Proceedings of the 28th International Conference *"Mixed Design of Integrated Circuits and Systems"*
June 24-26, 2021, Łódź, Poland

Hardware Obfuscation of the 16-bit S-box in the MK-3 Cipher

Jason Blocklove
Department of Computer Engineering
Rochester Institute of Technology
jmb8314@rit.edu

Steve Farris
L3Harris Technologies
Steve.Farris@L3Harris.com

Michael Kurdziel
L3Harris Technologies
Mike.Kurdziel@L3Harris.com

Marcin Łukowiak
Department of Computer Engineering
Rochester Institute of Technology
mxleec@rit.edu

Stanisław Radziszowski
Department of Computer Science
Rochester Institute of Technology
spr@cs.rit.edu

***Abstract*—At different stages of the Integrated Circuit (IC) lifecycle there are attacks which threaten to compromise the integrity of the design through piracy, reverse engineering, hardware Trojan insertion, side channel analysis, and other physical attacks. Some of the most notable challenges in this field deal specifically with Intellectual Property (IP) theft and reverse engineering attacks. One method by which some of these concerns can be addressed is by introducing hardware obfuscation to the design in various forms. In this work we evaluate the effectiveness of a few different forms of netlist-level hardware obfuscation of a 16-bit substitution box component of a customizable cipher MK-3. These obfuscation methods were attacked using a satisfiability (SAT) attack, which is able to iteratively rule out classes of keys at once. This has been shown to be very effective against many forms of hardware obfuscation. A method to successfully defend against this attack is described in this paper. This method introduces a modified SIMON block cipher as a One-way Random Function (ORF) that is used to generate an obfuscation key. The S-box obfuscated using this 32-bit key and a round-reduced implementation of the SIMON cipher is shown to be secure against a SAT attack for at least 5 days.**

***Keywords*—hardware obfuscation, substitution box, SAT attack**

I. Introduction

With the continuous growth of cyberinfrastructure in modern society, secure computing, storage, and communication require that the underlying framework utilizes both software and dedicated hardware components. These hardware components have traditionally been used to enhance overall systems' performance, cost, and energy efficiency. Unfortunately, adding specialized hardware into different points of the state-of-the-art cyberinfrastructure can create new security threats that did not exist in the past. These threats necessitate development of new methodologies and approaches for secure hardware design and manufacturing. The objective is to prevent malicious behavior such as insertion of hardware Trojans, overproduction, manufacturing of counterfeit devices, reverse engineering, or side-channel analysis [1], [2], [3].

One common technique used to protect hardware IP against these threats is to obfuscate the implementation itself, such that necessary design analysis becomes too expensive, in either time or resources, for an adversary to realistically accomplish.

The purpose of this work was to evaluate the effectiveness of some of the proposed hardware obfuscation techniques on a substitution box (S-box) from a customizable, authenticated encryption (AE) cipher MK-3 [4]–[6]. We utilized satisfiability (SAT) attacks [7] as a method of attacking hardware obfuscation. The effects which different methods have on protecting the S-box component from these types of attacks were evaluated.

II. Background

A. MK-3 Algorithm

The MK-3 algorithm is a single pass, customizable, authenticated encryption algorithm supporting 128- and 256-bit key sizes. The core of the MK-3 cipher is a permutation function f, which is comprised of 4 transformations: substitution, permutation, mixing, and add round constant. All of these transformations are computed on an internal state S of $b = 512$ bits. The state is split into a rate $r = 128$ and capacity $c = 384$, such that $b = r + c$ [4]. This permutation function f is used as the core of a duplex sponge construction [8]. This is a slight modification to the basic sponge, which maintains state between calls while absorbing and squeezing the data during each iteration [8]. Figure 1 illustrates this approach in the context of the MK-3 authenticated encryption scheme. Desired functionality can be achieved through successive duplexing calls with the key, initialization vector (IV), additional authenticated data (AAD) and blocks of the plaintext (M_i) provided as input. Each absorbed r-bit plaintext block M_i is used to produce one ciphertext block C_i, and the authentication tag T is produced at the end.

