SAT-based Key Determination Attack for Improving the Quality Assessment of Logic Locking Mechanisms

Marcel Merten[1] Mohammed E. Djeridane[1]
Sebastian Huhn[1,2] Rolf Drechsler[1,2]

[1] University of Bremen, Germany
`{mar_mer,djeridam,huhn,drechsle}`
`@informatik.uni-bremen.de`
[2] Cyber-Physical Systems, DFKI GmbH
28359 Bremen, Germany

Abstract. Today, the manufacturing of *Integrated Circuits* (ICs) is distributed over various foundries, resulting in untrustworthy supply chains. Therefore, major concerns regarding the security, privacy, and reliability of the fabricated ICs exist. Logic locking is one widely disseminated protection technique against malicious intentions. Recently, the emerging technology of *Reconfigurable Field-Effect Transistors* (RFETs) has been utilized to implement logic obfuscation based on polymorphic logic gates to protect intellectual property. However, the mechanisms' assessment is required to reinforce the newly introduced logic locking and, hence, to tackle the growing security threats. Previous research like [1] proposes a promising approach on determining the protection quality of such a logic obfuscation by assessing the security threat of incorrect keys. However, the number of assessable keys is limited by the computation time during the assessment. Therefore, it is essential to assess the keys, forming the most critical security breach for the logic locking mechanism. This work proposes a novel approach on determining the most intimidating incorrect keys for improving the assessment quality of arbitrary logic locking mechanisms. In particular, based on the concept of a SAT-based attack, a procedure for determining the most safety-critical keys is developed. The experimental evaluation proves that the proposed key determination procedure unveils weaknesses of the protection mechanism that remain undetected when using existing techniques and, hence, clearly outperforms any other existing key determination procedures.

Keywords: logic obfuscation, camouflaging, IP protection, formal methods, SAT-based attacks, pseudo-Boolean optimization

⋆ This work was financially supported by the German Federal Ministry of Education and Research BMBF under the framework of VE-CirroStrato and the AI initiative of the Free Hanseatic City of Bremen. We would like to thank Verific Design Automation Inc. for providing the SystemVerilog frontend used for the implementation of our technique.

1 Introduction

Nowadays, designers can benefit from access to advanced technology nodes without having the large capital expenditure of operating their own semiconductor foundries. This is thanks to the distributed manufacturing of the *Integrated Circuits* (ICs). However, such a distribution also yields a growing threat of compromising the integrity of once trusted IC processes by unauthorized or untrusted users [2]. During the last decade, *Complementary Metal-Oxide-Semiconductor* (CMOS)-based protection mechanisms have been the dominant technology for implementing various protection measures. However, the layout-level obfuscation by using CMOS-based camouflaging causes a significant overhead with respect to the resulting power consumption and the required area [3].

Recent research works like [2, 4, 5] have been focusing on achieving high protection while still preserving low overhead by utilizing reconfigurable silicon nanowire field-effect transistor-based polymorphic logic gates [2]. In [2], an algorithm is proposed that replaces gates impacting the original functional behavior of the circuit by reconfigurable polymorphic logic gates. Afterward, the quality of the resulting logic locking functionality is assessed by a metric based on the *Hamming Distance* (HD) of the outputs over randomly applied stimuli and keys. The result is considered optimal if the HD is 50% of the maximal HD. The formal approach proposed in [1] shows the limitations of simulation-based approaches, unveiling further weaknesses in the protection mechanisms. In [1], a limited number of corrupting keys is calculated, which later be assessed. Thereby, a corrupting key is defined as a key that behaves equivalent to the correct key when considering at least one stimulus.

This work proposes a novel technique to determine corrupting keys, forming the most critical security breaches. More precisely, a framework is designed to calculate the most intimidating corrupting keys based on the concept of a SAT-based attack. In contrast to other techniques, the proposed approach calculates the corrupting keys based on *Distinguishing Input Patterns* (DIPs), maximizing the number of equivalent behaving stimuli. This improves the quality of the assessment of potential security threats using logic locking mechanisms.

Various experiments have been conducted on the ITC'99 benchmark set. The results prove that the improved key calculation unveils weaknesses in the protection structures that remain undetected when using current approaches. The proposed technique utilizes the concept of a SAT-based attack to provide a metric for evaluating the threat of a specific corrupting key. In conclusion, the proposed approach allows a more accurate evaluation of the security of a logic locked circuit.

