CycSAT-Unresolvable Cyclic Logic Encryption Using Unreachable States

Amin Rezaei, You Li, Yuanqi Shen, Shuyu Kong, and Hai Zhou
Northwestern University, Evanston, IL, USA
me@aminrezaei.com, {you.li, yuanqishen2020, shuyukong2020}@u.northwestern.edu, haizhou@northwestern.edu

ABSTRACT

Logic encryption has attracted much attention due to increasing IC design costs and growing number of untrusted foundries. Unreachable states in a design provide a space of flexibility for logic encryption to explore. However, due to the available access of scan chain, traditional combinational encryption cannot leverage the benefit of such flexibility. Cyclic logic encryption inserts key-controlled feedbacks into the original circuit to prevent piracy and overproduction. Based on our discovery, cyclic logic encryption can utilize unreachable states to improve security. Even though cyclic encryption is vulnerable to a powerful attack called CycSAT, we develop a new way of cyclic encryption by utilizing unreachable states to defeat CycSAT. The attack complexity of the proposed scheme is discussed and its robustness is demonstrated.

KEYWORDS

Cyclic Logic Encryption; Unreachable States; CycSAT Attack

1 INTRODUCTION

Increasing the design costs of Integrated Circuits (ICs) on the one hand, and growing the number of untrusted third-party foundries on the other hand, make chip protection one of the vital priorities for the semiconductor industry. Leakage of the IC layout to an insecure environment may lead to piracy and overproduction. The main approach to prevent unauthorized products from working is logic encryption in which the functionality of the IC is first locked by inserting key-controlled gates [2, 7, 9] and then obfuscated by a sequence of resynthesis. However, the SAT-based attack [14] on encrypted acyclic combinational circuits gives an attacker the chance to efficiently decrypt the circuit using merely a few input-output observations taken from an activated IC.

In addition, there is a common conjecture that combinational circuits should be designed without any cycles, but circuits with cycles can be combinational as well [1]. Therefore, a cycle in a circuit can be either combinational or non-combinational. Although a combinational cycle is harmless to the SAT-based attack, a non-combinational cycle may prevent the SAT solver from finishing or return a wrong key. Thus, in order to overcome the inefficiency of the original SAT-based attack on cyclic logic encryption, two versions of the CycSAT attacks [21] are proposed.

ASPDAC '19, January 21–24, 2019, Tokyo, Japan

ACM ISBN 978-1-4503-6007-4/19/01…$15.00
https://doi.org/10.1145/3287624.3287691

However, almost all ICs are sequential circuits in which scan chain is widely used for testing purposes. In scan chain, the sequential circuit works in two different modes named as regular and testing. A 2-1 MUX is placed at the input of each Flip-Flop (FF) in order to connect all FFs in a shift register for one MUX selection while the FFs work in the regular mode for the other MUX selection. In the testing mode, the circuit acts like a combinational one. If FFs have initial values correspond to the initial state of the circuit, the set of reachable states may be smaller than the entire space. Thus, there are some value combinations for FFs which can never be realized in the regular mode starting with the initial state [5].

In this paper, first we review state-of-the-art works in logic encryption and explain their vulnerabilities toward existing attacks. Then, we consider the concept of state unreachability more precisely and discuss the opportunity that unreachable states can provide in logic encryption. Afterward, we propose a cyclic logic encryption approach using unreachable states in order to address the drawbacks of the previous works. The attack complexity discussion is also given. Finally, we demonstrate the efficiency of the proposed approach with experimental results.

2 BACKGROUND

To protect a combinational circuit with an n-bit key, a simple procedure is developed that uses n new gates [9]. First, n wires are selected and matched with the bits of the key. Then, for each selected wire, its driver is disconnected from the sinks and an XOR (or XNOR) gate is inserted. Moreover, MUX-based scheme is proposed [7] such that one input of the 2-1 MUX is the correct wire while the other input is the wrong one. In this case, the selector of the MUX is the associated key bit.