Customized versions of the MK-3 algorithm can be implemented by altering the round function in specific ways designed to preserve the algorithm's security while ensuring that different versions of the algorithm are not interoperable [6]. One of the components that can be modified in the MK-3 algorithm is its 16-bit S-box. The S-box is the IP

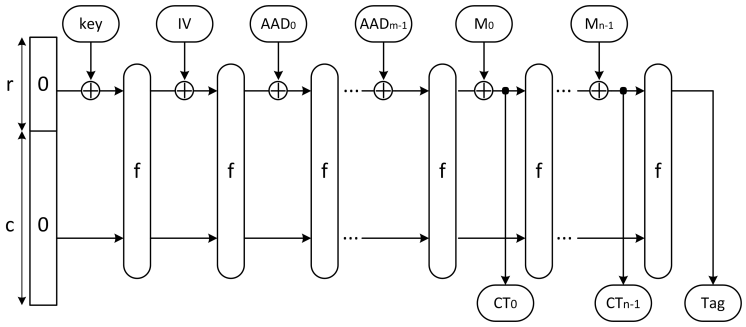

Fig. 1. The duplex construction

being obfuscated in this work [5]. Similar to the well-known Advanced Encryption Standard (AES) S-box, the MK-3 S-box finds the inverse of the input in the Galois field $GF(2^{16})$ and passes it through an affine transformation. The default case is as follows:

$$\begin{pmatrix} 0&0&1&0&0&0&0&1&0&0&1&1&1&1&1&0\\ 1&1&0&0&0&0&0&1&0&1&1&0&1&0&1&0\\ 1&1&0&0&1&0&1&1&0&1&0&1&0&0&1&1\\ 1&1&1&0&0&0&1&0&0&1&1&0&0&0&0&0\\ 1&1&0&0&0&1&1&0&0&1&1&1&1&0&1&1\\ 0&1&0&0&0&0&1&1&0&1&1&1&1&1&0&1\\ 0&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0\\ 1&0&1&1&1&0&1&1&0&0&0&1&0&1&1&1\\ 0&1&0&0&0&0&0&0&1&0&0&1&1&1&0&1\\ 1&0&1&1&0&0&0&1&0&0&1&0&1&0&0&0\\ 1&0&1&0&0&1&1&1&0&0&1&1&0&1&0&0\\ 1&0&1&1&1&0&1&1&1&0&1&1&0&0&0&1\\ 1&0&1&0&0&1&0&1&1&0&0&1&0&0&0&1\\ 0&1&0&0&0&1&1&1&1&0&0&0&0&0&0&1\\ 1&0&0&0&1&1&0&1&0&1&1&1&1&0&0&0\\ 1&1&0&1&0&1&1&0&1&0&0&1&1&0&0&0 \end{pmatrix} \times \begin{pmatrix} x_{15}\\ x_{14}\\ x_{13}\\ x_{12}\\ x_{11}\\ x_{10}\\ x_9\\ x_8\\ x_7\\ x_6\\ x_5\\ x_4\\ x_3\\ x_2\\ x_1\\ x_0 \end{pmatrix}^{-1} \oplus \begin{pmatrix} 0\\1\\0\\0\\0\\1\\0\\1\\1\\0\\1\\1\\0\\1\\1\\1 \end{pmatrix}.$$

B. Hardware Obfuscation

Hardware obfuscation in the context of this work refers to different methods by which the functionality of a design is modified to make it infeasible to understand or use without the knowledge of a secret key, which is provided to the system through additional input(s).

