Trust in IoT Devices: A Logic Encryption Perspective

Yasaswy Kasarabada, David Luria, Ranga Vemuri

Trust in IoT Devices: A Logic Encryption Perspective

Yasaswy Kasarabada, David Luria, and Ranga Vemuri
Digital Design Environments Laboratory
University of Cincinnati, Cincinnati OH, USA
{kasarayv, luriadm}@mail.uc.edu, vemurir@ucmail.uc.edu

Abstract. Tremendous technological advancement has led to the development of an ecosystem of highly connected ubiquitous computing devices called the Internet of Things (IoT). Considering the sensitive nature of the data collected by the IoT devices, it is essential to ensure the security of these devices. Logic encryption is a popular design-for-trust technique used for protection against hardware IP piracy, design counterfeiting, and hardware Trojan insertion. The introduction of various attack methods that leverage vulnerabilities in known logic encryption techniques has prompted the development of new techniques able to thwart the proposed attacks. Major research effort in the field of logic encryption focuses on increasing resilience to known and potential attacks while often ignoring considerations of cost overhead, especially die area and power consumed. Since area and power optimization are key aspects in the design and development of most IoT devices, it is important to evaluate the cost incurred by logic encryption schemes, especially in the context of these two metrics. In this paper, we survey some of the most popular logic encryption and decryption techniques proposed in the past decade. An analysis of the area and power overhead of these logic encryption techniques using several standard benchmark circuits is presented to assess their suitability for resource constrained systems.

Keywords: Logic Encryption · Internet of Things · Trust · Security

1 Introduction

The proliferation of ubiquitous computing devices has led to the creation of a densely connected global network - IoT. Many of these devices (like wearable technology) continuously measure and transmit sensitive and critical data. Therefore, it becomes essential to incorporate a security mechanism into the device to protect against prospective attack scenarios. Implementation of traditional cryptographic algorithms that are effective against cyber attacks is typically not light-weight, an important requirement for the resource-constrained IoT devices. Hardware-based security measures like logic encryption are suitable alternatives for protecting IoT devices against foundry based attacks like reverse engineering, counterfeiting, and piracy.

Logic encryption is a hardware protection technique that locks the design functionality using a set of newly introduced inputs called *key inputs*. An overview of logic encryption is shown in Figure 1. If the correct logic value (chosen by the designer) is applied on the key inputs, the design behaves as expected. An incorrect key value causes corruption of the input-output relationship of the circuit. Figure 2 shows a simple encryption scheme. Initial logic encryption schemes [14]

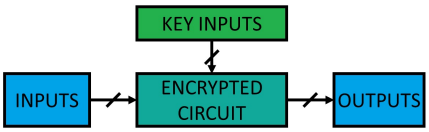


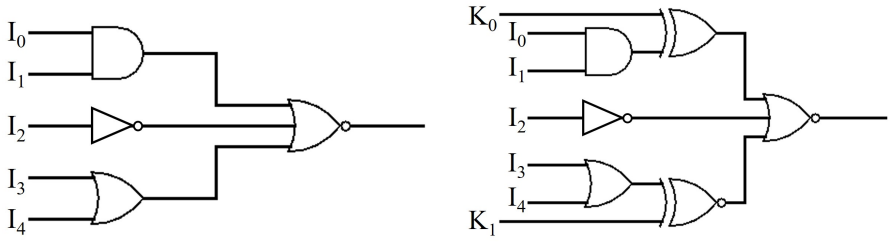
Fig. 1. Overview of Logic Encryption

proposed to insert these key-controlled gates (*key gates*) randomly into the circuit. This technique was shown to be susceptible to an attack method[12] that was able to sensitize wrong key values to primary outputs. Subsequently, encryption techniques[32, 6, 9] that focused on increasing the dependency between the key gates were proposed to thwart the sensitization attack. An oracle-guided Boolean Satisfiability (SAT) based attack[20], that uses a SAT-solver to iteratively eliminate multiple incorrect key values to obtain the correct key assignment, was able to defeat many previously proposed logic encryption methods.

The advent of the SAT-based attack led to the proposal of various SAT-resilient techniques[24, 29, 30, 10, 34, 33]. By inserting SAT-hard blocks, these techniques ensured that the attack was unable to eliminate more than a limited number of incorrect keys per iteration, leading to extremely long, unfeasible attack times. Other SAT-resilient techniques[17, 13] focused on preventing the SAT attack from modelling the circuit by introducing key controlled feedback cycles into the design. Although the proposed SAT-resilient schemes were able to defeat the original SAT attack, they were found to be susceptible to other attack methods like approximate-SAT[16], cyclic-SAT[35], removal[31], and bypass[27].

The original SAT attack[20] could directly be applied to sequential circuits if the flip-flops were accessible using scan-chains. It has been asserted in recent works[18] that disabling the scan-chains after the testing procedure is relatively inexpensive. Without a scan-accessible oracle circuit, the traditional SAT attack cannot be applied directly to sequential circuits. Therefore, new unrolling-based SAT attack methods[4, 7, 18] to decrypt sequential circuits have been proposed. These techniques have been successful in defeating popular sequential logic encryption techniques[3, 5, 7].

Research in the field of logic encryption primarily focuses on finding new attack methods that are able to defeat known encryption schemes and new defense techniques to thwart proposed attacks. Evaluation of cost metrics such as penalties on IC performance (timing, area, and power) and impact on testing (test coverage and testability) are often overlooked and rarely mentioned in the literature. Application of popular encryption techniques to resource-starved IoT devices must be undertaken after an assessment of 1) the level of security offered by these techniques, and 2) the cost incurred by their implementation. This paper aims to assist in this assessment by performing an analysis of the impact of many popular logic encryption techniques on two important cost parameters - area and power. Selected, well-known combinational and sequential logic encryption techniques are implemented on benchmark circuits from the ISCAS '85 and ITC '99 suites respectively to obtain an estimate of the area and power overhead of these schemes.