The remainder of this work is structured as follows: Section II briefly introduces the preliminaries of this work. Section III describes the proposed key determination procedure in detail. Finally, Section IV presents the experimental evaluation. A conclusion and an outlook on future work are given in Section V.

2 Preliminaries

Within the last decade, a lot of research has been spent on enhancing electronic systems, while the classical CMOS technology has exceeded its physical limits. Research in the field of reconfigurable technologies has gained a lot of interest since it has a great potential to realize even more complex systems. This emerging technology is a promising candidate for overcoming the constraints of Moore's law by employing polymorphic logic gates.

2.1 Reconfigurable Field-Effect Transistors

Different concepts have been proposed on realizing a device-level reconfiguration capability like RFETs. An RFET introduces a control gate that can be configured between an n-channel and p-channel behavior [3]. The reconfiguration capabilities of this new emerging technology can be used to implement new protection mechanisms, e.g., an on-chip key storage by the polymorphic logic behavior [3]. Furthermore, the RFET technology is promising to introduce effective protection mechanisms against optical reverse-engineering attacks.

A popular approach to prevent reverse engineering, even given the entire layout, is adding logic locking mechanisms to the circuit. The correct functional behavior of a circuit C is defined in Def. 1.

Definition 1. *Given a circuit C, a set of appliable stimuli $\mathcal{S}$, a set of reachable states $\mathcal{F}$ and a set of possible outputs $\mathcal{P}$, the function $C : \mathcal{S} \times \mathcal{F} \rightarrow \mathcal{P}$ defines the intended functional behavior of C. In particular, $C(s, \psi)$ describes the functional behavior $\forall C, s \in \mathcal{S}, \psi \in \mathcal{F}$, with $s \in \mathcal{S}$ be a stimuli and $\psi \in \mathcal{F}$ be an internal state.*

Logic locking encrypts the correct functional behavior by encrypting the circuit C using a secret key k_c. The functional behavior of a logic locked circuit is defined in Def. 2.

Definition 2. *The functional behavior of a logic locked circuit C is defined given a stimuli $s \in \mathcal{S}$, an internal state $\psi \in \mathcal{F}$ and a key $k \in \mathcal{K}$. Applying the secret key k_c yields the correct functional behavior $C(s, \psi, k_c) = C(s, \psi)$.*

CMOS-based approaches usually introduce XOR/XNOR key gates [6–8] or MUX gates [9–12] to obfuscate the correct functional behavior of the circuit, resulting in a huge overhead regarding the area- and power-consumption [2]. Figure 1 gives a basic example of an XOR gate inverting the behavior of the preceding logic when an incorrect key is applied. In the example, the locked output has the functionality of a NAND gate instead of the intended AND gate behavior.

Polymorphic logic gates like RFETs realize multiple functionalities in the same cell and, hence, are an effective way to implement a logic locking mechanism. The intended functionality is chosen by configuring a control signal. To insert key gates without the high-performance overhead of CMOS- based techniques, polymorphic

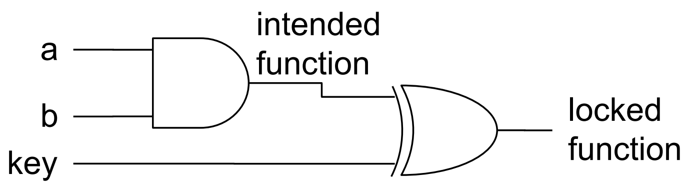
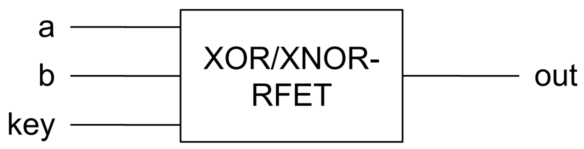


Figure 1: Simple CMOS-based logic locking example



XOR			
a	b	key	out
0	0	0	0
0	1	0	1
1	0	0	1
1	1	0	0

XNOR			
a	b	key	out
0	0	1	1
0	1	1	0
1	0	1	0
1	1	1	1

Figure 2: XOR/XNOR-RFET

logic gates can replace gates of the original circuit. Meaning gates with a high impact on the primary outputs are replaced [2]. Various functionalities can be implemented by RFET-based cells like the NAND/NOR- or the XOR/XNOR-RFET. An example of the RFET visualized in Figure 2, can be configured as an XOR or XNOR gate, depending on the control signal serving as a key bit.

