SAL: Function Search Attack on Logic Locked Circuits

Yuqiao Zhang*, Pinchen Cui†, Ziqi Zhou* and Ujjwal Guin*
*Dept. of Electrical and Computer Engineering, Auburn University
†Dept. of Computer Science and Software Engineering, Auburn University
Email: {yuqiao.zhang, pinchen, ziqi.zhou, ujjwal.guin}@auburn.edu

***Abstract*—Logic locking becomes an effective way of enabling trust in the outsourced IC design and manufacturing processes, where a design is obfuscated by inserting a lock. A chip will function properly, when it is activated by programming with a secret key. Over the years, researchers have proposed different logic locking mechanisms to prevent Boolean satisfiability (SAT)-based attacks, and successfully preserve the security of a locked design. However, an untrusted foundry, as an adversary, can use many different techniques to break the security of logic locking. In this poster paper, we present a novel attack that relies on identifying repeated functions for determining the secret key. The proposed attack is based on self referencing and can estimate the secret key in few minutes. It does not require any data from an unlocked chip to launch this attack and eliminates the need for an oracle. This proposed attack can fortunately be prevented by inserting keys in all repeated function instances in the circuit.**

I. Introduction

Due to the globalization of the design and manufacturing of integrated circuits (IC), different threats have been emerged in recent years. Logic locking has gained popularity primarily to address the threats from untrusted manufacturing [1], [2]. In logic locking, a circuit is obfuscated and the complete functionality will only be recovered when a secret key is programmed into the chip [3], [4]. Different logic locking techniques have been proposed over the years, and they are - XOR-based, MUX-based, LUT-based, and state space based locking. However, XOR-based logic locking becomes popular due to its simplicity.

Even though the wide popularity of SAT attacks introduced by Subramanyan et al. [5] among the research community, these attacks may have practical limitations for determining the secret key of a locked netlist. As the attacks require an oracle to discard equivalent classes of incorrect keys, an adversary must possess a working chip in order to launch the attack. An untrusted foundry, cannot unlock the netlist, when it receives the layout information (GDSII or OASIS files) from the SoC design house, and needs to wait for an unlocked chip to be available in the market. This can be challenging as many of the chips used in DoD or other critical applications are highly unlikely to circulate (unless it is a commercial-off-the-shelf, COTS part) in the market. Second, it is yet to be demonstrated that SAT-attacks can be launched in industrial designs. The attack even fails to estimate the correct key for a small benchmark circuit (e.g., *c6288*, see the details in [5]). In this poster paper, we assess the security from a completely different perspective. *Can we determine the secret key by simply analyzing the circuit netlist?* Here, we present a novel oracle-less attack to break logic locking.

II. Proposed Function Search Attack

The basic idea for launching SAL attack is based on finding repeated internal functionality of a circuit. Generally, the Boolean functions are not unique in a circuit and repeated multiple times to implement the overall circuit functionality. Several functions (denoted as unit functions or *UFs*) are repeated in an arithmetic logic unit (ALU) of a processor, adders, multipliers, advance encryption standards (AES) and RivestShamirAdleman (RSA) cryptosystems, and many other digital circuits. If any of such repeated *UFs* are not locked during the logic locking process, all the locked functions will be unlocked by simply comparing them with their unlocked version. Thus, the key will be determined.

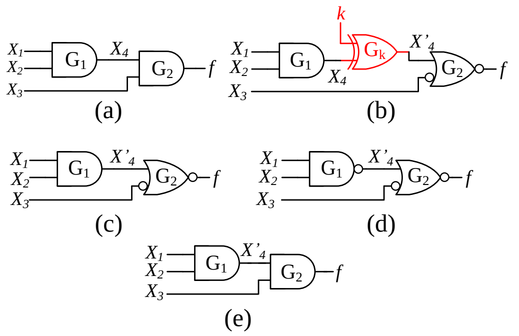

Figure 1: *EUFs* for different hypothesis keys. (a) Original netlist. (b) Locked netlist with $k = 1$. (c) *EUF* with hypothesis key $k_h = 0$. (d) *EUF* with $k_h = 1$. (e) *EUF* with $k_h = 1$ based on DeMorgan's Theorem.

Figure 1 provides an example to construct equivalent *UF*, which is used to launch our proposed *SAL*. The proposed attack can be lauched in three stages. First, the equivalent unit functions *EUF*s are constructed based on the hypothesis key value. Second, we propose an efficient *EUF* search algorithm in the locked netlist. Since the structure of a circuit can be transformed and represented by a directed graph, an *EUF* could be searched in the complete graph reconstructed from the circuit. We propose to use a Depth-First-Search (DFS)-based algorithm to find a match. Finally, the key bits are determined when a match is found.

Table I: Success Rate (*SR*) and Misprediction rate (*MR*) metric.

Benchmark	**# Total Gates**	**Success Rate (%)**			**Misprediction Rate (%)**		
		Min.	**Avg.**	**Max.**	**Min.**	**Avg.**	**Max.**
c3540	1669	75.2	79.6	87.5	0.0	1.8	3.1
c5315	2307	81.3	87.8	94.5	0.0	1.2	3.9
c6288	2406	98.4	98.5	99.2	0.0	0.1	0.1
c7552	3512	69.5	79.9	88.3	0.0	2.0	4.7
b14	3461	85.2	93.4	97.7	0.0	0.5	3.1
b15	6931	89.8	95.7	98.4	0.0	0.5	1.6
b20	7741	92.9	96.4	99.2	0.0	0.3	1.6
b21	7931	89.1	94.6	98.4	0.0	0.4	1.6
b22	12128	92.9	95.6	98.4	0.0	0.4	1.6
b17	21191	92.2	95.6	99.2	0.0	0.5	3.1

The complexity of *SAL* itself is linear to the key size and is $O(|K|)$ as all the inserted key gates are analyzed individually. As each *EUF* will be searched in the netlist based on our proposed algorithm, the actual overall complexity will be $O(|K| * n * u)$ where n and u represent the size of the netlist and average size of the *UFs*, respectively. For a particular circuit, the size of netlist n is fixed, and the size of *UF* can range from few layers.