(a) Unencrypted Circuit (b) Encrypted Circuit

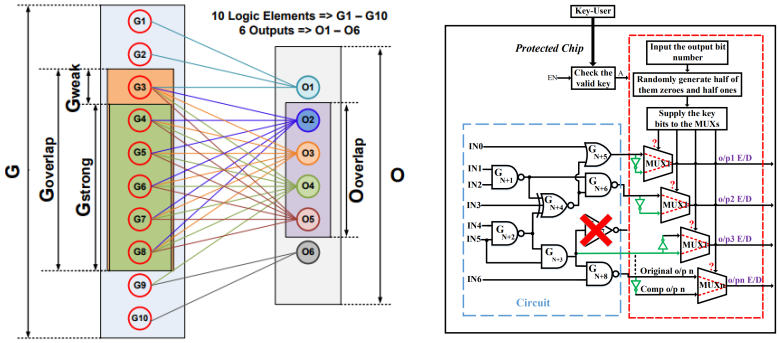
Fig. 2. Logic Encryption with Key Gates

The rest of the paper is organized as follows. Section 2 discusses some of the salient features of many popular logic encryption techniques. Major attack methods are summarized in Section 3. Resiliency of encryption techniques described in Section 2 against these attacks is presented in Section 4. Section 5 discusses our analysis of the cost (area and power) incurred by the implementation of these proposed techniques using standard benchmark circuits. Section 6 presents the conclusions derived from this work.

2 Logic Encryption Techniques

In this section, we describe several logic encryption methods proposed over the past decade. We begin by discussing traditional encryption schemes, that are vulnerable to the SAT attack, followed by the SAT-resilient techniques. Finally, we detail the sequential encryption methods and some of their important features.

1. **EPIC[14]** proposed to lock the design by inserting key controlled XOR/ XNOR gates coupled with inverters into the circuit. These gates are placed on randomly chosen nets in the circuit while avoiding critical paths. If the wrong key value is applied on any key gate, its output is inverted. Since the attacker does not know if the inverter is a part of the original design or a part of the added locking mechanism, using a direct reverse engineering attack to infer the key values may be unsuccessful.
2. The **Logic Cone Size based scheme[11]** was proposed as a technique to design a well-obfuscated hardware Trojan. Inserting key gates on nets with large fan-in and fan-out cones is advocated to avoid graph isomorphism-based equivalency attacks. Using a weighted normalized metric[2] for all gates in the design, suitable locations for key gate insertion can be chosen.
3. **Strong Logic Locking[32]** focuses on intelligent insertion of key controlled gates by evaluating two important metrics - *pairwise security* and *dominance*. Two key gates are considered to be pairwise secure if the key value on any one of these key gates cannot be sensitized to a primary output without knowing the correct key value on the other key gate. Meanwhile, a key gate lying on any path from another key gate to any primary output is said to dominate the other key gate. The key gates are inserted such that each key



(a) Key Interdependency[6] (b) E2LEMI[1]

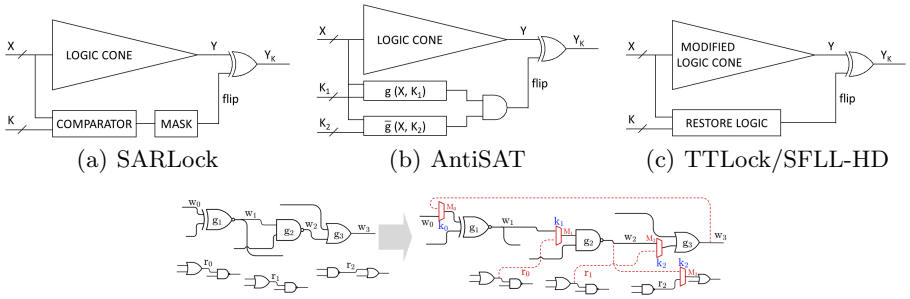
Fig. 3. Traditional Logic Encryption Techniques

gate is pairwise secure with at least one other key gate. Additionally, a one-way random function like AES is used with a subset of the in-memory key acting as its input while its output is fed to some of the key gates.

4. The **Key Interdependency based technique[6]** uses a three-step pruning process to obtain a set of gates that are considered ideal locations for key gate insertion. The first step involves finding a set of gates (G_{strong}) with the largest number of primary outputs ($O_{overlap}$) in the combined output cone of dependency of G_{strong}, as shown in Figure 3(a). Following this, the fault impact metric $FI(g) = NoP_0 \cdot NoO_0 + NoP_1 \cdot NoO_1$ for each gate g ($\in G_{strong}$) is calculated using 10000 random input patterns, where NoP_i ($i \in \{0, 1\}$) is the number of input patterns that are able to sensitize a stuck-at-i fault at g to a primary output and NoO_i ($i \in \{0, 1\}$) is the total number of outputs that are affected by the sensitization of the stuck-a-i fault at g. A dependency graph, constructed with $O_{overlap}$ and the set of gates (G_{oc}) with highest FI values, is used to calculate the *dependency* metric for each gate in G_{oc}. The gates with the highest dependency values are chosen and key gates are inserted at the inputs of these gates. The motivation behind this techniques is to increase the overlap in output cones of dependency of the added key gates while ensuring maximum interference between them.
5. A light-weight logic encryption method called **Hardware Enlightening[15]** was proposed to thwart the insertion of hardware Trojans[23] into the design. To avoid detection during the testing procedure, nets with low controllability/observability values are chosen as trigger signals for the Trojan. The encryption algorithm begins by finding nets with low probability values. For each of these signals, a key controlled XOR/XNOR gate is added on the fan-in signal with the lowest probability value and adequate slack time. After the addition of a key gate, the probability values of all nets in the circuit are re-calculated and the process is repeated for the next low probability signal. To avoid direct reverse engineering attacks, an inverter is inserted at the output of the key gate based on a hidden signature (generated by the designer).
6. To reduce power consumption, the **Energy-Efficient Logic Encryption using Multiplexer Insertion (E2LEMI)[1]** proposed to insert multi-

plexers on all primary outputs of the circuit with the output nets serving as the 'correct' inputs to the multiplexer while the inverted output nets are fed as the 'incorrect' inputs, as shown in Figure 3(b). The select input of each multiplexer is connected to the output of a linear feedback shift register (LFSR) designed to produce an output pattern with equal number of 0s and 1s. A hardware Trojan based circuit is inserted into the design to check if the correct key value is applied. On the application of an incorrect key, the payload activates the LFSR which feeds half the multiplexer select inputs with 0s and the rest with 1s. Therefore, half the multiplexer's select the correct output values while the rest select inverted output values, thus achieving 50% Hamming distance from the correct output. If the correct key is provided, the Trojan deactivates the LFSR and the correct select pattern is provided to the multiplexers such that the outputs of the design hold the correct value.

