the inputs of the gates.

Any attempt to harden Anti-SAT against the proposed signal probability skew attack by reducing the skew in the p values would make Anti-SAT vulnerable to SAT attacks. Anti-SAT is thus cornered by the SAT attacks and the proposed signal probability skew attack.

A. Signal Probability Skew

We define *signal probability skew* (SPS) of a signal x as,

$$s = Pr[x = 1] - 0.5 \tag{2}$$

where, $Pr[x = 1]$ indicates the probability that signal x is 1. As $0 \leq Pr[x = 1] \leq 1$, the range of s is $[-0.5, 0.5]$. The SPS of a signal denotes the amount by which a signal is distinguishable from a random guess, i.e., $\mathrm{Pr[x = 1]} = 0.5$. An attacker is said to have negligible advantage of guessing the signal value over random guessing if the corresponding SPS s is close to zero. For instance, all primary inputs and key inputs are equi-probable, hence their skew is zero.

Consider a two-input AND gate with inputs in_1 and in_2 with the corresponding SPS values s_1 and s_2, respectively. The SPS of the output, s_{AND} is defined as,

$$\begin{aligned} s_{AND} &= Pr[y = 1] - 0.5 = Pr[in_1 = 1]Pr[in_2 = 1] - 0.5 \\ &= 0.5(s_1 + s_2) + s_1 s_2 - 0.25 \end{aligned} \tag{3}$$

If the inputs to an AND gate have zero SPS values, then $s_{AND} = -0.25$, demonstrating the skew that every AND gate introduces. The SPS of OR and XOR are shown in Figure 6. It can also be noted that OR gates add a positive skew, while XOR gates reduce the absolute skew, restoring it closer to zero.

In MUX-based logic encryption, the select input of MUX is a key input with zero skew; the data inputs are intermediate signals from Anti-SAT and the logic encrypted circuit. The SPS of a MUX output can be derived as,

$$s_{MUX} = 0.5(s_1 + s_2) \tag{4}$$

where s_1 and s_2 are the SPS of the inputs. Neither XOR nor MUX-based logic encryption introduces skew in secure integration as key gates are inserted at the primary inputs.

B. SPS Attack on Anti-SAT

In this section, we propose signal probability skew attack that detects the output signal Y of Anti-SAT. The threat model of SPS attack is identical to that of SAT attack [12], and Anti-SAT [13]. We show that

the *absolute difference of the probability skew (ADS)* of the inputs of a gate is maximum for the gate G, the output of Anti-SAT block.

Let us consider the skew of individual gates in Anti-SAT block shown in Figure 4(b). The XOR key gates produce zero skew signals. The blocks $g(X, K_{l1})$ and $\overline{g(X, K_{l2})}$ comprise an n-input AND and an n-input NAND gate, respectively. The SPS s_{n-AND} for the AND gate is defined as,

$$s_{n-AND} = \prod_{i=1}^{n}(0.5 + s_i) - 0.5 \tag{5}$$

where s_i is the SPS of the i^{th} input. As $s_i = 0$, the SPS of n-input AND gate in $g(X, K_{l1})$ is,

$$s_{g(X,K_{l1})} = 0.5^n - 0.5 \tag{6}$$

For large n, $s_{g(X,K_{l1})} \approx -0.5$, indicating $p \approx 1$. Similarly, for the n-input NAND gate output in $\overline{g(X, K_{l2})}$, the SPS is,

$$s_{n-NAND} = 0.5 - \prod_{i=1}^{n}(0.5 + s_i) \tag{7}$$

As $s_i = 0$, the SPS of the NAND gate in $\overline{g(X, K_{l1})}$ is,

$$s_{\overline{g(X,K_{l1})}} = 0.5 - 0.5^n. \tag{8}$$

For large n, $s_{\overline{g(X,K_{l1})}} \approx 0.5$, indicating $p \approx 2^n - 1$. The absolute difference of the probability skew of the inputs of the AND gate G, ADS_G, can be computed as,

$$ADS_G = |s_{g(X,K_{l1})} - s_{\overline{g(X,K_{l1})}}| = 1 - 2 \times 0.5^n \tag{9}$$

If the number of inputs to Anti-SAT block is high, $ADS_G = |s_{g(X,K_{l1})} - s_{\overline{g(X,K_{l1})}}| \cong 1$. ADS_G close to 1 indicates that the two inputs of the gate G exhibit highest skews with opposite polarity. This property of gate G distinguishes it from the rest of the gates in Anti-SAT block.

