Quantifying Security and Overheads for Obfuscation of Integrated Circuits

Vivek Venugopalan, Gaurav Kolhe, Andrew Schmidt, Joshua Monson, Matthew French
University of Southern California,
Information Sciences Institute
{vivekv, gkolhe, aschmidt, jmonson, mfrench}@isi.edu

Yinghua Hu, Peter A. Beerel, Pierluigi Nuzzo
University of Southern California,
Ming Hsieh Department of Electrical and Computer Engineering
{yinghuah, pabeerel, nuzzo}@usc.edu

***Abstract*—Logic obfuscation techniques are used to deter intellectual property piracy, reverse engineering, and counterfeiting threats in the manufacturing of integrated circuits (IC). The security of these obfuscation algorithms has been, however, compromised by Boolean satisfiability (SAT) based attacks. SAT attacks can reveal the deobfuscation key in seconds, rendering the IC design vulnerable to reverse engineering. The ever-changing landscape of attacks and defenses are typically vetted on small benchmark circuits where security is measured in terms of the time required to recover the encryption key from the obfuscated circuit. This paper introduces a uniform security metric for evaluating the existing obfuscation methods. The benchmark circuits are synthesized after each obfuscation method to determine the overhead in terms of area, power, and timing, including the impact of logic obfuscation algorithms on practical circuits with reasonable gate count (>100K gates). A thorough evaluation is conducted to determine the contributing factors toward attack resiliency time such as gate count, logic depth, and (area, power, timing) overhead before recommending the best obfuscation method for a specific circuit.**

***Keywords*—logic encryption; obfuscation; reverse engineering**

I. Introduction

The semiconductor industry manufactures integrated circuits (IC) in offshore foundries to ensure profitability and offset capital costs associated with the operation of a dedicated in-house foundry. This leads to IC design, validation, and integration by in-house fabless design houses or third party vendors and finally fabrication by the contract foundry. In this process, the IC vendor has limited visibility into the manufacturing process by the contract foundry leading to increased concerns of IC counterfeiting, piracy, and unauthorized over-production. In addition to financial losses due to intellectual property (IP) piracy, national security is also potentially affected by counterfeited ICs, hampering mission critical operations [1].

Logic obfuscation techniques consist of additional logic added to the existing IC design along with the new inputs known as key inputs. The correct key unlocks the IC, producing correct output values for valid input patterns. The attack model consists of methods to determine the correct encryption key so that the obfuscated design can be easily reverse engineered.

Assumptions: Logic obfuscation methods have been rendered vulnerable based on assumptions resulting in various threat models. In this paper, it is assumed that the attacker has access to the obfuscated IC layout/gate-level netlist and an activated IC to to obtain correct input-output pairs.

Motivation and Contributions: Logic obfuscation methods have been studied in detail; however, each technique utilizes a different security metric, the techniques are only studied on trivially sized circuits, and the overheads in terms of size, power, and performance impact are rarely measured. This work attempts to unify the obfuscation security metric in terms that are meaningful to an end user: attack resiliency time. Attack resiliency time is the time required to recover the encryption key from an obfuscated circuit when exposed to an attack. Today, most obfuscation techniques are implemented on the ISCAS benchmark circuits. These circuits are categorized as combinational/sequential and are used to evaluate the effectiveness of the attack and defense models. However, ISCAS circuits are generally small in terms of gate/transistor count and do not accurately represent an actual IC netlist. This work extends the security and overhead analysis to include more complex circuits from the Common Evaluation Platform (CEP) [2]. Finally, the overhead due to the additional area, power, and delay needed to encrypt a circuit is reported along with the attack resiliency time to enable trade-off evaluation between cost and robustness of each obfuscation method.