2.2 Boolean Satsifiability Problem

The *Boolean Satisfiability* (SAT) problem is one of the first proven NP-hard problems [13]. However, a lot of research on SAT-solving techniques has significantly increased the effectiveness of SAT-solvers over the years. For example, *Dynamic Clause Activation* (DCA) allows to activate or deactivate a CNF Φ with an activation literal a in an extended $\Phi_a = \Phi + a$. To satisfy Φ_a, Φ only has to be satisfied if $a = 0$. Therefore, Φ can be deactivated by setting $a = 1$. The application of DCA in SAT can result in significant speed-up [14].

Moreover, some modern SAT-solvers are extended to solve *Pseudo Boolean* (PB) logic. PB allows defining inequalities in clauses. Furthermore, weights can be assigned to specific literals or clauses that are not required to be true in a satisfiable solution. *Pseudo Boolean Optimization* (PBO) can be used to determine a solution maximizing or minimizing the weights of a PB instance. The optimization is performed utilizing an objective function Θ. The function Θ is usually defined as a maximization or minimization of a sum of weighted literals. These PBO-based optimization techniques have been heavily orchestrated in various domains like IC testing [15].

2.3 SAT-based Attacks

While camouflaging and logic obfuscation try to protect intellectual property from malicious misuse, attackers constantly work on techniques to remove or unlock such protection mechanisms. A popular attacking algorithm is the SAT-attack first proposed in [16]. The idea of the SAT-based attack is to use SAT to unlock the circuit by determining the correct key k_c or an equivalent behaving key. First, a miter structure of two instances of the logic locked circuit is created. By solving the miter instance, a pair of keys (k_1, k_2) and a DIP D is calculated for the *Primary Inputs* (PIs). The DIP D is an input pattern, which results in a differing output behavior using k_1 and k_2, meaning that at least one of the output behaviors of the two compared keys is incorrect. Next, an unlocked product C of the chip is used to get the correct output behavior $C(D)$ for the D. Before the next DIP D' is calculated, the key space of k_1 and k_2 is constrained to satisfy the correct output behavior $C(D)$ for the previously calculated DIP D. This is done by adding a SAT-instance Φ_D consisting of two inverted miters. Each inverted miter forces equivalence between the logic locked circuit using $keyX$ and the oracle output $C(D)$ on the stimuli D. The basic principle of the SAT-based attack is illustrated in Figure 3.

2.4 Quality Assessment of RFET-based Logic Locking Mechanisms using Formal Methods

This section describes the SAT-based quality assessment approach proposed in [1]. The assessment framework analyzes a *Circuit-under-Assessment* (CuA) using (RFET-based) logic locking mechanisms. First, corrupting keys - incorrect keys that result in correct functional behavior given at least one stimulus - are collected for a later assessment. A formal definition of a corrupting key k_f is given in Definition 3.

Definition 3. *Given a logic locked circuit C, a stimuli $s \in \mathcal{S}$ and an internal state $\psi \in \mathcal{F}$, a key $k_f \in \mathcal{K}$ is a corrupting key, iff $k_f \neq k_c$ and $\exists s, C(s, \psi, k_c) = C(s, \psi, k_f)$.*