SPS attack on a logic encrypted circuit with Anti-SAT block comprises computing the SPS of all the gates in the circuit. The gate with the highest SPS, i.e., a gate with oppositely skewed inputs is the suspect gate G, the output gate of Anti-SAT block.

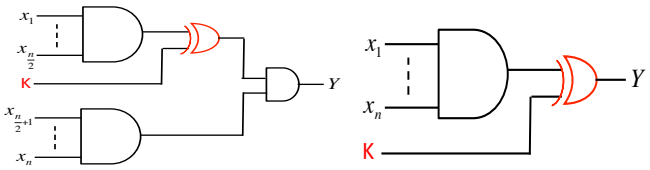
SPS attack applies to arbitrary g and $\bar{g}$. In case of n-input OR gate and n-input NOR gate for the functions g and $\bar{g}$, the corresponding SPS values are,

$$s_{n-OR} = 0.5 - \prod_{i=1}^{n}(0.5 - s_i), \tag{10}$$

$$s_{n-NOR} = \prod_{i=1}^{n}(0.5 - s_i) - 0.5 \tag{11}$$

C. SPS Attack on Functionally Obfuscated Anti-SAT

Now, we show how SPS attack is effective in presence of functional obfuscation, where n additional key gates are inserted at the internal wires of Anti-SAT block. These key gates affect the skew of the gate G; the change in the skew depends upon the location of the key gates.



(a) Key gate inserted inside the tree of n-input AND gate.

(b) Key gate inserted at the output of n-input AND gate.

Fig. 7. Inserting key gates inside the AND gate tree of function g in Anti-SAT changes the probability skew to assist SPSA detection scheme, while placing it outside the AND gate output helps SAT attack scheme.

TABLE I

ADS VALUES OF COMBINATIONAL GATES OF ANTI-SAT BLOCK IN FIGURE 5 IN DESCENDING ORDER.

Gate	$G11$	$M1$	$M2$	$G8$	$G10$
ADS	0.6875	0.5	0.25	0.25	0.125

Let us consider an n-input AND gate that constitutes the function g in Anti-SAT block. In Figure 7(a), the XOR key gate is inserted at a net inside the AND-tree, at the input of final AND gate in this specific case. Let us assume s_1 and s_2 represent the skew at the inputs of the final AND gate. Prior to insertion of the key gate, $s_1 = s_2 = 0.5^{\frac{n}{2}} - 0.5$, and $s_{n-AND} = 0.5^n - 0.5$ for the AND-tree. After the insertion of the key gate, $s_1 = 0$, and hence the modified skew of the n-input AND becomes $s'_{n-AND} = 0.5^{\frac{n}{2}+1} - 0.5$.

When the key gate is moved further to the output of AND gate as shown in Figure 7(b), $s_Y = 0$. In other words, $p = 2^{n-1}$, and the SAT attack can break Anti-SAT in $\lambda_l = \frac{2^{2n}-2^n}{p(2^n-p)} = 4$ iterations. Thus, inserting key gates closer to the output of the AND/NAND tree significantly reduces the resilience against the SAT attack. For higher SAT attack resilience, the key gates must be placed closer to the inputs of Anti-SAT block. However, in that case, Anti-SAT becomes vulnerable to SPS attack.

D. SPS Attack on Structurally Obfuscated Anti-SAT

In structural obfuscation, n MUXes are inserted in Anti-SAT block; one input of each MUX is driven by a random net in the logic encrypted circuit. The SPS of the output of a MUX is the average of the SPS of its inputs. Similar to the case for functional obfuscation, MUXes should be placed closer to the inputs of Anti-SAT block, otherwise the resilience to SAT attack will be minimal. SPS attack will be ineffective against structural obfuscation only if the SPS_G becomes close to zero, however, in that case, SAT attack is more effective.

E. Removal Attack on Anti-SAT

In SPS attack, the gate G is identified using the highest ADS trace. The logic encrypted circuit may contain few signals that exhibit high ADS values, close to ADS_G. These false candidates can be filtered out by checking for simple structural traces. By analyzing the transitive fan-in (TFI) of the candidate

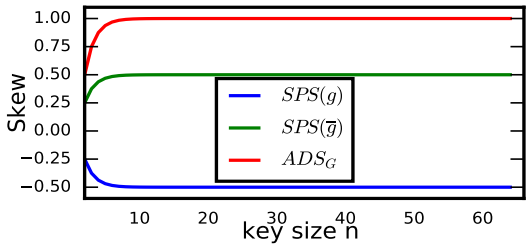


Fig. 8. Impact of n on ADS_G, the absolute difference of skew at the inputs of gate G, the output of Anti-SAT block, for $p = 1$. $SPS(g)$ and $SPS(\bar{g})$ represent the skew of the AND and NAND tree, which constitute Anti-SAT block.

gates and eliminating the gates whose TFI do not include at least $2n$ key inputs, we can correctly identify the gate G. This is further illustrated in Section IV.D.