7. A part of the first wave of SAT-resilient techniques, **SARLock [29]** focuses on thwarting the SAT attack by making the attack time exponentially large. A comparator block is inserted into the circuit to compare the applied key value with the value on the inputs of a chosen logic cone. The comparator output is connected to an XOR gate placed on the output of the chosen logic cone to ensure that the output of the logic cone is corrupted when the key value is the same as the value on the input of the logic cone. Since this is undesirable for the correct key, a masking block is used to invert the output of the comparator when the correct key is applied and the comparator output is high. Since the output of the comparator becomes high for only one input vector per key value, the error rate of the encrypted circuit is very low. This could lead to the attacker using the encrypted netlist with a wrong key assignment for 2^k-1 input patterns, where k is the number of key inputs. To avoid this scenario, the authors propose to use this technique in conjunction with traditional (potentially SAT vulnerable) encryption schemes. Such a compound scheme ensures high error rates due to the traditional scheme and boasts SAT-resilience due to the SARLock instance. An overview of SARLock is shown in Figure 4(a).
8. Similar to SARLock, **Anti-SAT[24]** also uses a point function to make the SAT attack infeasible. The construction of the point function in Anti-SAT is achieved using two logic blocks operating on the same set of inputs (of the chosen logic cone) and locked with two different key inputs. The function implemented by each logic block is a complement of the function of the other logic block. When the output of both logic blocks is 1, the output of the Anti-SAT block turns high and flips the output of the chosen logic cone, thus enforcing corruption. The functions of the complementary logic blocks are chosen such that when the correct key value is applied, the output of the Anti-SAT block does not go high for any input value. If an incorrect key is applied, the output of the Anti-SAT block goes high for at most one input pattern. To increase the error rate of the encrypted circuit, Anti-SAT can be used in a compound scheme with a traditional encryption technique. Figure 4(b) shows a high-level representation of Anti-SAT.

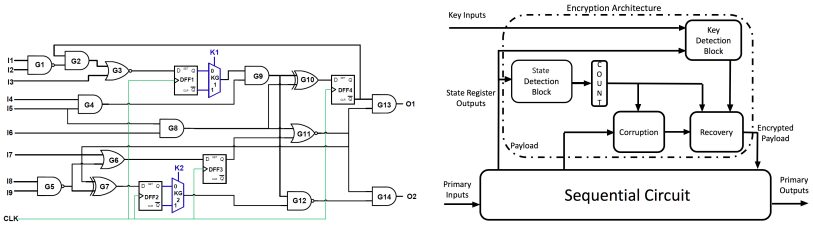


(a) SARLock (b) AntiSAT (c) TTLock/SFLL-HD

(d) Cyclic[17]

Fig. 4. SAT-Resilient Logic Encryption Techniques

9. In **TTLock[34]**, a secret input value, called *Protected Input Pattern* (PIP), is removed from the chosen logic cone such that when the PIP is applied to the cone, the output is inverted. A restoration block is added to the circuit as shown in Figure 4(c). When the input pattern is equal to the applied key value, the output of the restoration block goes high. For all incorrect key values $\vec{K}_i$ ($\neq PIP$), the output of the modified logic cone is incorrect for two input patterns - the PIP and $\vec{X}_i$, where $\vec{X}_i = \vec{K}_i$. For the correct key value $\vec{K}_C$ ($=PIP$), the output of the modified logic cone is inverted when the PIP occurs, thus obtaining the correct output value. Although the error rate for TTLock is higher than that of SARLock/Anti-SAT, it is still very low. Therefore, it is still advisable to use TTLock as a part of a compound scheme.
10. In **Stripped-Functionality Logic Locking (SFLL-HD)[33]**, similar to TTLock, the logic cone is modified and a restore logic is used to restore the output of the modified cone when the correct key value is applied. However, unlike TTLock, SFLL-HD^h inverts the output of the logic cone for multiple PIPs, where the Hamming distance between each PIP and the k-bit secret key is h. The restore block checks if the Hamming distance between the value on the input of the logic cone and the applied key is equal to h and inverts the output of the modified logic cone if this condition is satisfied. This technique can protect as many as $\binom{k}{h}$ input patterns.
11. Unlike all SAT-resilient schemes discussed so far, **Cyclic obfuscation[17]** focuses on thwarting the SAT attack by impairing the ability of the SAT solver in finding a suitable assignment for the input vector. To achieve this, combinational feedback cycles are introduced into the design that are activated on the application of an incorrect key value. 'Hard' irreducible loops with at least two removable edges are used to increase the effort of an attacker trying to remove the added cycles. As illustrated in Figure 4(d), a key controlled MUX is added on each edge (net) of these hard loops. Additionally, each reducible loop is made irreducible to make the identification and removal of the loop using depth first traversal (DFT) non-trivial.
12. **SRCLock[13]** aims at increasing the attack time by densely connecting all feedback cycles introduced by Cyclic obfuscation. Several techniques are proposed to significantly increase the number of cycles in the netlist. One of

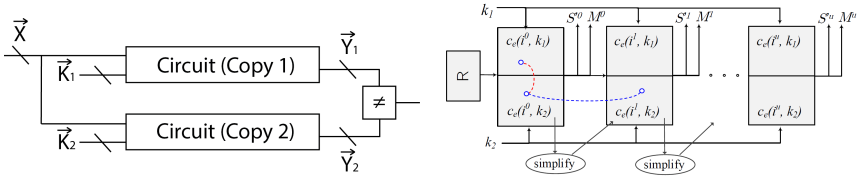


(a) Encrypt flip-flop[5] (b) Deep State[8]

Fig. 5. Sequential Logic Encryption Techniques

these techniques focuses on creating *super cycles* out of smaller *micro cycles*, such as the ones generated in cyclic obfuscation. The algorithm focuses on generating super cycles by strongly connecting all micro cycles with two-way signals between micro cycle nets. Additionally, signals in micro cycles that are not used to connect a super cycle are used in two-way connections with other unused micro cycle signals or random signals in their fan-in cone. This method results in a set of strongly connected micro cycles and an exponential increase in the number of overall cycles with a linear increase in the number of inserted feedback edges.