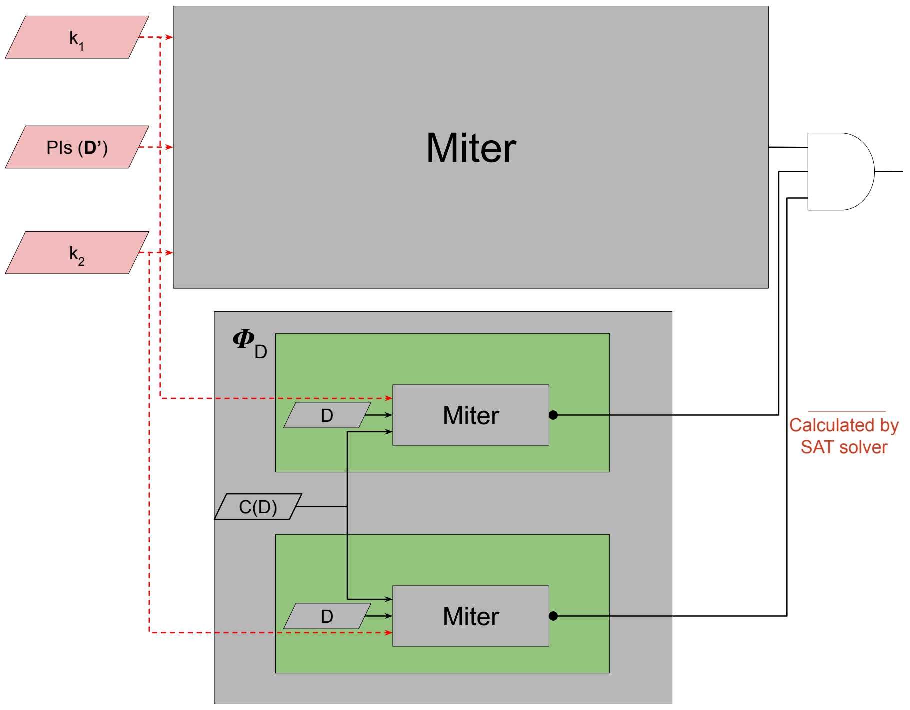


Figure 3: Basic concept of a SAT-based attack

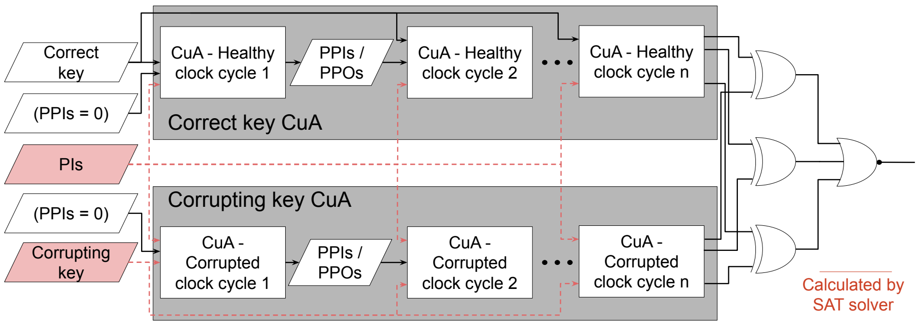

Correct key
(PPIs = 0)
PIs
(PPIs = 0)
Corrupting key
CuA - Healthy clock cycle 1
PPIs / PPOs
CuA - Healthy clock cycle 2
CuA - Healthy clock cycle n
Correct key CuA
Corrupting key CuA
CuA - Corrupted clock cycle 1
PPIs / PPOs
CuA - Corrupted clock cycle 2
CuA - Corrupted clock cycle n
Calculated by SAT solver

Figure 4: Abstract model of the miter structure

Therefore, a miter circuit is created from the *CuA* considering the correct key k_c - yielding the SAT instance Φ_{k_c} - and any incorrect key $\widehat{\mathcal{K}}$ yielding $\Phi_{\widehat{\mathcal{K}}}$. The basic principle of this construction is given in Figure 4. The CuA is unrolled for N clock cycles to consider sequential elements. The FFs are modeled as *Pseudo Primary Inputs* (PPIs), initialized with 0.