As the SAL attack relies on self referencing, it can be prevented if insertion of the keys is carried out in such a way that the equivalent function search always returns null. In other words, the attack will fail if we choose to place a key gate in a unique *UF* in the netlist or lock all of the *UFs* simultaneously. We propose an automated key insertion algorithm that performs *UF* search in the netlist before placing a key gate into the circuit. The same DFS-based search algorithm, used for the topological attack, can be used as well.

III. Simulation Results

The effectiveness of our proposed *SAL* attack is evaluated using ISCAS'85 and ITC'99 benchmark circuits. Table I shows the *SR* and *MR* of SAL attacks on different benchmark circuits. Each benchmark circuit is randomly inserted with 128 key gates for 100 instances. Here, the *SR* and *MR* present the ratio of successful predictions and incorrect predictions to estimate the complete key. The minimum, average, and maximum *SR* are shown in Columns 3, 4, and 5. Columns 6, 7, and 8 provide the minimum, average, and maximum *MR*.

Figure 2 shows the histogram plots of *SR* metric for four different benchmark circuits. Figure 2.(a) shows the distribution of *SR* for c5315 benchmark circuit. Note that the overall variance of *SR* is decreased when increasing the size of the circuit, which indicates that *SAL* can be more effective in industrial designs. The histogram plots of *MR* for the same benchmark circuits are illustrated in Figure 3. We observe an exponential distribution with a mean (λ^{-1}) and variance (λ^{-2}) for each circuit. The overall mean and variances are decreased when increasing the size of the circuit, which makes our proposed *SAL* more accurate for lager designs.

IV. Conclusion

In this poster paper, we present a novel SAL attack that uses function search to defeat existing logic locking techniques. As the unit functions are generally instantiated multiple times, the attacker can recover the correct key value by using the hypothesis key if one of these repeated functions is locked. The simulation result shows that we can accurately determine majority of the key values, and the complexity of *SAL* is linear to key size. A solution against SAL attack is also presented, where we propose to lock all the repeated unit functions.

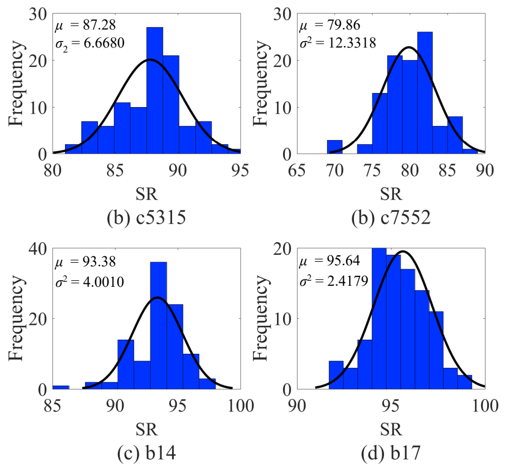

Figure 2: Histogram plots of the *SR* for different benchmark circuits with 128 key bits: (a) *c5315* (b) *c7552* (c) *b14* (d) *b17*

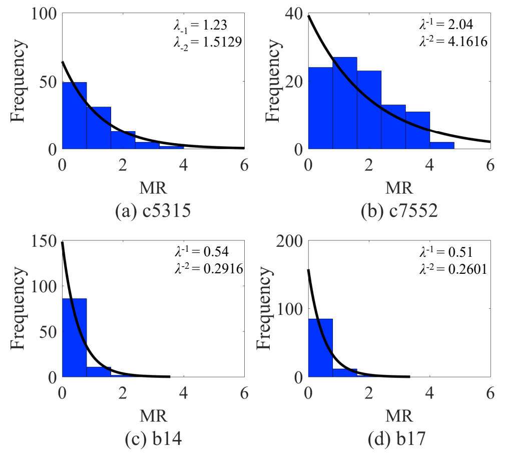

Figure 3: Histogram plots of the *MR* for different benchmark circuits with 128 key bits: (a) *c5315* (b) *c7552* (c) *b14* (d) *b17*

Acknowledgment

This work was supported by the National Science Foundation under grant number CNS-1755733.

References

[1] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," in *DATE*, pp. 1069 –1074, 2008.

[2] U. Guin, Q. Shi, D. Forte, and M. M. Tehranipoor, "FORTIS: a comprehensive solution for establishing forward trust for protecting IPs and ICs," *ACM Transactions on Design Automation of Electronic Systems*, 2016.

[3] U. Guin, Z. Zhou, and A. Singh, "Robust design-for-security architecture for enabling trust in IC manufacturing and test," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 818–830, 2018.

[4] U. Guin, Z. Zhou, and A. Singh, "A novel design-for-security (DFS) architecture to prevent unauthorized IC overproduction," in *Proc. of the IEEE VLSI Test Symposium (VTS)*, pp. 1–6, 2017.

[5] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Int. Symp. on Hardware Oriented Security and Trust*, pp. 137–143, 2015.