Once G has been identified, the value of the output signal Y can be determined from s_Y. If $s_Y < 0$, the value of Y in the functional IC is 0, otherwise it is 1. Knowing the correct value of Y, one can trace back and discard the gates that are in the fan-in of signal Y alone. Alternatively, the signal Y can be set to 0 (or 1, if $s_Y > 0$) during simulation, which effectively removes Anti-SAT block. To identify the values for the key gates in the logic encrypted circuit, SAT attack can be launched on the logic encrypted circuit.

Example. The objective of SPS attack on the circuit in Figure 5 is to identify the output gate of Anti-SAT block, i.e., $G11$. The highest five ADS values for the circuit are shown in Table I. The pair of complementary signals, $G8$ and $G10$ with opposite SPS values leads to the highest ADS for $G11$, enabling the correction detection of the output of Anti-SAT block. The SPS for the output of $G11$ is $s_Y = -0.398$, implying that the signal is skewed towards 0.

IV. Experimental Results

A. Experimental Setup

SPS attack experiments are conducted using ISCAS benchmark circuits [15] and OpenSPARC microprocessor controllers [16]. SPS attack as well as SAT attack are executed on a server with 6-core Intel Xeon W3690 CPU, running at 3.47GHz, with 24 GB RAM [12]. Anti-SAT block is integrated with *fault analysis based logic encryption* [7], which is referred to as TOC'13(5%), following the convention used in [13].

B. Impact of Key Size (n) on SPS Attack

The number of keys in basic Anti-SAT block is $2n$, where n is the number of keys in individual blocks g and $\bar{g}$. For SPS attack to be effective ADS_G must increase with n. Figure 8 demonstrates that as n increases, ADS_G increases exponentially at first and is then saturated close to a value of 1. SPS attack is successful when ADS_G is close to 1, representing a gate whose inputs are skewed towards opposite values. As an example, for $n = 16$, the skew at the output of the block g (an AND tree) will be ≈ -0.5,

whereas the the skew at the output of the block $\bar{g}$ (a NAND tree) will be ≈ 0.5. The ADS_G will be ≈ 1. ADS_G approaches 1 as n increases, and can be used to identify the gate G. Thus, *the attack effectiveness increases with n*, which is counter-intuitive for any attack.

C. SAT Attack vs. SPS Attack

Impact of p on Attack Success. Anti-SAT block offers highest resistance against SAT attack when $p \approx 1$ or $p \approx 2^n$; then, the number of iterations for SAT attack is $\approx 2^n$. The resistance is the least when $p \approx 2^{n-1}$. Figure 9 displays SAT attack resistance normalized by $2^n = 65536$ for $n = 16$.

The resistance to SPS attack can be represented as $1 - ADS_G$. When $ADS_G \approx 0$, the resistance to SPS attack is the maximum; this also implies p$\approx 2^{n-1}$ and minimum resistance to SAT attack. The resistance to SPS attack is minimum when $p \approx 1$ or $p \approx 2^n$ as demonstrated in Figure 9; for these of p, SAT attack resistance is the maximum. Thus, *the two attacks are complementary to each other. One of the attacks is highly effective for any value of p*. The regions of effectiveness of SPS and SAT attack are shown as red and blue regions, respectively, in Figure 9.

Attack Execution Time. Figure 10 shows that the execution time of SAT attack depends on the value of p, which dictates the number of iterations of the attack. For $p = 1$ and $p = 65535$, the attack takes more than a day to complete. For SPS attack, which involves computing the signal probabilities of few gates (≈ 100 for $n = 16$), the attack time is in few seconds, and practically negligible compared to the execution time of SAT attack.

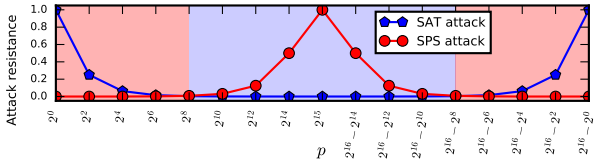


Fig. 9. Normalized attack resistance of Anti-SAT block for n=16. For SAT attack, the resistance is the number of iterations of the attack normalized by 65536. For SPS attack, the resistance is specified as 1-ADS_G. SPS attack is highly effective in region shaded red; SAT attack in the region shaded blue.