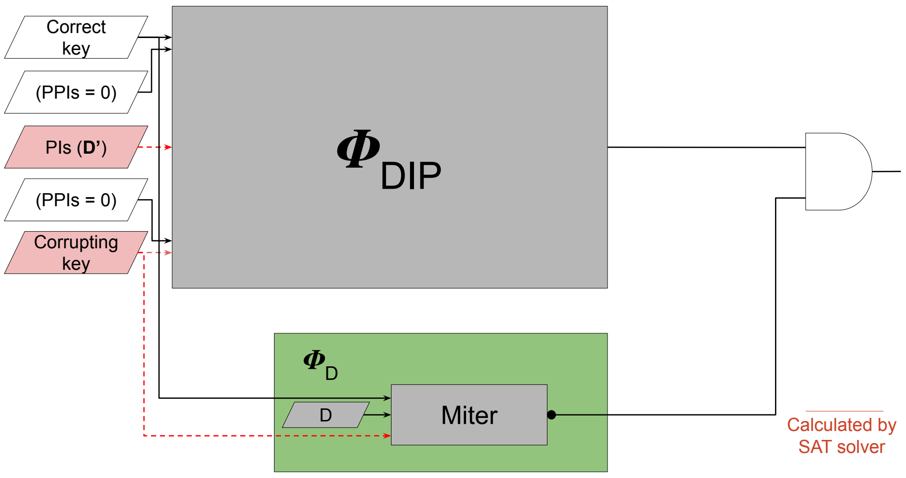


Figure 5: Adapted SAT-based attack to collect constraints for the key-space

The entire model is stored as one SAT instance Φ_{comp} and processed by a state-of-the-art SAT solver. The inverted miter compares the unrolled Φ_{k_c} with the unrolled $\Phi_{\widehat{\mathcal{K}}}$, i.e., considering any incorrect key $k_i \neq k_c, k_i \in \widehat{\mathcal{K}}$. In particular, both states – defined by the stored FFs' values – and the primary output values can be compared for all N observed clock cycles. If a satisfiable solution is determined, i.e., a corrupting key k_f has been detected, this circumstance results in a functional equivalent of the CuA given at least one stimulus.

Next, the calculated security threat is assessed. More precisely, every determined corrupting key k_f is evaluated against possible stimuli leading to functional equivalence to the correct key k_c. More precisely, the individual corrupting key is enforced in $\Phi_{\widehat{\mathcal{K}}}$ by additional clauses. The key detection procedure – including the security threat evaluation regarding the discovered corrupting key – is repeated until Φ_{comp} is unsatisfiable or a user-defined limit has been exceeded.

3 SAT-based Key Determination Attack

This section describes the approximative determination of the most intimidating key to improve the quality assessment of a CuA using logic locking mechanisms. The key determination procedure is divided into two parts. First, an adapted SAT-based attack is applied to collect constraints for the keys. In the second step, DCA methods are combined with the constraints to calculate a key, which forms a maximal threat to the logic locking mechanism.

To collect the constraints narrowing the key-space, a miter SAT instance Φ_{DIP} is generated to calculate DIPs. Subsequently, the miter structure is constructed from the CuA while considering the a-priori known *correct* key k_c yielding

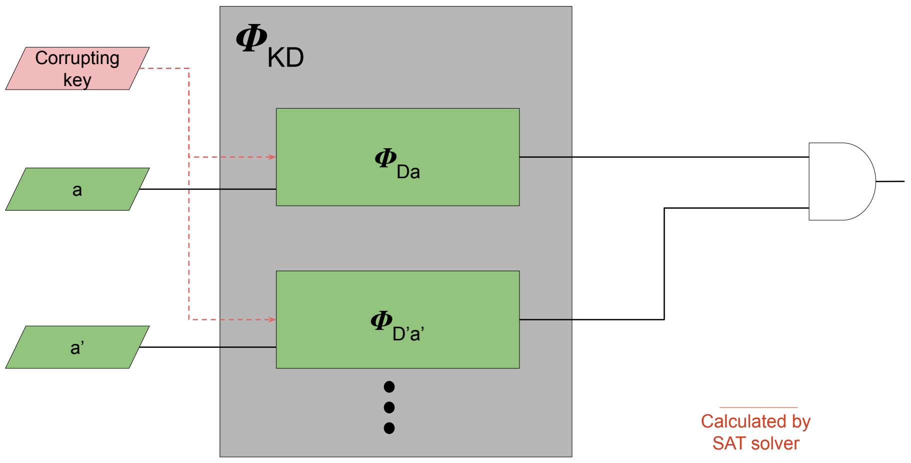


Figure 6: Complete key determination instance including all activatable key-space constraints

the SAT instance Φ_{k_c} and any incorrect key in $\widehat{\mathcal{K}}$ yielding $\Phi_{\widehat{\mathcal{K}}}$. The FFs are modeled as *Pseudo Primary Inputs* (PPIs) in cycle $n+1$ and are connected to the corresponding *Pseudo Primary Outputs* (PPOs) of the previous cycle n. Furthermore, similar to the SAT-based approach, the primary inputs use the same stimuli for both unrolled instances (of the CuA) and are kept constant during the unrolling. Contrary to the SAT-based assessment framework, in the attack framework, a miter is constructed to detect functional inequivalence. After the miter has been added, the key is constrained for both instances of the unrolled CuA. For Φ_{k_c} , the correct key k_c is set by adding clauses implying k_c, whereby $\Phi_{\widehat{\mathcal{K}}}$ is extended by a conflict clause excluding k_c. The entire model is stored as one SAT instance Φ_{DIP} and processed by a state-of-the-art SAT solver.