II. Background & Proposed Approach

Several methods have been proposed over the last decade for logic obfuscation, or logic locking, of ICs, addressing different vulnerabilities. Roy et al. [3] introduced *random obfuscation* (`RN`), which inserts XOR and XNOR key-gates randomly in the netlist such that the circuit functionality can only be retrieved upon applying the correct key pattern to these gates. *Strong logic locking* [4], denoted by `DAC12` in this paper, uses a method to insert XOR/XNOR gates in appropriate locations, determined via an *interference graph*, to achieve resilience to fault-analysis based attacks, that is, to make it harder for the attacker to identify the correct value of a key bit in isolation, without knowing the rest of the key bits. *Fault analysis-based logic locking* (`FLL`) [5] inserts XOR/XNOR gates in a efficient way to maximize the

Hamming distance (HD) between the two values obtained at the output ports when applying the correct key and a random wrong key, respectively. Alternatively, multiplexers may be inserted instead of XOR/XNOR gates, a variant denoted by `FLL with MUX`.

`SARLock` [6] is a technique that is resilient to SAT attacks. `SARLock` is based on the premise that a SAT attack would require a number of iterations (and queries to the SAT solver) that grows exponentially with the size of the key in order to unlock the circuit. However, SAT-attack resilient techniques are usually based on single-point functions, hence they tend to corrupt only one output bit for each input pattern and expose for most of the time the correct output functionality.

Finally, Dupuis et al. [7] (`IOLTS`) propose to insert AND or OR gates to minimize low-controllability nodes in the IC, i.e., nodes whose values are very rarely set to 1 or 0, which makes them amenable to the insertion of hardware Trojans that can easily hide, i.e., are never triggered, during IC testing. All of these methods were originally proposed for combinational circuits, but can be extended to sequential circuits by regarding the input and output ports of each register as part of the cell primary outputs and primary inputs, respectively [5]. All these obfuscation methods use some or a combination of metrics such as output HD, key recovery time, or code coverage to evaluate their resiliency from key recovery attacks [8]. A single uniform set of metrics is absent in the evaluation of these obfuscation methods. This paper focuses on SAT attack, since it is deemed as being the most efficient to date [9], and has been reported as capable of thwarting all of the above obfuscation methods [10].

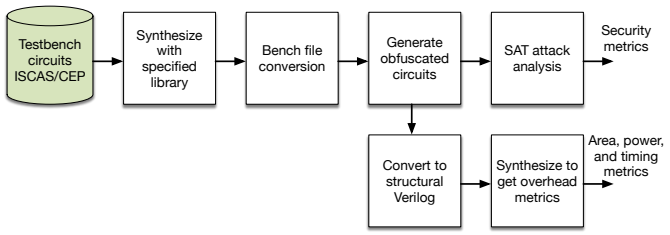


Fig. 1. Obfuscation Analysis Flow

A *testbench database* consisting of ISCAS and CEP circuits are obfuscated using several different methods and then subjected to an attack simulator to recover the encryption key as shown in Figure 1. The attack simulator module (currently SAT attack) can be updated as new attacks are discovered. In addition to the attack resiliency time, overhead metrics are also captured from the obfuscated circuits by utilizing standard ASIC EDA tools, and comparing the original unobfuscated design against the obfuscated design.

III. Optimization-based Obfuscation

The obfuscation requirements specified by a user can be represented as mixed integer linear constraints, a formalism that is expressive enough to capture topological properties of circuits as well as overhead constraints on the estimated area, power, and performance. A new optimization-based obfuscation method is proposed – *constraint-driven fault-analysis based logic locking* (`CDFLL`). `CDFLL` encodes the user requirements as well as constraints from fault-impact analysis into mixed integer linear constraints. It then solves an optimization problem to directly determine the key gate locations that maximize HD subject to an upper bound on the gate count. In `CDFLL`, the mixed integer linear programming formulation introduces a set of binary variables, one for each of the input ports, output ports, and gate outputs of the original netlist. Each binary variable evaluates to 1 if and only if a key gate is added to the corresponding location. Additional constraints enforce a set of rules that help optimize the average HD of the obfuscated netlist. For instance, to guarantee a lower bound on the average HD, a constraint is added to prescribe that at least one node in the fan-in cone of each output port be selected for key gate insertion. Similarly, to minimize masking effects between key gates, it is required that, if a node is selected for key gate insertion, then its immediate fan-out nodes should not be selected. Similarly, `CDFLL` can detect and avoid instances of multiple key gates connected in cascade (the so-called runs of key gates [11]) which are essentially equivalent to one key gate and therefore less effective for obfuscation. Since `CDFLL` is constraint-based obfuscation, it enables mixing with other obfuscation methods such as `SARLock` to increase the attack resiliency time.