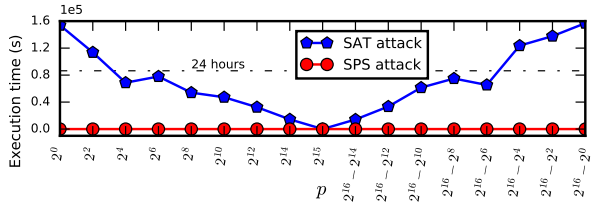


Fig. 10. Execution time of SAT attack and SPS attack on basic Anti-SAT block for $n = 16$. The execution time of SAT attack is more than day for $p \in \{1, 2^n - 1\}$, whereas, the execution time of SPS attack is less than 2 minutes for all values of p.

TABLE II

SPS ATTACK ON SECURE INTEGRATION FOR $p = 1$. HC ADS REPRESENTS THE HIGHEST ADS VALUE FOR THE GATES IN THE ENCRYPTED CIRCUIT (EXCLUDING THE GATES IN ANTI-SAT). THE BENCHMARKS ARE SORTED BY THE NUMBER OF GATES. $\#cand$ REPRESENTS THE NUMBER OF CANDIDATES FOR GATE G.

			$n = 16$		$n = 64$	
Benchmark	# gates	HC ADS	$\#cand$	Exec. time (s)	$\#cand$	Exec. time (s)
s5378	1214	0.97978	1	0.3	1	0.6
ifu dcl	1384	0.74609	1	0.4	1	0.5
fpu in	1501	0.8125	1	0.3	1	0.6
lsu rw	1501	0.8125	1	0.7	1	1.1
lsu excp	1651	0.81211	1	0.9	1	0.6
s9234	1677	0.98526	1	0.6	1	0.8
fpu div	2137	0.8125	1	0.5	1	1.0
lsu stb	2371	0.93749	1	1	1	0.7
c5315	2695	0.5616	1	0.6	1	0.8
c7552	2697	0.58069	1	0.8	1	1.1
ifu ifq	3663	0.92281	1	2	1	1.9
tlu mmu	5559	0.98828	1	4.8	1	4.6
s13207	13371	0.99994	1	18.2	1	20.1
s15850	15876	0.99999	3	18.3	1	19.1
s35932	16457	0.60127	1	47.8	1	43.4
s38584	19511	0.99805	1	55.7	1	56.2
s38417	22501	0.99644	1	54.4	1	57.7

D. SPS Attack on Secure Integration

In secure integration of Anti-SAT block (referred to as TOC'13(5%)+n-bit BA (Basic Anti-SAT) in [13]), n inputs of Anti-SAT block are connected to n primary inputs of the logic encrypted circuit [13]. ADS_G is represented as $1 - 0.5^{n-1}$, irrespective of the logic encrypted circuit. For a successful attack, ADS_G must be higher than the ADS of the rest of the gates in the circuit.

Table II presents the results for SPS attack on secure integration. The column "HC ADS" displays the highest ADS value for the gates in the original circuit (not including the gates in Anti-SAT block). With $n = 16$, the gates with $ADS \geq (1 - 0.5^{15})$ are candidates for the gate G. We observe that there is only one candidate for gate G in all the circuits except s15850. The circuit s15850 has two gates whose ADS values are higher than the ADS_G.

As mentioned in section III.E, the false candidates for G are filtered out by analyzing the TFI of the candidate gates and eliminating the gates whose TFI do not include $2n$ key inputs. The attack then correctly identifies G in all of the circuits. The execution time of SPS attack is less than two minutes for all the circuits in the study, which have up to 22K gates. Thus, the attack scales well for large circuits.

E. SPS Attack on Random Integration

In random integration, referred to as TOC'13(5%)+n-bit OA (Obfuscated Anti-SAT), the n Anti-SAT inputs of Anti-SAT block are connected to randomly selected wires in the logic encrypted circuit. $2n$ additional key gates are added for functional and structural obfuscation. Overall, TOC'13(5%)+n-bit OA

has $4n$ key gates in Anti-SAT block, in addition to 5% key gates in the logic encrypted circuit. SPS attack will be successful only if ADS_G values do not deviate significantly as a result of obfuscation. However, if ADS_G deviates from its desired value of 1, the value of p tends towards 2^{n-1}, and the circuit becomes vulnerable to SAT attack.