Like in the SAT-based assessment framework, the CuA is unrolled for N clock cycles since for the assessment of sequential circuits, sequential elements – meaning *Flip-Flops* (FFs) – have to be considered [17]. Here, the value N has to be adjusted for the CuA characteristics. The number of clock cycles required to achieve the relaxation given a stimuli of the circuit varies depending on the circuit. Similar to the approach proposed in [1], 0 is assumed as the initialization value for all FFs in cycle $n = 1$.

Next, a DIP $D \in \mathcal{D}$ is calculated, distinguishing the behavior of an arbitrary key from k_c. Similar to a SAT-based attack, a constraint is modeled as instance Φ_D of the circuit that forces the equivalence to the correct key on this specific DIP. Only one inverted miter instance is required since the correct key k_c is given. Next, a new $D' \in \mathcal{D}$ can be calculated. Like in a SAT-based attack, this procedure is repeated to narrow the search space for the keys until every remaining key results in a functional equivalent behavior (as yielded when the correct key is being applied). The algorithm terminates after the calculation of all constraints $\Phi_X, X \in \mathcal{D}$, meaning that $\Phi_D, \Phi_{D'}, \Phi_{D''}, ...$ constrain the corrupting key to fully unlock the circuits. The basic principle of this adapted SAT-based attack is visualized in Figure 5.

Afterward, DCA is used to add a new activation variable $a \in \mathcal{A}$ for Φ_D, such that $\Phi_{Da} = \Phi_D + a$. Next, Φ_{Da} is added to the final key determination problem instance Φ_{KD}, so that $\Phi_{kd} = \Phi_{kd} * \Phi_{Da}$. By assuming $a = 0$, $\Phi_{Da} = \Phi_D$ and, hence equivalence to the correct key on this specific DIP is forced. Now, the next DIP D' can be calculated. In Figure 6 a complete key determination instance $\forall D, D' \in \mathcal{D}$ and $\forall a, a' \in \mathcal{A}$ is illustrated.

Once Φ_{KD} is complete, containing all the activatable Φ_{Da}, the most intimidating key is determined. First, the weight $w(a) = -1$ is assigned for every activation signal $a \in A$. PBO-solving techniques are utilized to determine the most intimidating key. In particular, an objective-function θ, defined in Equation 1, is used to maximize the number of activated instances Φ_D. Therefore, the PBO-solver increases the functional equivalence to the correct behavior on the calculated DIPs.

$$\theta = max(\sum_{a \in A}(w(a)) \tag{1}$$

A corrupting key is calculated by solving the problem instance $\Phi_{KD} * \theta$ that satisfies the functional equivalence to k_c on the maximum number of DIPs. The DIPs of a SAT-attack are iteratively narrowing the search space of the keys to find a key that unlocks the circuit. Therefore, a key is considered a most intimidating corrupting key that satisfies the maximum number of constraints as given by the DIPs.Consequently, $\Phi_{KD} * \theta$ is solved for a predefined number of keys, which will be assessed afterward.

The assessment of the detected keys can be performed with an arbitrary assessment technique, for example, the HD-approach or the approach proposed in [1].

4 Experimental Evaluation

This section describes the experimental evaluation of the proposed technique to determine corrupting keys during the quality assessment of logic locking mechanisms. The experiments compare the novel approach with the determination of corrupting keys proposed in [1], which are used as the baseline. For the assessment of the detected keys, the assessment framework defined in [1] is used for both key determination approaches.

All experiments have been executed on an AMD 4750U processor with 32 GB system memory. All implementations are solely in C++. For the evaluation, different benchmark circuits of the ITC'99 benchmark suite are considered. For each of these circuits, 15 of the *NOR*, *NAND*, *XOR* and *XNOR* gates have been randomly replaced by RFETs, while the functional behavior is retained