On the other hand, the SAT-based attack is proposed [14] to efficiently defeat almost all of the traditional methods. The attack uses two copies of the encrypted circuit with the same input but different key values under a given constraint to check whether it is still possible to generate different outputs. Such input patterns are called Differentiating Input Patterns (DIPs.) The idea of using DIP is to exclude at least one wrong key from consideration. However, each DIP can exclude a large number of wrong keys in most cases.

In order to increase the required number of DIPs exponentially with the key size, counter measures like SARLock [20] and Anti-SAT [17] have been introduced. Since these incremental techniques have very small error number, they are vulnerable to approximate attacks such as AppSAT [11], Double DIP [13], and Bypass attack [18] that can return an almost correct key. The mentioned SAT-proof techniques are also vulnerable to Bit-Flipping attack [12] since they have a recognizable logic signature in their design.

Different from previous works, a cyclic logic encryption approach is proposed [10]. The scheme inserts key-controlled dummy feedbacks to the originally acyclic circuit based on the fact that

Algorithm 1: CycSAT-I attack

Input: Encrypted circuit $g(x, k)$ and original function $f(x)$
Output: Correct key k^* such that $g(x, k^*) \equiv f(x)$
Initialization: Find a set of feedback signals $(w_0, ..., w_m)$ and compute "no structural path" formulas $F(w_0, w_0'), ..., F(w_m, w_m')$

$NC(x, k_1) = \bigwedge_{i=0}^{m} F(w_i, w_i')$;
$g(x, k_1) = g(x, k_1) \wedge NC(k_1)$;
$g(x, k_2) = g(x, k_2) \wedge NC(k_2)$;
while $\hat{x} = SAT(g(x, k_1) \neq g(x, k_2))$ **do**
  $g(x, k_1) = g(x, k_1) \wedge (g(\hat{x}, k_1) = f(\hat{x}))$;
  $g(x, k_2) = g(x, k_2) \wedge (g(\hat{x}, k_2) = f(\hat{x}))$;
$k^* = SAT(g(x, k_1))$;

Algorithm 2: CycSAT-II attack

Input: Encrypted circuit $g(x, k)$ and original function $f(x)$
Output: Correct key k^* such that $g(x, k^*) \equiv f(x)$
Initialization: Find a set of feedback signals $(w_0, ..., w_m)$ and compute "no sensitizable path" formulas $F(w_0, w_0'), ..., F(w_m, w_m')$

$NC(x, k_1) = \bigwedge_{i=0}^{m} F(w_i, w_i')$;
while $\hat{x} = SAT(NC(x, k_1) \wedge NC(x, k_2) \wedge g(x, k_1) \neq g(x, k_2))$ **do**
  $g(x, k_1) = g(x, k_1) \wedge (g(\hat{x}, k_1) = f(\hat{x}))$;
  $g(x, k_2) = g(x, k_2) \wedge (g(\hat{x}, k_2) = f(\hat{x}))$;
while $\hat{x} = SAT(\neg NC(x, k_1) \wedge g(x, k_1))$ **do**
  $g(x, k_1) = g(x, k_1) \wedge NC(\hat{x}, k_1)$;
$k^* = SAT(g(x, k_1))$;

cycles sometimes make trouble for the original SAT-based attack. Creating dummy feedbacks is based on two conditions. First, any created cycle has to have multiple entry points, and second, at least two edges in a cycle have to be removable. However, two versions of the CycSAT attacks [21] can efficiently decrypt the circuit with key-controlled feedbacks.

CycSAT-I assumes there is a correct key under which the circuit is acyclic. In other words, it considers all the feedbacks to be the dummy ones which is the case in [10]. Thus, it first computes a Conjunctive Normal Form (CNF) formula to capture the condition that there is no structural cycle in the circuit. Then, it adds this constraint to the encrypted circuit. The original SAT-based attack can finish the job on the constrained circuit.