Netlist Logic Locking: Netlist-based logic locking describes several obfuscation methods which rely on inserting additional combinational logic into a gate level netlist. This additional logic, usually in the form of Key Programmable Gates (KPGs), can be added to the circuit randomly [9], based on a number of different heuristics [10], or by employing methods similar to those used in fault-analysis of digital circuits [11]. This additional logic can also take the form of complete circuits added alongside the original logic being obscured [12]. XOR and XNOR gates are often used as these KPGs, as they have the unique characteristic that based on a single bit input they can act as either a simple buffer or an inverter for the other input. Several techniques were developed with the specific intention of preventing particular attacks on obfuscated circuits such as key sensitization or SAT attacks [7], [13]. For example, SARLock, or SAT Attack Resistant Logic Locking, aims to reduce the number of possible keys that the SAT attack is able to rule out with every iteration [12]. This is done by introducing a comparator into the design, which compares the key to the circuit input. For certain combinations of the key and input, the comparator produces a *flip* signal, which is XORed with a primary output, thus inverting it. This ensures that for each input pattern only a single key will produce an incorrect value, while only the correct key will produce the correct output for all inputs. This method on its own is not enough to fully protect a circuit, though, so it was recommended to be combined with Strong Logic Locking (SLL) for additional protection against other attacks. Anti-SAT is another netlist locking method of hardware obfuscation designed to thwart the SAT attack [14]. The aim of Anti-SAT is to include a small additional circuit that greatly increases the number of iterations the SAT attack will take to break the function, similar to SARLock. This is achieved by adding two additional functional blocks, g and $\overline{g}$, which share the primary inputs but have differing keys and whose outputs are either ANDed or ORed together. These blocks generate a similar *flip* signal to that found in SARLock and are also recommended to be combined with another obfuscation method like SLL. LUT-Lock [15] describes another algorithm which aims to place locking logic such that the difficulty of a SAT attack [7] is greatly increased. LUT-Lock points to several different ways to utilize Look-Up Tables (LUTs) for stronger logical obfuscation. In particular, in Field Programmable Gate Arrays (FPGAs) logic can be placed in larger-than-necessary LUTs, with the additional inputs being fed by Non-Linear Feedback Shift Registers (NLFSRs) or Physical Unclonable Functions (PUFs) to create locks. These can act as internal keys, so that the bitstream will only function properly on a specific device that is configured to produce that particular key input.

ORF Insertion: To provide a barrier between the primary key inputs to a circuit and a locked netlist, Yasin and his team proposed inserting an ORF between the primary key inputs and the key inputs of the locking gates within the locked

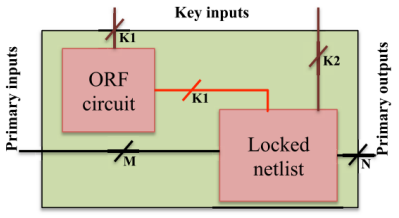

Fig. 2. One-way Random Function (ORF) Insertion [16]

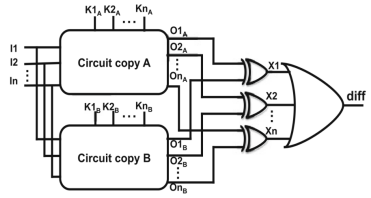

Fig. 3. DIP Miter Circuit [17]

netlist [16]. This is shown in Fig. 2.

The introduction of an ORF prevents attacks that can determine the key inputs to the locking gates from its outputs by disassociating the outputs from the primary key inputs. The researchers discuss using a modified AES block cipher with a hard-coded key as the ORF for these locking schemes. The hard-coded key value is implemented to prevent possible removal attacks. Since the architecture of the AES is well known, it is possible that an attacker could identify and remove it; however, by including the key in the design itself, once it is synthesized it would become much harder to achieve.

C. Satisfiability Attack

Several attacks have been developed with the intention of determining the secret key that allows unlocking an obfuscated circuit. The attack which was used in this work is referred to as the SAT attack. Originally proposed in [7], it rules out equivalence classes of keys that do not lead to the correct output. To accomplish this attack, the adversary must have access to both the locked netlist and a functional unlocked IC (unobfuscated netlist in the case of simulated attack). By defining equivalence classes, the SAT attack is able to rule out sets of keys rather than individual values as it would be in a simple brute-force approach, thus increasing the efficiency of the search. Once equivalence classes are defined, the attack iteratively rules out incorrect keys through the use of Distinguishing Input Patterns (DIPs). A DIP is defined as an input vector where, for two different keys $\vec{K_1}$ and $\vec{K_2}$ the output is different. This output can then be compared to the output of the unlocked circuit for that input vector and one or both equivalence classes can be ruled out. This allows the attack to rapidly reduce the size of the key space in order to narrow in on the correct key.