Table III presents the results for SPS attack on TOC'13(5%)+n-bit OA. It can be noted ADS_G changes slightly across the benchmarks now. There is only one candidate for the gate G in all the circuits except s15850; the circuit is an exception with three candidates. The true candidate is found by analyzing the TFI of the candidates, as mentioned in section III.E. Thus, *the attack is 100% successful on Anti-SAT even with functional and structural obfuscation. The attack time is less than two minutes for circuits with up to 22K gates.*

TABLE III

SPS ATTACK ON TOC'13(5%)+n-BIT OA FOR $p = 1$. ADS_G SLIGHTLY CHANGES ACROSS THE BENCHMARK CIRCUITS, WHEN $n = 16$. $\#cand$ REPRESENTS THE NUMBER CANDIDATE FOR THE GATE G.

	$n = 16$			$n = 64$	
Benchmark	ADS_G	$\#cand$	Exec. time (s)	$\#cand$	Exec. time (s)
s5378	0.999967	1	0.4	1	1.0
ifu dcl	0.999971	1	0.7	1	1.1
fpu in	0.999971	1	0.7	1	1.4
lsu rw	0.999971	1	0.6	1	1.6
lsu excp	0.99996	1	0.6	1	1.1
s9234	0.999967	1	0.8	1	1.2
fpu div	0.999973	1	0.9	1	1.4
lsu stb	0.999973	1	0.9	1	1.4
c5315	0.999969	1	0.9	1	1.2
c7552	0.999973	1	0.8	1	1.4
ifu ifq	0.999969	1	0.9	1	2.1
tlu mmu	0.999971	1	1.4	1	2.4
s13207	0.999973	1	4.1	1	6.8
s15850	0.999971	3	19.2	2	22.0
s35932	0.999972	1	28.2	1	30
s38584	0.999972	1	75.1	1	83.2
s38417	0.999973	1	89.3	1	93.2

V. CONCLUSION

In this paper, we propose signal probability skew attack to break Anti-SAT block. We demonstrated that the output gate of Anti-SAT block has a characteristic trace: the difference of probability skew at

its inputs is high. SPS attack enables removal of Anti-SAT block by identifying its output gate, even if the block is structurally and functionally obfuscated. While SAT attack requires more than a day to break Anti-SAT block, the proposed SPS attack breaks it within minutes, with 100% success rate. Attack results demonstrate that SPS attack becomes more effective with increasing key size.

REFERENCES

[1] "Defense Science Board (DSB) study on High Performance Microchip Supply," 2005, [March 16, 2015]. [Online]. Available: www.acq.osd.mil/dsb/reports/ADA435563.pdf

[2] SEMI, "Innovation is at Risk Losses of up to $4 Billion Annually due to IP Infringement," 2008, [June 10, 2015]. [Online]. Available: www.semi.org/en/Issues/IntellectualProperty/ssLINK/P043785

[3] 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.

[4] J. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending Piracy of Integrated Circuits," in *Proc. Design, Automation and Test in Europe*, 2008, pp. 1069–1074.

[5] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC Piracy Using Reconfigurable Logic Barriers," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 66–75, 2010.

[6] M. Yasin, J. Rajendran, O. Sinanoglu, and R. Karri, "On Improving the Security of Logic Locking," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2016.

[7] J. Rajendran, H. Zhang, C. Zhang, G. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault Analysis-Based Logic Encryption," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 410–424, 2015.

[8] R. S. Chakraborty and S. Bhunia, "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 10, pp. 1493–1502, 2009.

[9] F. Koushanfar, "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management," *EEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 51–63, 2012.

[10] S. M. Plaza and I. L. Markov, "Solving the Third-Shift Problem in IC Piracy with Test-Aware Logic Locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 961–971, 2015.

[11] B. Liu and B. Wang, "Embedded Reconfigurable Logic for ASIC Design Obfuscation Against Supply Chain Attacks," in *Proceedings of the conference on Design, Automation & Test in Europe*. European Design and Automation Association, 2014, p. 243.

[12] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the Security of Logic Encryption Algorithms," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust*, 2015, pp. 137–143.

[13] Y. Xie and A. Srivastava, "Mitigating SAT Attack on Logic Locking," *IACR Cryptology ePrint Archive*, vol. 2016, p. 590, 2016.

[14] 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*, 2016, pp. 236–241.

[15] M. C. Hansen, H. Yalcin, and J. P. Hayes, "Unveiling the ISCAS-85 Benchmarks: A Case Study in Reverse Engineering," *IEEE Des. Test. Comput.*, vol. 16, no. 3, pp. 72–80, 1999.

[16] "OpenSPARC T1 Processor,," 2015, [Nov 1, 2015]. [Online]. Available: {http://www.oracle.com/technetwork/systems/opensparc/opensparc-t1-page-1444609.html}