Table 1: Results - 15 Key-bits

	SAT-attack-based approach					**Baseline approach [1]**			
circuit	DIPs	$\#\{k_c\}$	#stimuli			$\#\{k_c\}$	#stimuli		
			minimum	average	maximum		minimum	average	maximum
b05	0	-	-	-	-	1,024	2	2	2
b06	2	1,024	4	4	4	1,024	2	3	4
b07	1	3	2	2	2	3	2	2	2
b08	2	63	256	256	256	63	256	256	256
b09	1	1	2	2	2	1	2	2	2
b10	2	1,024	1,024	1,024	1,024	1,024	256	256	512
b11	1	1,024	128	128	128	1,024	126	126	128
b12	1	1,024	32	32	32	1,024	32	32	32
b13	1	1,024	1,024	1,024	1,024	1,024	512	768	1024
b14	2	1,024	1,024	1,024	1,024	1,024	1,024	1,024	1,024
b15	2	1,024	1,024	1,024	1,024	1,024	1,024	1,024	1,024
b17	3	1,024	1,024	1,024	1,024	1,024	1,024	1,024	1,024
b20	1	0	0	0	0	0	0	0	0
b21	1	0	0	0	0	0	0	0	0
b22	1	0	0	0	0	0	0	0	0

if the correct key is applied. Experimental evaluations have shown that 15 RFETs can be considered a sufficient number of key gates to create logic locking structures with weaknesses that are non-trivial to analyze and, hence, hard to detect. Consequently, each circuit has 15 control signals resulting in $2^{15} = 32,768$ possible keys. Similar to the results in [1], the 1,024 most intimidating keys are assessed on both the proposed and the baseline approach. Furthermore, up to 1,024 stimuli with functionally correct behaving *Primary Outputs* (POs) (per corrupting key k_f) are captured.

The FFs of the CuA are initialized with 0, and the stimuli are kept constant over all five clock cycles. Each circuit has been unrolled for five clock cycles since it has been proven as an appropriate parameter to cover the functional behavior's majority of the considered benchmark circuits [18].

Table 1 shows the detailed results of the two approaches for determining the corrupting keys. It illustrates the number of detected corrupting keys, their minimum, the average and maximum number of corrupted stimuli for the novel SAT-based key determination approach and the baseline approach proposed in [1]. Furthermore, the number of calculated DIPs for the SAT-attack-based approach are shown.

For the b05, b07, b08, b09, b12, b14 and b15, the results are equivalent regarding the number of detected corrupting keys and corrupted stimuli. However, in the case of the b05, the proposed approach shows that no DIP can be calculated, meaning all 32,768 keys are behaving equivalent. This provides additional information about the poor quality of the underlying logic locking mechanism. Considering the circuits: b06, b10, b11, and b13, the novel approach shows that there are more critical corrupting keys than those ones as detected by the baseline approach. In

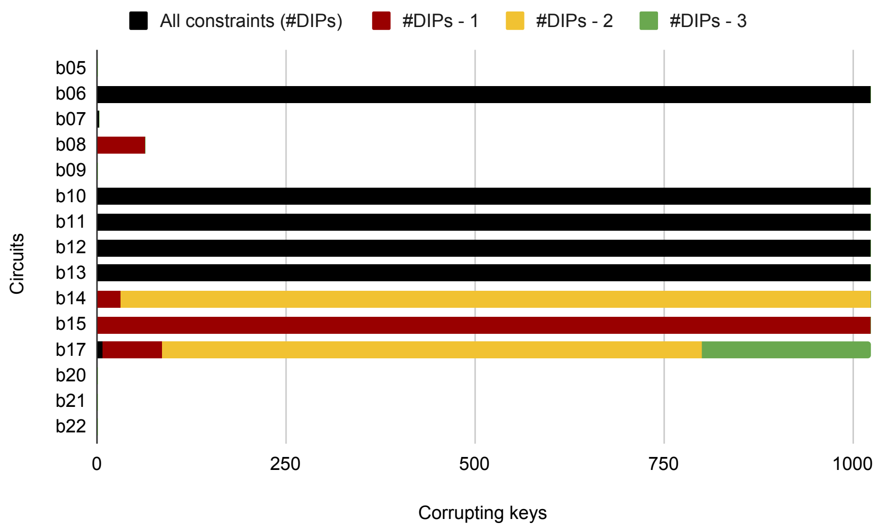


Figure 7: Number of DIPs with correct functional behavior for corrupting keys

fact, the baseline key collection algorithm can lead to a major misjudgment of the quality of a logic locking mechanism.