IV. Results

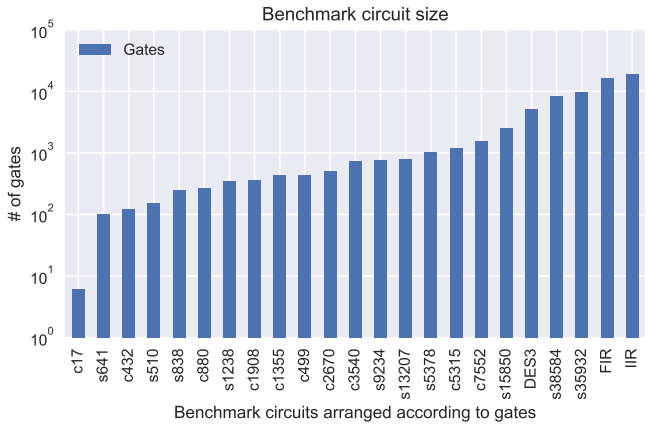


Fig. 2. Benchmark circuits considered for obfuscation.

The security metrics are evaluated alongside overhead performance metrics so an end user can make the appropriate decisions for their particular use case. These experiments were conducted on a *test-bench database* consisting of ISCAS and CEP circuits that are obfuscated using different methods explained in Sections II and III, and then subjected to an attack simulator to recover the keys. The obfuscation overhead is also evaluated by using standard ASIC EDA tools, and by comparing the original designs against the obfuscated designs. In this paper, the designs are synthesized using a 65-nm CMOS10LPE technology to extract area, power, and timing estimates. Attacks were allowed to run to completion, with no arbitrary limitations. There are however, a handful of data