13. A gate-level sequential logic encryption technique, **Encrypt Flip-Flop[5]** proposes to add key-controlled MUXes on the outputs of scan-chain accessible flip-flops, as shown in Figure 5(a). The non-select inputs of each MUX are connected to the Q and $\overline{Q}$ outputs of the preceding flip-flop. If an incorrect key value is applied to the MUX, the incorrect output of the flip-flop is passed on to the next level on the following clock-edge. A placement strategy similar to the one used in the Key Interdependency technique is followed to increase the overlap in the output cone of dependency of the key gates.
14. Similar to Encrypt Flip-Flop, the **Chain Based Encryption[7]**, dictates the placement strategy of key gates, based on the relative position of the flip-flops in the design. Following an analysis of the unrolling-based Sequential SAT attack, the authors argue that encrypting long flip-flop chains ensures that the attacker needs to unroll the design for more number of clock cycles, thus increasing the complexity of the attack process. The depreciation in protection offered by the scheme when encrypting chains with large fan-out (*sneak paths*) leading outside the chain is discussed. The ensuing algorithm recommends the placement of key gates at the inputs of flip-flops with maximum distance from the nearest primary output.
15. **Deep State Encryption[8]** is a SAT-resilient gate-level scheme that incorporates a technique conceptually similar to a hardware Trojan to induce locking on the design. Two unique Monte Carlo methods are proposed to estimate a *deep state* in the design, where depth is the minimum distance (in clock cycles) from the reset state. On a chosen fixed number of occurrences of the deep state, the payload is corrupted by inverting the logic value on these signals. The corrupted signals are inverted back using a restoration block if the correct key value is applied. If an incorrect key value is applied, the payload stays corrupted and the key checking mechanism is disabled, thus



(a) Miter used in SAT attack (b) Iterative unrolling in KC2[18]

Fig. 6. Logic Decryption Techniques

forcing the design into a corrupt state forever. Figure 5(b) gives an overview of the proposed encryption architecture.

3 Logic Decryption Strategies

Over the past decade, many innovative decryption techniques against logic encryption have been proposed. In this section, we discuss notable properties of some of the most widely cited attack methods.

1. One of the first oracle guided attacks, the goal of the **Key Sensitization attack[12]** is to decipher the correct key assignment by sensitizing wrong key bits to primary outputs for observation. This is similar to the ATPG technique of sensitizing a stuck-at-fault on a net to the primary output. After finding a suitable input pattern to sensitize the required key bit, the output pattern from the encrypted circuit is compared with the output from an oracle circuit (acquired from the market) to eliminate an incorrect value on the key bit. Iteratively all incorrect values are identified to derive the correct key assignment.
2. The **SAT attack[20]** uses a SAT solver to satisfy the Quantified Boolean Formula (QBF) of a model of a miter of two encrypted circuits (as shown in Figure 6(a)). Each copy of the circuit in the miter is assigned unique keys ($\vec{K}_1$ and $\vec{K}_2$) to obtain the initial QBF, $F = C(\vec{X}, \vec{K}_1, \vec{Y}_1) \wedge C(\vec{X}, \vec{K}_2, \vec{Y}_2)$, where $\vec{X}$ is the input vector and $\vec{Y}_1$, $\vec{Y}_2$ are the output vectors. This QBF is solved by the SAT solver with an additional condition that the two output vectors must be different ($\vec{Y}_1 \neq \vec{Y}_2$) to obtain an assignment on the input vector $\vec{X}_d$, called a *distinguishing input pattern* (DIP). The DIP is applied to the oracle IC to obtain the correct output vector $\vec{Y}_d$. Either one or both the keys can be eliminated by comparing $\vec{Y}_d$ with $\vec{Y}_1$ and $\vec{Y}_2$. Additionally, all keys that result in the incorrect output(s), $\vec{Y}_1$ and/or $\vec{Y}_2$, on the application of $\vec{X}_d$ are also eliminated using this DIP. This process is continued to iteratively eliminate all incorrect key values to obtain the correct key assignment.
3. Proposed as a countermeasure to point-function based SAT-resilient schemes, **AppSAT[16]** aims to find a key assignment for which the output corruption is very low. It uses intermediate error estimation and random query reinforcement to optimize the run time of the SAT attack. After every d DIP queries, error ε is estimated using r random queries and the attack is stopped if ε stays below a chosen threshold. The disagreeing inputs from the error estimation process are added to the QBF as additional constraints.

Using this strategy, AppSAT assists in the convergence of the SAT solver when stuck in point-functions.

4. **CycSAT[35]** overcomes the disability of the SAT solver in finding a satisfiable assignment for a QBF when the model of the circuit is not a directed acyclic graph (DAG), such as in the presence of key-controlled feedback cycles. A pre-processing step that identifies key conditions resulting in the creation of these feedback cycles is performed. These conditions, expressed in the Conjunctive Normal Form (CNF), are added to the QBF before the SAT solver is invoked. The added cycle avoidance clauses make the QBF satisfiable, thus enabling the solver in finding a suitable key assignment.

5. **Removal attacks[31]** use structural and functional analysis of the netlist to strip away the SAT-resilience of the circuit. The *Signal Probability Skew* (SPS) attack calculates the probability skew ($P(1) - 0.5$) of each net in the circuit to identify the highly skewed outputs of point-function based SAT-resilient blocks. Once identified, these nets are broken and are set to their respective skew value, thus nullifying the protection offered by the SAT-hard blocks. The *AppSAT Guided Removal* (AGR) attack uses AppSAT to first identify the key bits belonging to the SAT-resilient locking block. Next, a structural analysis starting from these key bits reveals the output of the SAT-resilient block, which is then removed.

6. To attack encrypted sequential circuits, an unrolling based **Sequential SAT attack[7]** has been proposed. In this technique, the attacker unrolls the encrypted design S_e and the unencrypted design S_o for an iteratively increasing number of clock cycles (unr) to obtain equivalent combinational circuits C_e^{unr} and C_o^{unr}. These circuits are decrypted using the combinational SAT attack to obtain a key assignment $\vec{K}_C^{unr}$. The correctness of the obtained key is checked by applying a sufficiently large number of input vectors to S_e (mounted on an FPGA) and comparing the outputs with the outputs of S_o. When a correct key assignment is obtained, the algorithm stops and returns $\vec{K}_C^{unr}$ as the correct key. Note that the combinational SAT attack[20] can be applied directly to sequential circuits if the flip-flops in the design are scan-accessible. However, recent works have shown that the scan-access pin can either be disabled (by burning a fuse) or obfuscated (using additional key gates). In such a scenario, the flip-flops in the oracle IC are no longer scan-accessible for the attacker to launch the combinational SAT attack.

7. A more powerful attack on encrypted sequential circuits, **KC2[18]** combines a traditional sequential SAT attack[4] 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 boast a speedup in decryption times of up to two orders of magnitude in comparison to previous methods[4]. Experimental results have demonstrated the success of KC2 against random XOR/XNOR locking techniques.