Figure 7 presents the number of activated constraints for the corrupting keys when using the SAT-based key determination technique. The diagram shares further information about the actual equivalence of the corrupting keys to the correct behavior. For example, in the case of the b17, seven corrupting keys are able to fully unlock the circuit, while 80 corrupting keys can satisfy two constraints and 712 corrupting keys are able to satisfy one constraint. In the case of the b10, all 1,024 assessed keys are able to satisfy the equivalent behavior to the correct key on both calculated DIPs. Therefore, at least 1,024 corrupting keys exist that fully unlock the circuit's functional behavior resulting in an unbearable security breach. On the other hand, the results for the b14 and b15 show that no corrupting key fully unlocking the circuit's functional behavior exists.

This clearly shows that the novel approach outperforms other approaches of determining the most intimidating corrupting keys, providing a more appropriate quality assessment of logic locking mechanisms.

5 Conclusions

This paper presented a novel method of calculating the most intimidating corrupting keys for logic locking mechanisms. In the end, the proposed technique allows determining keys, which form an enormous security threat, by adapting the conceptual structure of SAT-based attacks and enhancing the idea with

PBO techniques. The resulting metric ensures the detection of potential security breaches and outperforms the existing key determination mechanisms. Future work will investigate a sophisticated weight calculation for the activation signals to prefer activating instances Φ_D, implying that the most equivalent functional behavior is achieved.

References

1. M. Merten, S. Huhn, and R. Drechsler, "Quality Assessment of RFET-based Logic Locking Protection Mechanisms using Formal Methods," in *IEEE European Test Conference (ETS)*, 2022, pp. 1–2.
2. Q. Alasad, J.-S. Yuan, and Y. Bi, "Logic locking using hybrid CMOS and emerging SiNW FETs," *Electronics*, vol. 6, no. 3, 2017.
3. S. Rai, S. Srinivasa, P. Cadareanu, X. Yin, X. S. Hu, P.-E. Gaillardon, V. Narayanan, and A. Kumar, "Emerging reconfigurable nanotechnologies: Can they support future electronics?" in *IEEE/ACM International Conference on CAD*, 2018, pp. 1–8.
4. Q. Alasad and J. Yuan, "Logic obfuscation against IC reverse engineering attacks using PLGs," in *IEEE International Conference on Computer Design*, 2017, pp. 341–344.
5. Q. Alasad, J.-S. Yuan, and P. Subramanyan, "Strong logic obfuscation with low overhead against IC reverse engineering attacks," *IEEE Transaction on CAD of Integrated Circuits and Systems*, vol. 25, no. 4, pp. 1–31, 2020.
6. J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending piracy of integrated circuits," in *Design, Automation and Test in Europe*, 2008, pp. 1069—1074.
7. J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Design Automation Conference*, 2012, pp. 83–89.
8. J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015.
9. Q. Alasad, Y. Bi, and J.-S. Yuan, "E_2LEMI: Energy-efficient logic encryption using multiplexer insertion," *Electronics*, vol. 6, pp. 1–16, 02 2017.
10. J. B. Wendt and M. Potkonjak, "Hardware obfuscation using PUF-based logic," in *IEEE/ACM International Conference on CAD*, 2014, pp. 270–271.
11. S. M. Plaza and I. L. Markov, "Solving the third-shift problem in IC piracy with test-aware logic locking," *IEEE Transaction on CAD of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 961–971, 2015.
12. Y. Lee and N. Touba, "Improving logic obfuscation via logic cone analysis," 05 2015.
13. S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the Third Annual ACM Symposium on Theory of Computing.* Association for Computing Machinery, 1971, p. 151–158.
14. S. Eggersgluss and R. Drechsler, "Increasing Robustness of SAT-based Delay Test Generation Using Efficient Dynamic Learning Techniques," in *IEEE European Test Symposium*, 2009, pp. 81–86.
15. S. Huhn, S. Eggersglüß, K. Chakrabarty, and R. Drechsler, "Optimization of retargeting for IEEE 1149.1 TAP controllers with embedded compression," in *Design, Automation and Test in Europe Conference and Exhibition, 2017*, 2017, pp. 578–583.
16. P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *IEEE International Symposium on Hardware Oriented Security and Trust*, 2015, pp. 137–143.

17. R. Arora and M. Hsiao, “Enhancing SAT-based bounded model checking using sequential logic implications,” in *International Conference on VLSI Design*, 2004, pp. 784–787.
18. A. Finder, A. Sülflow, and G. Fey, “Latency analysis for sequential circuits,” in *IEEE European Test Symposium*, 2011, pp. 129–134.