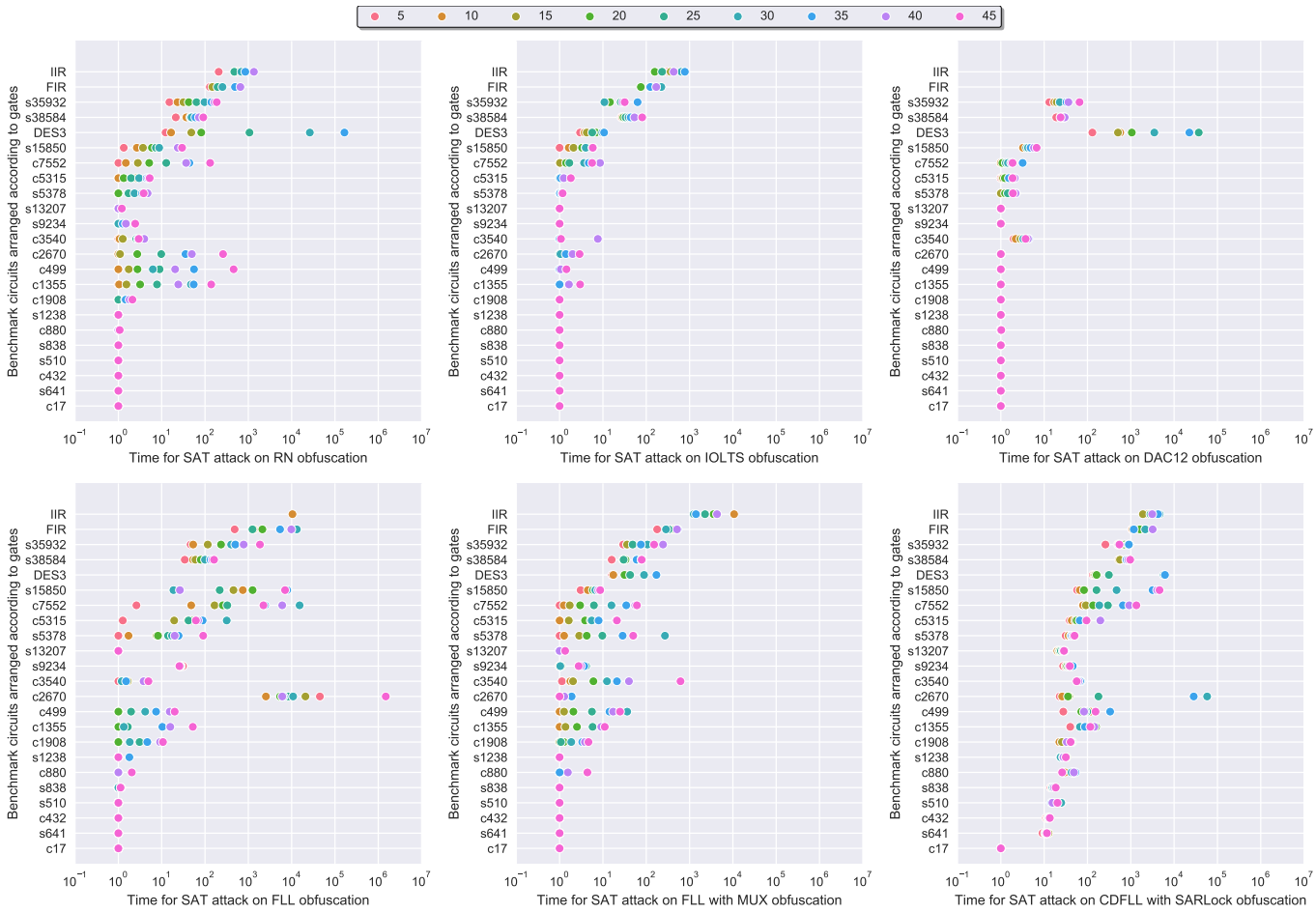


Fig. 3. Attack resiliency time for benchmark circuits – 6 methods

points that were unable to be completed due to excessive run times of > 14 days that are discussed below.

The database consists of 23 circuits, as shown in Figure 2, of which 20 ISCAS circuits (10 combinational and 10 sequential circuits) are selected based on their common use in the literature. In addition, three sequential circuit CEP cores (DES3, FIR, and IIR) representative of cryptography and signal processing cores commonly found in System on a Chip (SoC) designs, are included in the study to add diversity and complexity to the database.

To generate baseline measurements, each circuit is obfuscated using the following methods: (i) `RN`, (ii) `IOLTS`, (iii) `DAC12`, (iv) `FLL`, (v) `FLL with MUX`, and (vi) `CDFLL` with `SARLock`. The baseline experiments are designed for a combination of configuration parameters, including: (i) combinational or sequential circuit, (ii) obfuscation method, (iii) obfuscation coverage (obfuscation %), which correlates with the key size, and (iv) size of the circuit in terms of gate count. Each circuit is obfuscated based on a coverage parameter (percentage of gates introduced by the obfuscation method with respect to the original gate count) ranging from 5% to 45%, which results in 9 obfuscated circuits for each method. `CDFLL` with `SARLock` is a hybrid method, where `CDFLL` method uses the obfuscation % steps from (5-45) and `SARLock` method has the key size fixed to 10 bits, to help reduce the dimensionality of the data. Therefore, a total of 1,242 circuits (23 benchmark circuits $\times$ 6 methods $\times$ 9 obfuscation percentages) are evaluated.

