HARDWARE DEOBFUSCATION USING SAT ATTACK

BRANDEN LEONG | BRANDEN.LEONG.1@GMAIL.COM
ARCADIA HIGH SCHOOL | CLASS OF 2022
USC VITERBI DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING | SHINE 2020

USC Viterbi School of Engineering | SHINE Summer High School Intensive in Next-Generation Engineering

INTRODUCTION

Globalization of the modern economy has led to concerns about **pirating**, **overproduction**, and **counterfeiting**, especially for vendors of intellectual property (**IP**). If successful, these generate major **revenue loss** for IP vendors. To protect IP's from illegal and unauthorized manufacturing, a method for locking IP's has been proposed in recent years and is prevalently utilized in highly-classified systems.

Known as **logic locking**, this method integrates "**keys**" in IP's such that the IP will only behave as intended once the correct key combination is provided. This may be accomplished by inserting additional **logic gates** (for example, XOR and XNOR gates) that require one or more keys. Thus, the true functionality of the IP is **hidden** from the user while still accessable given the **correct key**.

Figure 1:
(left) the non-encrypted circuit; (right) the encrypted circuit, the correct key is $k_1 = 1$
PC: Yinghua Hu

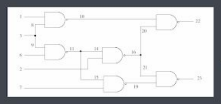
However, logic locking contains inherent **exploitable flaws**. Efficient attacks have been developed, such as **SAT attacks**, **sensitization attacks**, and **bypass attacks**.

For future reference, an IP containing keys will be known as the **encrypted circuit** and the same circuit sans logic locking components will be known as the **oracle**. Hence, the encrypted circuit will only function as the oracle given the correct key.

METHODS

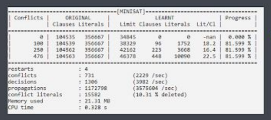
One major inherent flaw in traditional logic locking techniques is derived from the fact that logic locking **fails to obfuscate** the design of the circuit. All gate connections are still viewable by the user.

Figure 2:
1) We take the netlist of the circuit and note all connections...



2) ... then use Tseytin transformations to convert into something the SAT solver can read...

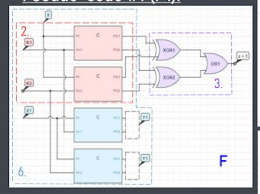
3) ...then run it through the SAT solver!
PC: Branden Leong

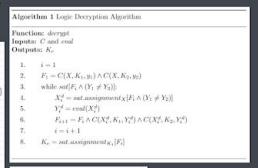


IMPLEMENTATION

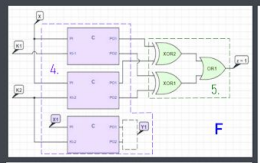
We implemented two different **algorithms** captured by the following **pseudo-codes** (methods of obtaining the desired result).

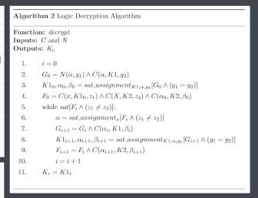
Pseudo-code #1 (P1):

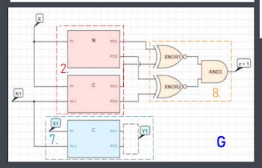




Pseudo-code #2 (P2):







P2 was developed to **balance** the two SAT solver calls. The run-time per call is **exponential** with the size of the logic formula; thus it is necessary to make each call as small as possible to reduce run-time.

Figure 3: (clockwise from top-left) 1st SAT solver (SS) call of P1; P1 code; 2nd SS call of P1; P2 code; 2nd SS call of P2; 1st SS call of P2
PC: Branden Leong

SKILLS LEARNED

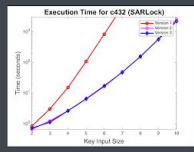
- **GitHub** to store and organize both current code and previous revisions
- Creation and utilization of a **Linux** environment on a Windows machine
- Understanding of **run-time complexity**
- How to express Boolean circuits in **DIMACS format**
- Operation of **MINISAT** satisfiability solver
- Primitive and advanced **logic locking functions** applied on combinational and sequential circuits

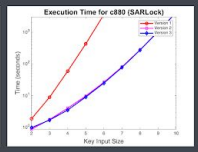
RESULTS

We created three versions of the code: 1) P1 works, but **not time-efficient**; 2) **improved time-inefficient** part of the preprocessing; and 3) P2 implementation, aiming to **reduce run-time further**.

All three versions have been tested against the **c432** and **c880** circuits in the **ISCAS-85 benchmark library** with **SARLock**, a SAT attack-**resilient** encryption technique.

Figure 4:
The average runtime for each of the three versions over 20 trials for (left) c432; and (right) c880
PC: Branden Leong





For 6 key inputs, the second and third versions have about a 98% decrease in run-time!

FUTURE PLANS

Because of how similar the runtimes are for the second and third implementations, we can logically deduce that the **preprocessing** before and after the SAT solver runs is taking **too long** in the third implementation. We expect the SAT solver time to be **significantly decreased**, hence we must work towards **decreasing** the preprocessing time.

We should further implement a method for solving for keys in **sequential circuits**. During the research program, only the combinational circuits have been solved, so we wish to expand on our progress by solving sequential circuits as well.

Logic locking is not the only method for protecting IP's; there are **many other methods** that must be tested to identify vulnerabilities should we continue on the journey to discover an enigma of the future.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation and gratitude to the following people, all of whom have made this great research experience possible...

...to **Professor Nuzzo** for accepting me into his research team and the opportunity to engage in electrical engineering research; ...
...to Ph. D student **Yinghua Hu** for his endless encouragement, support and invaluable teaching, and most of all for his patience in guiding me through obstacles; ...
...to **Rachel Loh** for going through this amazing experience with me; ...
...to **Dr. Mills** and the **SHINE team** for coordinating the program during especially difficult times and for cultivating a warm atmosphere for research; ...
...and most certainly **my family** for supporting me through this transformational journey.