Table 1. Logic Encryption Resiliency Matrix

	Sens. [12]	**SAT** [20]	**AppSAT** [16]	**CycSAT** [35]	**Removal** [31]	**SeqSAT** [7]	**KC2** [18]
Random[14]	✗	✗				✗	✗
LCB[11]	✗	✗					
SLL[32]	✓	✗					
KeyDep[6]	✓	✗					
AntiHT[15]	✓	✗					
E2LEMI[1]	✓	✗					
SARLock*[29]	✓	✓	✗		✗		
AntiSAT*[24]	✓	✓	✗		✗/†		
TTLock*[34]	✓	✓	≈		✓		
SFLL*[33]	✓	✓	✓		✓		
Cyclic[17]	✓	✓		✗	✓		
SRCLock[13]	✓	✓		≈	✓		
EFF[5]	✓				✓	✗	?
CBE[7]	†				✓	✗	?
DSE[8]	✓				≈	✓	?

✓ Denotes resilience — † Denotes potential resilience
✗ Denotes susceptibility — ≈ Denotes potential susceptibility
? Denotes unknown — * Denotes compound scheme with SLL

4 Attack Resiliency of Encryption Schemes

In this section we evaluate the resiliency of the logic encryption schemes discussed in this paper against the attack methods from the previous section. Table 1 summarizes the resiliency analysis presented in this section.

1. **EPIC (Random)** - Due to the random key gate placement, EPIC is susceptible to the key sensitization attack and all SAT-based attacks.
2. **LCB** - Although the insertion of key gates on nets with large fan-in and fan-out cones is able to thwart graph equivalence based attacks, this scheme is susceptible to the sensitization attack since no requirement is placed on the interference between the key gates. Additionally the absence of SAT-hard instances makes it vulnerable to the SAT-based attacks.
3. **SLL** - Since the key gate placement strategy in SLL ensures maximum interference between each pair of key gates, the sensitization attack is unable to propagate an incorrect key value to a primary output without knowing the correct assignment on all interfering key gates. Also it is claimed that due to the presence of the AES block, the SAT-based attacks would not be able to determine the in-memory key (AES input) by determining the value at the inputs of the key gates (AES output). However, it has been asserted that AES blocks are highly recognizable and hence prone to removal.
4. **Key Interdependency** - This technique ensures large overlap between inserted key gates, in terms of affected primary outputs, thus thwarting the logic cone based attack[9]. Also, selecting nets with large fault impact values greatly increases the effort in sensitizing a wrong key value to the primary outputs. However, since this technique only dictates the placement strategy

of the key gates, a metric that does not directly affect the complexity of the SAT attack, it is vulnerable to all SAT-based attacks.

5. **Hardware Enlightening** - By improving the probability values of low probability nets, this technique increases the effort in finding suitable locations in the netlist to insert a hardware Trojan. Also, the authors have shown that with sufficiently large key values, a 50% Hamming distance between the correct output value and the incorrect values can be achieved. Absence of SAT-hard instances makes this technique vulnerable to SAT-based attacks.
6. **E2LEMI** - Since each incorrect key activates the LFSR and corrupts the primary outputs, potentially a single DIP can be used to eliminate the entire equivalence class of incorrect keys to obtain the correct key assignment using the SAT attack. Additionally, insertion of MUXes directly on the primary outputs makes this technique prone to a reverse engineering attack that is capable of removing the MUXes and rendering the circuit defenseless.
7. **SARLock** - When used in conjunction with SLL, the resultant compound scheme is able to thwart the sensitization attack as well as the SAT attack. The SAT attack is able to eliminate at most one key per DIP, thus requiring 2^k-1 DIPs to obtain the correct key value, where k is the number of key inputs. Newer attack methods like AppSAT and Removal are able to identify the key bits associated with the point function and can either ensure that the SAT solver does not get stuck solving the point function or can remove the SAT-resilient block.
8. **AntiSAT** - Similar to SARLock, a compound scheme of Anti-SAT and SLL is able to defeat sensitization and SAT attacks while being susceptible to AppSAT and Removal attacks. Improvement to the Anti-SAT architecture[26] using wire entanglement and design withholding has been successful in thwarting the Removal attack.
9. **TTLock** - In addition to being resilient to the SAT attack, the modification of the logic cone ensures that the Removal attack is unable to obtain the original logic cone after removing the point-function. Since the error rate of the SAT-hard instance is low, TTLock is still susceptible to attacks that derive approximate keys, like AppSAT.
10. **SFLL-HD** - Similar to TTLock, removing input patterns from the logic cone increases the resiliency of SFLL-HD against the removal attack, since a corrupted logic cone is obtained after removal of the restore logic. However, increasing the number of PIPs decreases resiliency against the SAT attack which is now able to eliminate more than one key per DIP. Newer attack methods like FALL[19] and Gaussian elimination[28] have been able to defeat SFLL-HD without the use of an oracle.
11. **Cyclic** - Since the resultant circuit cannot be represented as a directed acyclic graph (DAG), the SAT solver is unable to satisfy the given QBF to find a required DIP. However, the pre-processing step in CycSAT is able to identify the key condition that results in cycles and add this condition to the initial QBF. Experimental results have shown CycSAT to be successful against cyclic obfuscation.

12. **SRCLock** - Authors have shown that adding feedback edges between the cycles generated by cyclic obfuscation can lead to an exponential increase in the number of cycles that need to be handled by the CycSAT pre-processing step. Also, the use of an input-dependency based cycle generation technique causes one of the condition checking steps in CycSAT to return an unsatisfiable model. However, experimental results have shown that even after the cyclification of chosen Boolean functions, other condition checking steps in CycSAT are able to defeat SRCLock.
13. **Encrypt Flip-flop** - By restricting controllability/observability of the scan-chain, this technique can defeat the combinational SAT attack. However, the complexity of the unrolling-based Sequential SAT attack does not depend on scan-access to the flip-flops in the design. Therefore it is able to decipher the correct key values within a reasonable amount of time.
14. **Chain Based Encryption** - Due to the limited ability of finding large flip-flop chains with low number of sneak paths, it has been shown[7] that the Chain Based technique is susceptible to the Sequential SAT attack and offers very little additional protection over a random insertion technique.
15. **Deep State Encryption** - The attack time of the Sequential SAT attack increases exponentially with increase in unroll count. By choosing a state with a sufficiently large depth value, this technique can force the attack to unroll for a large number of clock cycles, thereby greatly increasing the time required to find the correct key. However, the proposed structure of the State Detection and Key Detection blocks may make this technique susceptible to the SPS Removal attack. The effectiveness of the KC2 attack on this technique is unknown and needs further investigation.

5 Cost Analysis - Area and Power

Typical IoT devices operate under the restriction of limited resources. Harsh area and power constraints are placed on these devices during the design process to which the added logic encryption blocks must also adhere. Therefore large logic encryption blocks are undesirable. In this section, we evaluate the area and power overhead of the discussed logic encryption techniques.