Figure 3 reports the attack resiliency time for all 6 methods considered in this paper and their coverage levels. The y-axis lists benchmark circuits arranged according to increasing gate count, while the x-axis shows the attack resiliency time in logarithmic scale. Before delving into analysis, it is important to denote the following outliers where data was not able to be collected in time for publication. Interestingly, the attack on the `FLL` obfuscation of CEP DES3 did not complete after 14 days. Similarly, the SAT attack on `FLL` obfuscation of CEP IIR completed on 10% obfuscation, but has not completed for any higher level in 14 days. These attack times are, therefore, not reported in the figure, but should be noted as a very strong results. It should also be noted that the interference graph generation step when applying `DAC12` obfuscation did not complete for CEP FIR and IIR after 20 days. Since no obfuscated circuit could be produced in these cases, no attacks

could be performed. All of these experiments are ongoing, but subject to uptime of the HPC system.

Inspection of Figure 3 reveals that is is difficult to pinpoint a specific trend based on the attack resiliency time for each obfuscation method, however some trends can be seen. Some of the most important observations are as follows: (i) the size and type of the circuit (i.e. combinational or sequential) does not always correlate well with the attack resiliency time, (ii) the dynamic range, or responsiveness of a circuit to a given obfuscation type as the coverage increases, can vary widely. (iii) the performance trend of a given obfuscation method on the ISCAS circuits, is not necessarily a good predictor of performance on the CEP circuits. All of these observations highlight the inherent complexities of analyzing SAT-based algorithms and the reliance of several obfuscation methods on specific structures within the reference structure and why it is important to evaluate the obfuscation performance on the desired reference circuit.

CEP DES3 provides detailed insights into how other methods perform as well, where its `RN` implementation showed higher attack resiliency time. The `DAC12` implementation of the CEP DES3 core shows comparable resiliency when compared to its `RN` implementation. Although the CEP DES3 core is a sequential circuit, it shows better attack resiliency as compared to some of the sequential ISCAS benchmarks with a higher gate count.

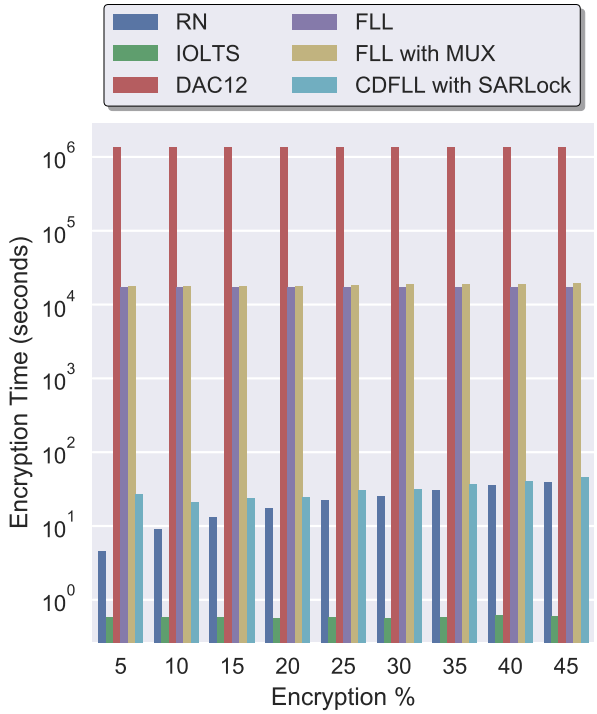
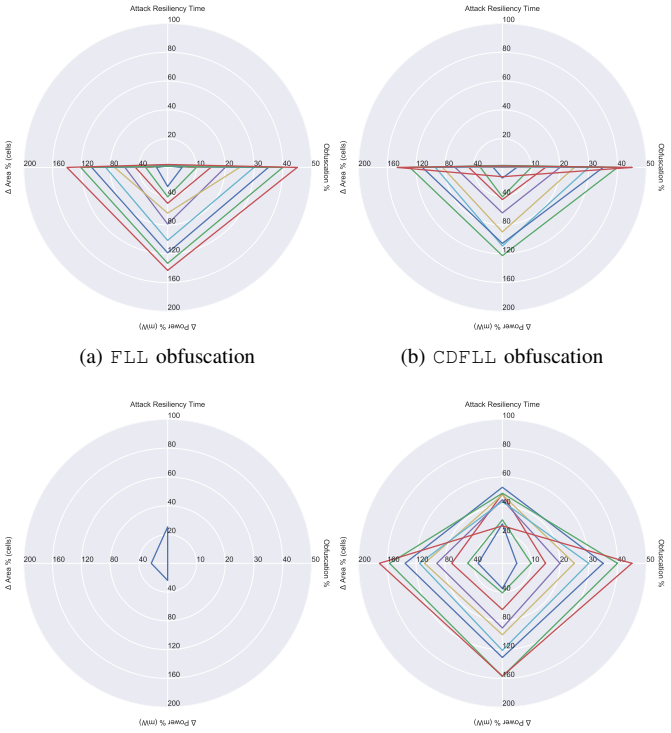