A miter-like circuit is created to find the DIPs, which compares two copies of the circuit with the same primary inputs and two different keys [17]. The outputs of these circuits are compared to form a `diff` signal, which is asserted high if any outputs differ, as shown in Fig. 3.

This circuit is converted to its Conjunctive Normal Form (CNF) and a SAT solver is run on the description of the circuit. A DIP has been found once there is a satisfiable assignment. Then the output of the activated IC is checked to rule out either or both of the keys used — and by extension, those keys' equivalence classes. This process is repeated either until no further DIPs can be found, indicating that the equivalence class of the key can no longer be narrowed down and the correct key has been found.

Algorithm 1 describes the full process of the attack.

Algorithm 1 SAT Attack [7]

Inputs: C and $eval$
Output: $\vec{K_C}$
1: **function** DECRYPT
2: $i := 1$
3: $F_1 = C(\vec{X}, \vec{K_1}, \vec{Y_1}) \wedge C(\vec{X}, \vec{K_2}, \vec{Y_2})$
4: **while** $sat[F_i \wedge (\vec{Y_1} \neq \vec{Y_2})]$ **do**
5: $\vec{X_i^d} := sat\_assignment_{\vec{X}}[F_i \wedge (\vec{Y_1} \neq \vec{Y_2})]$
6: $\vec{Y_i^d} := eval(\vec{X_i^d})$
7: $F_{i+1} = F_i \wedge C(\vec{X_i^d}, \vec{K_1}, \vec{Y_i^d}) \wedge C(\vec{X_i^d}, \vec{K_2}, \vec{Y_i^d})$
8: $i := i + 1$
9: **end while**
10: $\vec{K_C} := sat\_assignment_{\vec{K_1}}(F_i)$

Once the `while` condition (*line 3*) is no longer true, meaning the miter circuit is unsatisfiable, the correct key value $\vec{K_C}$ is output.

III. Methodology

A. Circuit Obfuscation

A Python software application was implemented that integrates multiple steps of the obfuscation process as well as the actual attack framework. If not already provided as a gate-level netlist, the circuits were first synthesized using Leodardo Spectrum. For this synthesis a generic Application-Specific Integrated Circuit (ASIC) standard gate library was used, and a helper script was written in Python to convert the gate definitions from that library into standard Verilog primitives. The locking methods implemented were: 1) the random locking method proposed in [9], as well as two which place the locking gates at either 2) the high or 3) the low stages of the circuit, 4) LUT-Lock — our slightly simplified version of a more recently proposed locking method made to specifically defend against SAT attacks [15], 5) a further modified LUT-Lock, and 6) using block cipher SIMON as an ORF along with the random logic locking. Our implementation of LUT-Lock is based on the original NB2-MO-HSC method from [15], with the only alteration being removal of the timing considerations. Modified LUT-Lock describes the modified instance of the NB2-MO-HSC locking scheme where all outputs are made to be candidates in the initial formation of the candidate gate list. The flow of using our framework and the SAT attack tool from [7] is shown in Fig. 4.

B. SIMON as an ORF

Several variations of the SIMON block cipher [18] were used as One-way Random Functions (ORFs) in the manner discussed in [16]. The SIMON configuration used, prior to any modifications, was the 32/64 implementation of the cipher with a 32-bit block size and a 64-bit key. The circuit model was derived from a VHDL implementation of the 64/128 SIMON block cipher [19]. In an attempt to establish a trend that relates the number of rounds of SIMON to its strength as an ORF for this application, several round-reduced implementations of