5.1 Experimental Setup

The combinational logic encryption techniques were implemented on benchmark circuits chosen from the ISCAS '85 suite while the sequential techniques were implemented on ITC '99 benchmarks. The benchmark details are listed in Table 2. These circuits were chosen due to their wide usage in other works on logic encryption. The chosen benchmark circuits were encrypted using the discussed techniques with key sizes of 32, 64, and 128. For larger benchmarks (more than 300 gates), key size 256 was also used. Some of the salient features of the encryption process are listed below.

- **Random, SLL, LCB** - The combinational benchmark circuits for these techniques were obtained from Trust-Hub[2]. Sequential circuits encrypted using the random technique were generated in-house.

Table 2. Benchmark circuits used for evaluation

ISCAS '85 benchmarks			
Circuit	**#PI**	**#PO**	**#Gates**
c432	36	7	160
c499	41	32	202
c880	60	26	383
c1355	41	32	546
c1908	33	25	880
c2670	233	140	1269
c3540	50	22	1669
c5315	178	123	2307
c6288	32	32	2416
c7552	207	108	3513

ITC '99 benchmarks				
Circuit	**#PI**	**#PO**	**#Gates**	**#DFF**
b04	11	8	632	66
b05	1	36	821	34
b07	1	8	380	49
b11	7	6	622	31
b12	5	6	963	121
b13	10	10	310	53
b14	32	54	9767	245
b15	36	70	8367	449
b21	32	22	20027	490
b22	32	22	29162	735

- **Key Interdependency** - Algorithm 1 from [6] was used as the key gate placement strategy. If the number of gate locations found was less than the number of key bits, the remaining locations were chosen randomly.
- **Hardware enlightening** - Key gates were inserted using the technique described in Algorithm 1 of [15]. The threshold values for $PROBAMIN$ and $SLACKMIN$ were chosen to be 0.
- **E2LEMI** - Key controlled MUXes were inserted on all primary inputs while the (sequential) LFSR was omitted, since combinational benchmarks were used for E2LEMI. Instead, a static 'incorrect' selection pattern with equal number of 0s and 1s was stored in the design and applied to the MUXes if the correct key value was not provided. A small hardware Trojan was inserted to detect the application of the correct key.
- **Cyclic** - The light-weight implementation algorithm from [17] was used to insert the key gates into the netlist. A loop length of 8 was chosen and $k/8$ key gates were added for each loop, where k is the number of key bits.
- **SRCLock** - Each micro-cycle generated by the cyclic obfuscation technique was connected with another micro-cycle using a key controlled gate to form the required super-cycle.
- **SARLock, AntiSAT, TTLock** - A compound scheme was used for these techniques. SLL with k=32 was chosen as the traditional scheme while internal signals were used as inputs to the SAT resilient block. The number of internal signals (proportional to the number of SAT resilient keys) was chosen such that the total key sizes were equal to 64, 128, and 256.
- **SFLL** - In addition to the strategy followed for TTLock, all input patterns with a hamming distance of (h=)2 from the applied key input were chosen as the PIPs. The restore logic was designed to detect these $\binom{k}{2}$ PIPs and restore the modified output when the correct key was applied.
- **Encrypt Flip-flop** - The flip-flop selection technique described in Algorithm 2 of [5] was used to obtain suitable key gate locations. For circuits with number of flip-flops less than the chosen key size, the remaining key gates were placed on randomly chosen combinational gate inputs.
- **Chain Based Encryption** - Using the chain-based selection technique described in [7], key gates were inserted at the inputs of the flip-flops with

Table 3. Encryption Scheme Rankings - Cost and Security

Combinational						Sequential		
Technique	**Cost**	**Security**	**Technique**	**Cost**	**Security**	**Technique**	**Cost**	**Security**
Random	A	D	**Cyclic**	B	B	**CBE**	A	C
LCB	A	D	**SRCLock**	B	A	**Random**	B	D
SLL	A	C	**SARLock**	C	B	**EFF**	C	B
KeyDep	B	C	**AntiSAT**	C	B	**DSE**	D	A
AntiHT	B	C	**TTLock**	C	A			
E2LEMI	B	C	**SFLL**	D	A			

maximum distance from the nearest primary output. Again, if the number of flip-flops in the design was less than the key size, randomly chosen combinational gates were encrypted.

- **Deep State Encryption** - The DESIDE-SIM algorithm from [8] was used to determine the deep state. The counter width was set to 5 while the payload was randomly selected from the internal signals in the circuit.

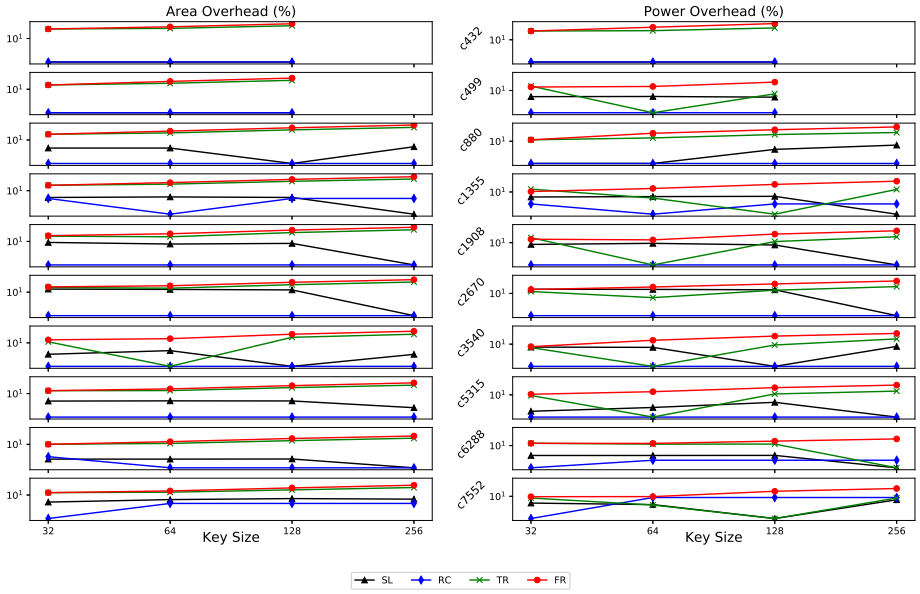
An ATPG tool, Synopsys TetraMax[22], was used to determine test patterns for 100% stuck-at-fault coverage for each unencrypted circuit. In addition to these test patterns, unique random patterns were generated to obtain a total of 10000 input patterns. These patterns were used to simulate each design using the Icarus Verilog simulation tool to derive the switching activity information for each net in the circuit. This switching activity data was fed to a synthesis tool, Synopsys Design Compiler[21], to determine the area and power requirements of each circuit using the SAED90nm_typ library. Note that these area and power values are post logic synthesis but pre phsycial synthesis values. For the purpose of this analysis, the area and power estimates from logic synthesis are used.