Fig. 4. Encryption time for CEP DES3 for all 6 methods

`FLL` exhibits a wider dynamic range for the attack resiliency time, while `RN` replicates the dynamic range for some of the circuits. The `FLL with MUX` is derived from the `FLL` method, but the attack resiliency time values are clustered closely. The hybrid strategy using `CDFLL` with `SARLock` shows that the attack resiliency time of all the circuits shifted to the right indicating a marginal improvement in the attack resiliency for the larger benchmark circuits.

Although it is important to evaluate circuits in terms of its attack resiliency time, obfuscation implementation time is important too. Figure 4 shows the time taken to generate an obfuscated circuit. The CEP DES3 core with $5,000$ gates took 14 days to generate the interference graph and hence the `DAC12` obfuscation takes the maximum time. The `FLL` and `FLL with MUX` obfuscation generation takes an average of 5 hours, whereas the remaining three methods take less than a minute.



(a) `FLL` obfuscation (b) `CDFLL` obfuscation

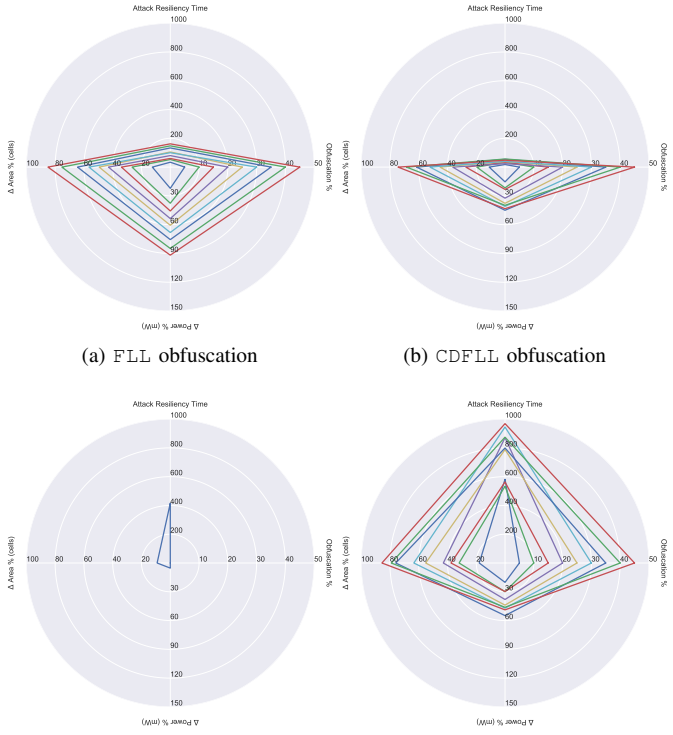
(c) `SARLock` obfuscation (d) `CDFLL` with `SARLock` obfuscation

Fig. 5. Spider plots comparing `FLL`, `CDFLL`, `SARLock`, and `CDFLL` with `SARLock` – ISCAS C880

In addition to the attack resiliency time, it is necessary to measure and track the overhead of obfuscation methods. This is useful in evaluating the overall effectiveness of the constraint-driven obfuscation (`CDFLL`) compared to other approaches. Figure 5 and Figure 6 provide an illustrative drill down on two of the 23 circuits in the database comparing four different methods: (i) `FLL`, (ii) `CDFLL`, (iii) `SARLock` with 10-bit key, and (iv) hybrid `CDFLL` with 10-bit `SARLock`. Figure 5 shows spider plots for the ISCAS C880 circuit, where the axes represent the obfuscation %, Δ Power %, Δ Area %, and the attack resiliency time. The timing overhead is also calculated for these circuits and is not plotted in the spider chart as the obfuscated cicruits meet the timing requirements.