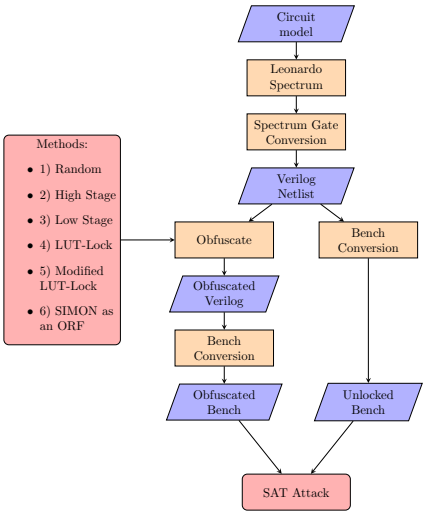

Fig. 4. Obfuscation and Attack Workflow

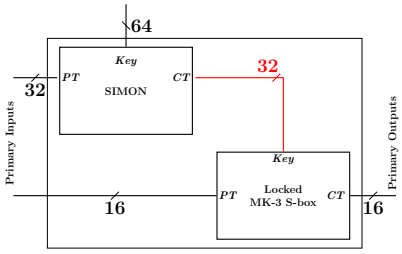

Fig. 5. Example SIMON ORF MK-3 S-box Test Setup

the cipher were also created and tested. For the round-reduced tests, a SAT attack was run on the round-reduced SIMON implementation, with the "unlocked" IC being an instance of the SIMON netlist with a fixed key. The goal of the SAT attack was to determine that fixed key.

Further experiments with this method involved fixing both the plaintext and the key inputs to the cipher, respectively, as it was being used as an ORF. Fig. 5 shows one of the main top-level configurations for these tests. The shown configuration uses a full implementation of SIMON with no fixed inputs being used as an ORF to generate the key for obfuscation. The configurations with a fixed SIMON plaintext and with a fixed SIMON key were set up in the same manner. The actual number of locking gates used for the obfuscation here was fixed at 32 to match the size of the SIMON output, and were randomly distributed. For each input variation of SIMON: 2, 4, 6, 8, 10, 12, 14, 16, 24, and 32-round instances were tested.

IV. Results & Analysis

A. Test Setup

For each obfuscation method and configuration (key size, number of SIMON rounds) the attack was executed ten times, and the amount of time the attack took to complete was recorded. A timeout was set so that if the attack ran without

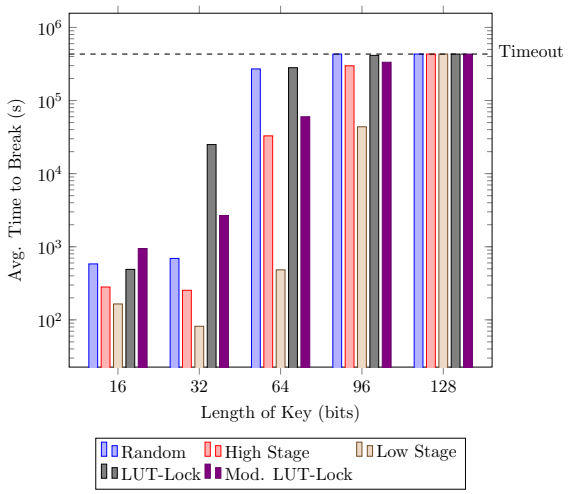

Fig. 6. SAT Attack on MK-3 S-Box obfuscated with Different Methods. "Timeout" indicates an unsuccessful attack after 5 days (432,000 seconds).

successfully finding the key for more than five days it was stopped. All tests were conducted on a server with Intel Xeon Gold 6150 CPUs running at 2.70GHz. Each instance of a test was run on a single thread and was given access to 24GB of RAM.

B. Logic Locking, Methods 1 – 5

The first series of tests conducted were applications of locking methods 1 — 5 with varying key size/number of locking gates. The results of these tests are shown in Tab. I and Fig. 6. The timeout condition is first achieved for a 96-bit key and pure random placement of the locking gates. For 128-bit key none of the implemented methods breaks within the specified timeout.

To further analyze the effect of a key size, a series of tests were run with finer granularity as shown in Fig. 7. In this experiment keys sizes were in the range between 96 and 128 bits, spaced by 4 bits.

As shown in the figure, not all tests consistently reached the timeout until a 128-bit key was used. The results showed that the best obfuscation outcomes/security for the S-box is achieved for random placement of the locking gates.