5.2 Results

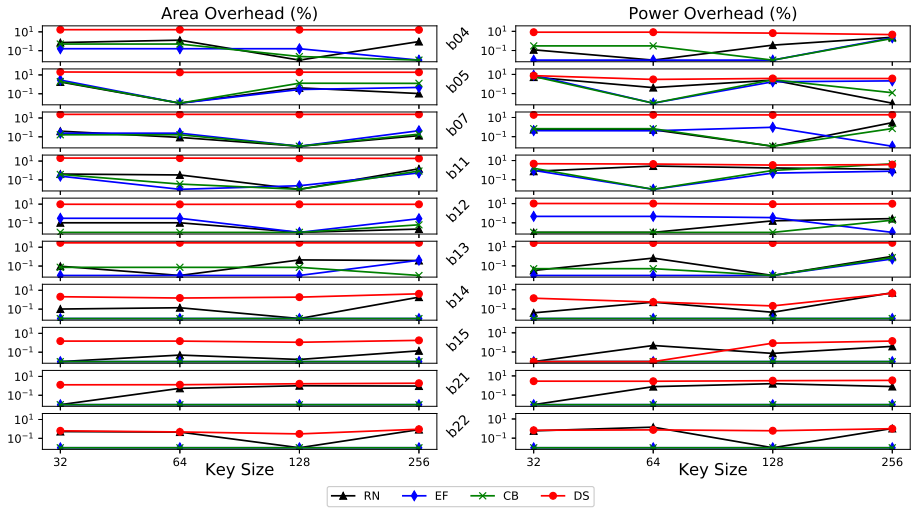
The overhead data from our experimentation was used to determine the cost efficiency of the surveyed techniques. Using the area overhead A_{ib}^k and power overhead P_{ib}^k values, a rank value $R(E_i) = \sum_{\forall b} \sum_{\forall k} A_{ib}^k \cdot P_{ib}^k / |B||K|$ for each encryption scheme E_i applied to benchmark b ($\in B$) for key size k ($\in K$) was calculated, where B is the set of all benchmarks and K is the set of all key sizes. Using these rank values and the resiliency matrix from Section 4, the encryption techniques were graded, in terms of cost and security efficiency, as shown in Table 3 (lower letter grade is better). For each unique cost efficiency grade (A-D), the scheme that offers the best security is chosen and the cost overhead values for these schemes are plotted in Figure 7[1]. Our observations regarding the obtained cost metrics are listed below.

- The area and power overheads for traditional (SAT-vulnerable) techniques are negligible. This follows from the fact that the number of key gates added by these techniques is linear to the number of key bits. This value stays relatively low compared to the size of the circuit. It can be postulated that

[1] The raw data used to generate these plots is made available online at https://github.com/kasarayv/costanalysis



(a) Combinational Logic Encryption



(b) Sequential Logic Encryption

Fig. 7. Cost overhead analysis

for large key sizes (k>512), the overhead introduced by the key gates may become significant for small designs.

- In many benchmark circuits, an increase in key size led to a decrease in cost for several encryption schemes. This can be attributed to the optimization techniques used by the synthesis tool to achieve a better fit. This leads to

an important conclusion that, unlike attack times, area and power overhead values may not follow a direct correlation with key size.
- Cyclic locking techniques exhibit low cost overhead values because the number of key gates added by these techniques increases linearly with key size, similar to traditional schemes.
- The combinational point-function based SAT-resilient techniques introduce a large number of gates into the design, thus contributing significantly to the cost overheads. An average 200% cost overhead is observed for smaller benchmarks; this decreases to 50% for larger benchmarks.
- SARLock and AntiSAT have relatively lower overhead values in comparison to TTLock and SFLL. The reason for this is the modification of the original logic cone in the latter two techniques, which in many cases adds additional gates to the circuit. Also, on average the overhead for SFLL is larger than that of TTLock due to SFLL's complex restoration block.
- Similar to corresponding combinational techniques, sequential techniques that are SAT-vulnerable also exhibit low area and power overheads due to the addition of low number of key gates to the design.
- The cost of using the sequential SAT-resilient technique is high due to the introduction of two detection blocks in addition to the corruption and restoration gates.

In summary, SAT-vulnerable techniques exhibit low overhead (high efficiency in cost) whereas SAT-resilient schemes incur higher cost due to the insertion of SAT-hard blocks. Cyclic locking techniques exhibit low cost overhead while offering high level of security. Techniques like delay locking [25] and parametric locking [34] that alter other properties of the design, instead of corrupting signals, have shown success in thwarting SAT-based attacks. However, these methods need to be evaluated for area and power requirements.

6 Conclusion

In this work, we have surveyed several logic encryption and decryption techniques. Benchmark circuits encrypted using these schemes are utilized to derive an estimate of the cost incurred (area occupied and power consumed) by these techniques. Traditional logic encryption methods are light-weight with negligible overhead, but are susceptible to SAT-based attacks. Successful SAT-resilient schemes introduce large overheads and are therefore undesirable for applications where resources are constrained. Although sequential logic encryption techniques exhibit relatively lower cost overheads, they are still attackable by newly proposed decryption methods. Therefore, a new light weight robust logic encryption scheme for application in IoT device security is desired.