For the ISCAS C880 circuit, the `FLL` and `CDFLL` methods

show comparable attack resiliency time with the CDFLL having marginal improvement in power overheads. SARLock with 10-bit key increases the attack resiliency time as compared to the baseline FLL. The hybrid CDFLL with 10-bit SARLock provides the best implementation with the highest attack resiliency time (50×) and does not increase the power overheads compared to the FLL implementation.



(a) FLL obfuscation (b) CDFLL obfuscation

(c) SARLock obfuscation (d) CDFLL with SARLock obfuscation

Fig. 6. Spider plots comparing FLL, CDFLL, SARLock, and CDFLL with SARLock – ISCAS S38584

Similarly, for the sequential circuit S38584, the hybrid method produces 5× higher attack resiliency time than FLL as shown in Figure 6. It is also observed for both the ISCAS circuits that the hybrid method increases the attack resiliency time while maintaining the power overheads close to the original FLL method.

V. Conclusions & Future Work

For the first time, logic obfuscation is evaluated by synthesizing the benchmark circuits into netlists and then quantifying the overhead metrics in terms of area, power, and timing, using the same security metric on multiple obfuscation types. Furthermore, realistic circuits with higher gate count (>100K gates) are also used for evaluating these obfuscation techniques. For the ISCAS benchmark circuits, the hybrid method consisting of CDFLL with SARLock shows an increased attack resiliency time (5-50×) with the same power overhead when compared to its baseline FLL implementation. For the larger and realistic CEP circuits such as DES3, FLL provides best resiliency followed by RN and DAC12. In conclusion, the performance of obfuscation methods can vary widely based on the input circuit. The overhead can provide additional insight into the obfuscation performance within the specification envelope an end user may have to operate within.

Acknowledgement

This material is based on research sponsored by the Air Force Research Labs (AFRL) and the Defense Advanced Research Projects Agency (DARPA) under agreement number FA8560-18-1-7817. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Labs (AFRL), the Defense Advanced Research Projects Agency (DARPA), or the U.S. Government.

References

[1] SIA Anti-Counterfeiting Task Force, "Winning The Battle Against Counterfeit Semiconductor Products," *Semiconductor Industry Association White Paper*, pp. 4–8, 2013.

[2] MIT Lincoln Laboratory. (2018) Common Evaluation Platform. [Online]. Available: {https://github.com/mit-ll/CEP}

[3] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending Piracy of Integrated Circuits," *Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[4] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security Analysis Of Logic Obfuscation," in *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012, pp. 83–89.

[5] 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.

[6] M. Yasin, B. Mazumdar, J. J. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *Hardware Oriented Security and Trust (HOST), 2016 IEEE International Symposium on*. IEEE, May 2016, pp. 236–241.

[7] S. Dupuis, P.-S. Ba, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "A Novel Hardware Logic Encryption Technique For Thwarting Illegal Overproduction And Hardware Trojans," in *On-Line Testing Symposium (IOLTS), 2014 IEEE 20th International*. IEEE, 2014, pp. 49–54.

[8] Q. Yu, J. Dofe, Z. Zhang, and S. Kramer, *Hardware Obfuscation Methods for Hardware Trojan Prevention and Detection*. Springer International Publishing, 2018, ch. 12, pp. 291–325. [Online]. Available: https://doi.org/10.1007/978-3-319-68511-3_12

[9] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "Circuit obfuscation and oracle-guided attacks: Who can prevail?" in *Proceedings of the on Great Lakes Symposium on VLSI 2017*. ACM, 2017, pp. 357–362.

[10] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2015, pp. 137–143.

[11] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2016.