C. ORF Added Logic Locking, Method 6

Our next tests involved the ORF insertion scheme as outlined in [16]. In this work it was suggested that the

TABLE I.
AVERAGE SAT ATTACK BREAK TIME (SECONDS) ON MK-3 S-BOX. "TIMEOUT" INDICATES UNSUCCESSFUL ATTACK AFTER 5 DAYS (432,000 SECONDS).

Key Size	Random	High Stage	Low Stage	LUT-Lock	Mod. LUT-Lock
16	582	282	165	490	946
32	692	254	82	24,974	2,676
64	270,520	32,866	483	281,141	59,993
96	Timeout	298,616	43,737	414,580	333,862
128	Timeout	Timeout	Timeout	Timeout	Timeout

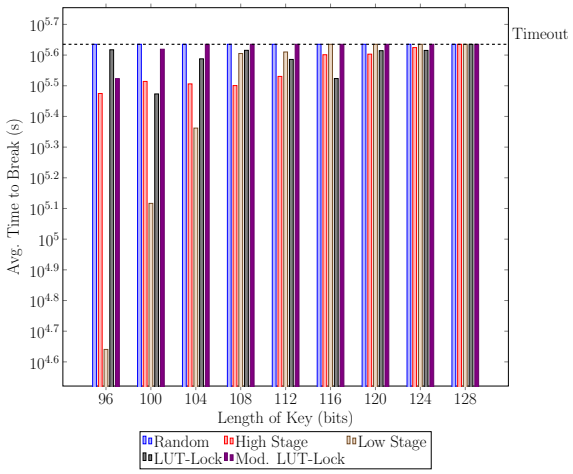

Fig. 7. SAT Attack on MK-3 S-Box obfuscated with different methods. Finer grain keys between 96 and 128-bits "Timeout" indicates an unsuccessful attack after 5 days (432,000 seconds).

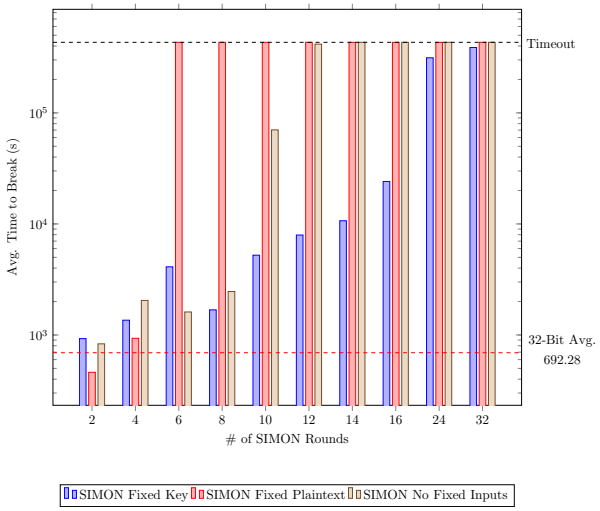

Fig. 8. SAT Attack Results on Locked circuits with SIMON Configurations as ORFs

ORF was an implementation of the AES with a fixed key. However, since hardware implementations of the AES require significant amount of resources and could incur too much overhead for obfuscating smaller circuits, we chose to use SIMON, a lightweight hardware-optimized block cipher in our approach [20].

First, we ran attacks on SIMON and round-reduced implementations of the SIMON cipher itself. This was done to evaluate the potential of different possible configurations of SIMON as an ORF. The implementations with 1 – 6 rounds were broken rather quickly, many in fractions of a second, but 7 rounds or more were unable to be broken in the given timeout range of five days with this attack. After those tests, three base configurations with fixed key, fixed plaintext, and no fixed inputs were developed as it is shown in Fig. 5. By fixing one of the cipher inputs before synthesis, the created implementation has unrecognizable structure and thus may prevent possible removal attacks. The implementation with no-fixed input was examined to determine if having the full cipher can provide additional security against SAT attacks, despite being more susceptible to removal attacks.