On the other hand, CycSAT-II considers the circuit to behave fully combinational but maybe cyclic under a correct key. It means, it supposes the encrypted circuit with both dummy and real combinational cycles which is the case in [8]. So, it first computes a formula postulating that there is no sensitizable cycle in the circuit. Then, it constrains the encrypted circuit by the formula and runs the original SAT-based attack on the constrained circuit. Then, more iterations are needed to exclude non-combinational cycles. The pseudo-codes of CycSAT-I and CycSAT-II are given in Algorithms 1 & 2 respectively. In this paper, we propose a CycSAT-unresolvable cyclic logic encryption approach by falsifying CycSAT assumptions.

3 MOTIVATION

An unreachable state is defined as a state which has no incoming path starting from the initial state. Suppose we plan to implement a string pattern detector that each time there is a sequence of "101", the output generates "1". Figure 1a shows the state transition diagram for such circuit. Since two FFs are required for the implementation, one state (i.e., state D) is unreachable.

Formally, if $c_s(X, S, \delta, s_0, Y, \gamma)$ is described as a sequential circuit in which X is the set of input vectors, S is the set of all reachable and unreachable states ($S = S_r \cup S_u$,) δ is the next state function, s_0 is the initial state, Y is the set of output vectors, and γ is the output function, a state $s \in S$ is unreachable if it satisfies Equation (1).

$$\forall x \in X : s \neq \delta^*(x, s_0) \Leftrightarrow s \in S_u \quad (1)$$

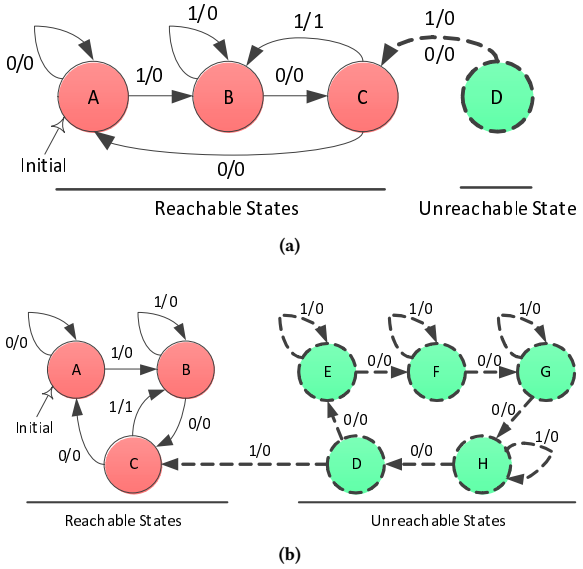

Figure 1: State transition diagram of string pattern detector (a) Two FFs (b) Three FFs

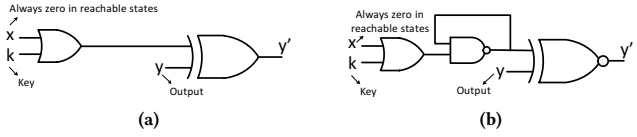

Figure 2: Unreachable state example (a) Acyclic logic encryption (b) Cyclic logic encryption

In which $\delta^*(x, s_0)$ is a set of next state functions without state repetition (i.e., with no loop) that starts from the initial state. Obviously, a state is either reachable or unreachable (i.e., $S_r \cap S_u = \emptyset$.)

Now, the question is whether the flexibility of the unreachable states can be exploited to mislead the attacker? In state-of-the-art

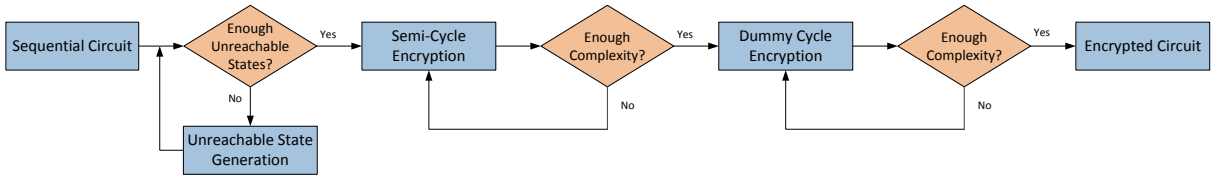

Figure 3: Unreachable state encryption

works of logic encryption, it is supposed that the attacker has access to the physical layout. Moreover, she can acquire a functioning circuit as a black-box and get the correct outputs for given input vectors. Also, since almost all ICs are sequential circuits, it is assumed that scan chain is accessible to the attacker.