References

1. Alasad, Q., Bi, Y., Yuan, J.S.: E2LEMI: Energy-efficient logic encryption using multiplexer insertion. Electronics **6**(1), 16 (2017)

2. Amir, S., Shakya, B., Xu, X., Jin, Y., Bhunia, S., Tehranipoor, M., Forte, D.: Development and evaluation of hardware obfuscation benchmarks. Journal of Hardware and Systems Security **2**(2), 142–161 (2018)
3. Chakraborty, R.S., Bhunia, S.: HARPOON: an obfuscation-based soc design methodology for hardware protection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **28**(10), 1493–1502 (2009)
4. El Massad, M., Garg, S., Tripunitara, M.: Reverse engineering camouflaged sequential circuits without scan access. In: 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). pp. 33–40. IEEE (2017)
5. Karmakar, R., Chatopadhyay, S., Kapur, R.: Encrypt Flip-Flop: A novel logic encryption technique for sequential circuits. arXiv preprint arXiv:1801.04961 (2018)
6. Karmakar, R., Kumar, H., Chattopadhyay, S.: On finding suitable key-gate locations in logic encryption. In: 2018 IEEE International Symposium on Circuits and Systems (ISCAS). pp. 1–5. IEEE (2018)
7. Kasarabada, Y., Chen, S., Vemuri, R.: On SAT-based attacks on encrypted sequential logic circuits. In: 20th International Symposium on Quality Electronic Design (ISQED). pp. 204–211. IEEE (2019)
8. Kasarabada, Y., Thulasi Raman, S.R., Vemuri, R.: Deep state encryption for sequential logic circuits. In: 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). pp. 338–343 (July 2019)
9. Lee, Y.W., Touba, N.A.: Improving logic obfuscation via logic cone analysis. In: 2015 16th Latin-American Test Symposium (LATS). pp. 1–6. IEEE (2015)
10. Li, M., Shamsi, K., Meade, T., Zhao, Z., Yu, B., Jin, Y., Pan, D.Z.: Provably secure camouflaging strategy for IC protection. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2017)
11. Narasimhan, S., Chakraborty, R.S., Chakraborty, S.: Hardware ip protection during evaluation using embedded sequential trojan. IEEE Design & Test of Computers **29**(3), 70–79 (2012)
12. Rajendran, J., Pino, Y., Sinanoglu, O., Karri, R.: Security analysis of logic obfuscation. In: Proceedings of the 49th Annual Design Automation Conference. pp. 83–89. ACM (2012)
13. Roshanisefat, S., Mardani Kamali, H., Sasan, A.: SRClock: Sat-resistant cyclic logic locking for protecting the hardware. In: Proceedings of the 2018 on Great Lakes Symposium on VLSI. pp. 153–158. ACM (2018)
14. Roy, J.A., Koushanfar, F., Markov, I.L.: Ending piracy of integrated circuits. Computer **43**(10), 30–38 (2010)
15. Samimi, M.S., Aerabi, E., Kazemi, Z., Fazeli, M., Patooghy, A.: Hardware enlightening: No where to hide your hardware trojans! In: 2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS). pp. 251–256. IEEE (2016)
16. Shamsi, K., Li, M., Meade, T., Zhao, Z., Pan, D.Z., Jin, Y.: AppSAT: Approximately deobfuscating integrated circuits. In: Hardware Oriented Security and Trust (HOST), 2017 IEEE International Symposium on. pp. 95–100. IEEE (2017)
17. Shamsi, K., Li, M., Meade, T., Zhao, Z., Pan, D.Z., Jin, Y.: Cyclic obfuscation for creating SAT-unresolvable circuits. In: Proceedings of the on Great Lakes Symposium on VLSI 2017. pp. 173–178. ACM (2017)
18. Shamsi, K., Li, M., Pan, D.Z., Jin, Y.: KC2: Key-condition crunching for fast sequential circuit deobfuscation. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 534–539. IEEE (2019)

19. Sirone, D., Subramanyan, P.: Functional analysis attacks on logic locking. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 936–939. IEEE (2019)
20. Subramanyan, P., Ray, S., Malik, S.: Evaluating the security of logic encryption algorithms. In: Hardware Oriented Security and Trust (HOST), 2015 IEEE International Symposium on. pp. 137–143. IEEE (2015)
21. Synopsys: Design compiler. https://www.synopsys.com/ (2018)
22. Synopsys: Tetramax. https://www.synopsys.com/ (2018)
23. Xiao, K., Forte, D., Jin, Y., Karri, R., Bhunia, S., Tehranipoor, M.: Hardware trojans: Lessons learned after one decade of research. ACM Transactions on Design Automation of Electronic Systems (TODAES) **22**(1), 6 (2016)
24. Xie, Y., Srivastava, A.: Mitigating SAT attack on logic locking. In: International Conference on Cryptographic Hardware and Embedded Systems. pp. 127–146. Springer (2016)
25. Xie, Y., Srivastava, A.: Delay locking: Security enhancement of logic locking against ic counterfeiting and overproduction. In: Proceedings of the 54th Annual Design Automation Conference 2017. p. 9. ACM (2017)
26. Xie, Y., Srivastava, A.: Anti-SAT: Mitigating SAT attack on logic locking. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **38**(2), 199–207 (2018)
27. Xu, X., Shakya, B., Tehranipoor, M.M., Forte, D.: Novel bypass attack and bdd-based tradeoff analysis against all known logic locking attacks. In: International Conference on Cryptographic Hardware and Embedded Systems. pp. 189–210. Springer (2017)
28. Yang, F., Tang, M., Sinanoglu, O.: Stripped functionality logic locking with hamming distance based restore unit (sfll-hd)–unlocked. IEEE Transactions on Information Forensics and Security (2019)
29. Yasin, M., Mazumdar, B., Rajendran, J.J., Sinanoglu, O.: SARLock: SAT attack resistant logic locking. In: Hardware Oriented Security and Trust (HOST), 2016 IEEE International Symposium on. pp. 236–241. IEEE (2016)
30. Yasin, M., Mazumdar, B., Sinanoglu, O., Rajendran, J.: CamoPerturb: Secure IC camouflaging for minterm protection. In: Computer-Aided Design (ICCAD), 2016 IEEE/ACM International Conference on. pp. 1–8. IEEE (2016)
31. Yasin, M., Mazumdar, B., Sinanoglu, O., Rajendran, J.: Removal attacks on logic locking and camouflaging techniques. IEEE Transactions on Emerging Topics in Computing pp. 1–1 (2017)
32. Yasin, M., Rajendran, J.J., Sinanoglu, O., Karri, R.: On improving the security of logic locking. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **35**(9), 1411–1424 (2015)
33. Yasin, M., Sengupta, A., Nabeel, M.T., Ashraf, M., Rajendran, J.J., Sinanoglu, O.: Provably-secure logic locking: From theory to practice. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1601–1618. ACM (2017)
34. Yasin, M., Sengupta, A., Schafer, B.C., Makris, Y., Sinanoglu, O., Rajendran, J.J.: What to lock?: Functional and parametric locking. In: Proceedings of the on Great Lakes Symposium on VLSI 2017. pp. 351–356. ACM (2017)
35. Zhou, H., Jiang, R., Kong, S.: CycSAT: SAT-based attack on cyclic logic encryptions. In: Proceedings of the 36th International Conference on Computer-Aided Design. pp. 49–56. IEEE Press (2017)