Rather than modulating the key size in these tests, the number of rounds of SIMON was changed. This was done to determine if the full cipher is required or if a round-reduced, smaller implementation could be used and still provide adequate security. The locking method used for all these tests was the random scheme with a 32-bit key, which corresponds to the 32-bit ciphertext output of the SIMON selected configuration. Tab. II shows the results of these tests on the MK-3 S-box. Fig. 8 presents the same results as a bar chart.

Adding SIMON as an ORF vastly increases security in both the cases of fixed plaintext and no fixed inputs. Fixing the plaintext seems to retain the most strength at fewer rounds. Across all circuits and implementations the tests which used a fixed key performed worse than the other two methods, though still offered notable added security over methods 1 – 5 with 32 locking gates/key. Implementations with as few as two rounds showed a greater resistance to the attack than the average amount of time the SAT attack took to break the MK-3 S-box circuit obfuscated with methods 1 – 5.

D. Cost

Including SIMON does incur some significant overhead which must be accounted for when designing a circuit. Tab. III gives data on the gate overhead for each configuration of SIMON that was tested.

Each implementation and number of rounds adds a different amount of gates to accompany the different amount of security

TABLE II.
AVERAGE SAT ATTACK BREAK TIME (SECONDS) ON MK-3 S-BOX WITH SIMON AS AN ORF. "TIMEOUT" INDICATES AN UNSUCCESSFUL ATTACK AFTER 5 DAYS (432,000 SECONDS).

# of Rounds	Fixed Key	Fixed Plaintext	No Fixed Inputs
2	927	461	832
4	1,359	934	2,047
6	4,108	Timeout	1,612
8	1,683	Timeout	2,462
10	5,228	Timeout	70,290
12	7,944	Timeout	416,078
14	10,679	Timeout	Timeout
16	24,125	Timeout	Timeout
24	313,616	Timeout	Timeout
32	388,227	Timeout	Timeout

TABLE III.
SIMON 32/64 OVERHEADS (# OF 2-INPUT GATES)

Rounds:	2	4	6	8	10	12	14	16	24	32
Fixed Key	107	211	320	433	551	659	770	870	1338	1765
Fixed Plaintext	67	200	435	675	919	1155	1391	1628	2583	3540
No Fixed Inputs	128	256	495	717	955	1194	1432	1672	2634	3586

offered. The fixed key implementation is the smallest but offered the least security, fixed plaintext is notably larger but gave much more SAT attack resistance at fewer rounds, and the no fixed input implementation is just slightly larger than with fixed plaintext and also offered substantial security with slightly more rounds. With the non-ORF logic locking methods, the overhead is one gate per key bit.

V. Conclusions

The methods explored in this work offered varying levels of success, with random gate placement and ORF insertion ultimately offering the most security for the tested configurations, in some cases increasing the attack time from fractions of a second to several days. Additionally, the MK-3 S-box showed a much higher resistance to SATs attacks than the standard ISCAS '85 benchmarks as reported in [21], which could indicate that non-linear components like S-boxes are by design harder to break in this way.

We are convinced that using an ORF to transform key bits before another locking mechanism is the right approach. With a sufficient number of key bits we can expect any SAT attack to fail, if the ORF being used has proper cryptographic strength. The 32 bits used in our experiments, however, may not be enough for high security requirements.

Future work might include more detailed analysis of the effectiveness of different configurations of SIMON (key, plaintext, and ciphertext sizes) or other lightweight ciphers for use as ORFs. For example - the National Institute of Standards and Technology (NIST) competition for lightweight ciphers [22] is ongoing and several interesting candidates are being considered for standardization. Further work may also need to be conducted in preventing possible removal attacks against the ORF, if its structure is too recognizable.

References

[1] R. Chakraborty and S. Bhunia, "HARPOON: An obfuscation-based SoC design methodology for hardware protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, pp. 1493–1502, 10 2009. [Online]. Available: http://dx.doi.org/10.1109/tcad.2009.2028166

[2] R. S. Chakraborty and S. Bhunia, "Security against hardware trojan attacks using key-based design obfuscation," *Journal of Electronic Testing*, vol. 27, pp. 767–785, 12 2011. [Online]. Available: http://dx.doi.org/10.1007/s10836-011-5255-2

[3] R. P. Cocchi, J. P. Baukus, L. Chow, and B. J. Wang, "Circuit camouflage integration for hardware IP protection," *Proceedings - Design Automation Conference*, 06 2017.