3.1 Acyclic Logic Encryption

In acyclic encryption, the circuit always behaves combinational whether a correct key is inserted or not. In this case, although we may force the output to be flipped in an unreachable state, the behavior is still combinational and does not make a serious problem for the SAT solver. This is because the activated IC as the attacker's test bench always gives a result that matches with k^*.

As can be seen in Figure 2a, if signal x is always "0" in the reachable states, in order to keep the correct output (i.e., $y = y'$,) the key k should be assigned "0" value. However, when the circuit is in the unreachable state, x can be equal to "1". In this case, the output flips regardless of the k value. So, when the SAT solver tests $x = 1$, no key will be pruned since the output flips for both copies no matter $k = 0$ or $k = 1$. On the other hand, the wrong key (i.e., $k = 1$) will be pruned by testing $x = 0$ that results in deciphering the correct key (i.e., $k = 0$.)

3.2 Cyclic Logic Encryption

Although acyclic encryption cannot take advantage of the flexibility that unreachable states provide, cyclic encryption can fully utilize this opportunity if we make the output to behave non-combinational in unreachable states.

As an example, Figure 2b shows a cyclic encryption circuit. As long as the correct key (i.e., $k = 0$) is applied, the output of the OR gate will be always "0" which makes the output of the NAND gate to stay at "1". In this case, $y = y'$ and the circuit remains combinational. On the other hand, when the circuit is in the unreachable state, x can be equal to "1". Thus, the output of the OR gate will be "1" which makes the output of the NAND gate to oscillate between "0" and "1". So, y' oscillates even under the correct key value. The SAT solver cannot handle this non-combinational behavior properly and will report a random Boolean.

4 UNREACHABLE STATE ENCRYPTION

CycSAT-I can efficiently decrypt a cyclic encryption circuit as long as there is no real cycle under a correct n-bit key. On the other hand, CycSAT-II with weakening CycSAT-I assumption can successfully decrypt the circuit if the existing cycles under a correct key are combinational in all the input patterns. However, both versions have trouble attacking the circuits in which there exists cycles that behave non-combinational under any correct key. Thus, it is sufficient to identify or insert some unreachable states and force the original circuit to act non-combinational in these states. This works because the CycSAT attacks cannot differentiate between reachable and unreachable states in their testing procedure. Figure 3 depicts the platform of the proposed unreachable state encryption.

Formally, if $c_c(X, S, S', Y)$ is described as a combinational circuit in which X is the set of primary inputs, S is the set of state inputs, S' is the set of state outputs, and Y is the set of primary outputs, the cyclic encryption circuit can be written as:

$$g(X, S, K, S', Y) \mid \exists k^* \in K : g(X, S, k^*, S', Y) \equiv c_c(X, S, S', Y)$$

We want to solve the following problem:

$$S_u \subset S \Rightarrow g(X, S_u, k^*, S', Y) \neq c_c(X, S_u, S', Y) \tag{2}$$

4.1 Unreachable State Generation

Since the designer has access to the original circuit, she may be aware of the safety properties and existing unreachable states in the design. Also, she can employ efficient model checking tools like IC3 [3] to investigate more unreachable states that are hard to guess and intentionally add them to the original circuit. Figure 1b depicts an unreachable state loop that is created by adding an additional FF to the string pattern detector of Figure 1a. As can be seen, the number of unreachable states can become easily dominant with an extra FF. This happens because a linear increase to the number of FFs has an exponential effect to the number of states.