[4] M. Kelly, A. Kaminsky, M. Kurdziel, M. Łukowiak, and S. Radziszowski, "Customizable sponge-based authenticated encryption using 16-bit S-boxes," in *MILCOM 2015 - 2015 IEEE Military Communications Conference*, 2015, pp. 43–48.

[5] C. A. Wood, S. P. Radziszowski, and M. Łukowiak, "Constructing large S-boxes with area minimized implementations," in *MILCOM 2015 - 2015 IEEE Military Communications Conference*. IEEE, 10 2015, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/milcom.2015.7357417

[6] P. Bajorski, A. Kaminsky, M. Kurdziel, M. Łukowiak, and S. Radziszowski, "Customization modes for the Harris MK-3 authenticated encryption algorithm," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, 2018, pp. 1–5.

[7] 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)*. IEEE, 5 2015, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/hst.2015.7140252

[8] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, "Duplexing the sponge: single-pass authenticated encryption and other applications," in *Selected Areas in Cryptography*. Springer, 2012, pp. 320–337.

[9] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending piracy of integrated circuits," in *2008 Design, Automation and Test in Europe*. IEEE, 3 2008, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/date.2008.4484823

[10] S. Roshanisefat, H. K. Thirumala, K. Gaj, H. Homayoun, and A. Sasan, "Benchmarking the capabilities and limitations of SAT solvers in defeating obfuscation schemes," in *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*. IEEE, 7 2018, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/iolts.2018.8474189

[11] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Logic encryption: a fault analysis perspective," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE 2012)*. IEEE, 3 2012, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/date.2012.6176634

[12] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 5 2016, inproceedings. [Online]. Available: http://dx.doi.org/10.1109/hst.2016.7495588

[13] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *the 49th Annual Design Automation Conference*. ACM Press, 6 2012, inproceedings. [Online]. Available: http://dx.doi.org/10.1145/2228360.2228377

[14] Y. Xie and A. Srivastava, "Anti-SAT: Mitigating SAT attack on logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, pp. 199–207, 2 2019. [Online]. Available: http://dx.doi.org/10.1109/tcad.2018.2801220

[15] H. Mardani Kamali, K. Zamiri Azar, K. Gaj, H. Homayoun, and A. Sasan, "LUT-lock: A novel LUT-based logic obfuscation for FPGA-bitstream and ASIC-hardware protection," in *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2018, inproceedings, pp. 405–410.

[16] 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, pp. 1411–1424, 9 2016. [Online]. Available: http://dx.doi.org/10.1109/tcad.2015.2511144

[17] J. Rajendran and S. Garg, *Logic Encryption*. Springer International Publishing, 1 2017, pp. 71–88. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-49019-9_3

[18] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK families of lightweight block ciphers," 2013. [Online]. Available: https://github.com/nsacyber/simon-speck

[19] J. Wetzels and W. Bokslag, "Simple SIMON: FPGA implementations of the SIMON 64/128 block cipher," *arXiv*, Jul 2015. [Online]. Available: http://arxiv.org/abs/1507.06368v1

[20] A. Aysu, E. Gulcan, and P. Schaumont, "SIMON says: Break area records of block ciphers on FPGAs," *IEEE Embedded Systems Letters*, vol. 6, pp. 37–40, 6 2014. [Online]. Available: http://dx.doi.org/10.1109/les.2014.2314961

[21] J. Blocklove, "Hardware intellectual property protection through obfuscation methods," MS thesis, Rochester Institute of Technology, 2020. [Online]. Available: https://scholarworks.rit.edu/theses/10470

[22] NIST, "Lightweight cryptography," 2018. [Online]. Available: https://csrc.nist.gov/Projects/lightweight-cryptography