4.2 Non-Combinational Cycle Formation

The output in a combinational cycle depends only on the primary inputs. On the other hand, a cycle is non-combinational if its output changes even if the primary inputs are kept fixed. In a formal manner, the non-combinational cycle that is encrypted with a key k^* is characterized as the following CNF:

$$g(x, s_u, k^*, s', y_1) \wedge g(x, s_u, k^*, s', y_2) \wedge y_1 \neq y_2$$

In which $s_u \in S_u$, $x \in X$, $s' \in S'$, $y_1, y_2 \in Y$.

4.3 Semi-Cycle Encryption

Leveraging the unreachable states, we can insert many semi-cycles in the encryption circuit. Insertion of one such a cycle is shown in Figure 4. Assume $un(s)$ is characterizing condition of the unreachable states in which:

$$un(s) = 1 \Leftrightarrow s \in S_u$$

Two signals u and v with a path from u to v will be randomly selected from the original circuit $c_c(X, S, S', Y)$. Assume the gate producing u is an AND gate, otherwise, use De Morgan's law to translate the OR gate into a NAND gate. The semi-cycle will be

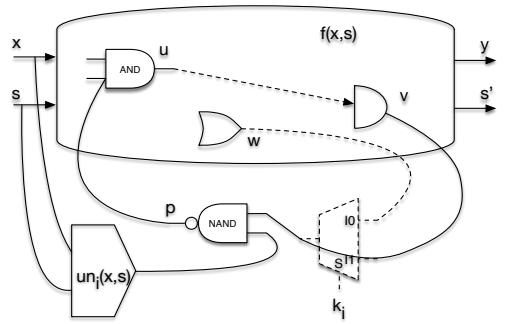

Figure 4: Semi-cycle encryption using unreachable states

introduced with a feedback path from v to u through a NAND gate with a random sub-condition $un_i(x,s)$.

$$\forall x \in X, s \in S : un_i(x,s) \Rightarrow un(s)$$

In order for the semi-cycle to behave non-combinational when $un_i(x,s) = 1$, we must make sure that $path(u,v) \wedge un_i(x,s)$ is satisfiable, where $path(u,v)$ is the sensitizable condition from the inputs of the chosen AND gate including new signal p to signal v. Under this condition, the value of v is dependent on the value of p. This implies that when we select the random path from u to v, we need to make sure that $path(u,v) \wedge un(s)$ is satisfiable. Then, we can select $un_i(x,s)$ to be any random implicant of $path(u,v) \wedge un(s)$.

At the end, at least one key bit with a MUX gate will be introduced on the semi-cycle to mislead the attacker. The other input of the MUX gate is a random signal w in the feed-backward path of the cycle. Algorithm 3 shows the semi-cycle encryption procedure.

THEOREM 1. *The encrypted circuit with semi-cycles cannot be decrypted by the CycSAT attacks if under any correct key, there is at least one semi-cycle that behaves non-combinational in an unreachable state.*

PROOF. The NC formula in CycSAT-I assures that there is no cycle in the circuit. Thus, if even one real cycle exists under k^*, the algorithm will break that cycle. So, k^* should be removed in one of the iterations of CycSAT-I. On the other hand, in CycSAT-II, each feedback signal i is broken into (w_i, w_i') where w_i' feeds to w_i before the break. Then, the algorithm excludes any key in which $w_i \neq w_i'$. However, based on the theorem assumption, under k^*, at least one feedback signal j exists in which $w_j \neq w_j'$. Thus, it exists a cycle that includes j even under k^*. So, k^* should be removed in one of the iterations of CycSAT-II. □

4.4 Dummy Cycle Encryption

In order to prevent a guessing attack in which all the key bits choose the cycles, dummy cycles with associated key bits should be also adopted [10]. Some random signals are chosen such that each of them is an input for more than one gate. Afterward, an additional 2-1 MUX is introduced for each of those signals. Subsequently, a random dummy feedback is introduced from the feed-forward path for one input of the MUX while the other input is connected to the original chosen signal. Here, the correct key bit should avoid the dummy feedback. In this case, differentiating between semi-cycles and dummy ones is not possible for the attacker.

Also in order to prevent a modified version of CycSAT that supposes the non-combinational cycles happens only under a correct key, there should be at least a dummy cycle that behaves non-combinational under a wrong key which can be achieved by the same sensitizable condition described in Section 4.3.

4.5 Attack Complexity Discussion

If we suppose $P = X \cup S$ as the set of primary and state input patterns each with size p, and K as the set of all possible key values each with size n, the error matrix can be seen as the following. We consider the encrypted circuit as $g(p,k)$ and the original circuit as $f(p)$ that is an arbitrary function with p inputs and one output.

$$E = \begin{array}{c} \\ p_0 \\ p_1 \\ \vdots \\ p_i \\ \vdots \\ p_{2^p-1} \end{array} \begin{array}{c} \begin{array}{ccccccccc} k_0 & k_1 & & k_j & & k^* & & k_{2^n-1} \end{array} \\ \begin{pmatrix} 1,N & 1,C & \dots & E_{0j} & \dots & 0,C & \dots & 0,C \\ 1,C & 0,N & \dots & E_{1j} & \dots & 0,N & \dots & 1,C \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ E_{i0} & E_{i1} & \ddots & E_{ij} & \ddots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1,C & 1,C & \dots & \dots & \dots & 0,C & \dots & 1,C \end{pmatrix} \end{array}$$

In which,

$$\begin{aligned} &\vee\ Com(g(p_i,k_j)) \wedge Com(f(p_i)) \wedge g(p_i,k_j) \neq f(p_i) \\ &\vee\ Com(g(p_i,k_j)) \wedge NonCom(f(p_i)) \\ &\vee\ NonCom(g(p_i,k_j)) \wedge Com(f(p_i)) \\ \Rightarrow\ & E_{ij} = 1 \end{aligned}$$

Simply speaking, if the outputs of g and f under a specific input and key combination behave combinational but produce different binary value, E_{ij} will be "1". In addition, if the outputs behave differently (i.e., one behaves combinational and the other non-combinational,) E_{ij} will be also "1". In this case, the error number of a key k_j is the number of input patterns in which $E_{ij} = 1$:

$$E_j = \sum_{i=0}^{2^p-1} E_{ij}$$

The minimal error number of an encryption scheme is the minimum error number among all the wrong keys:

$$E_{min} = \forall j \in 0 \ldots 2^n - 1 : min(E_j) \wedge E_j \neq 0$$

We also capture the behavior of the output under each input and key combination. C and N stand for combinational and non-combinational respectively. It is obvious that in order to prevent an approximate attack, E_{min} should be exponentially large. Also, the proposed unreachable state encryption is robust against structural attacks since the chosen v and w signals in semi-cycle insertion approach are randomly selected and $un_i(x,s)$ has randomness in it.

Now, we investigate two possible logic attacks on the proposed encryption scheme. We suppose that non-combinational behavior of each semi-cycle will be propagated to the primary output of the circuit. In addition, due to the state explosion problem [16], finding the set of all unreachable states in order to exclude them from the SAT solver is empirically impossible for the attacker.

Algorithm 3: Semi-cycle encryption

Input: Acyclic combinational circuit $c_c(X, S, S', Y)$, topological sort of signals T, set of all gates G, number of cycles to add l, and number of key bits to add in each cycle q

Output: Cyclic encryption circuit $g(X, S, K, S', Y)$

while *l>0* **do**

  Select random $i, j \in G$ s.t. $i \mapsto j \in T$;

  if i == OR **then**

    Apply De Morgan's law to i ;

  Find the set of all signals $(w_0, ..., w_m)$ from $i.In$ to $j.Out$;

  Compute sensitizable condition $path(i, j) = F(w_0, ..., w_m)$;

  if $SAT(path(i, j) \wedge un(s))$ **then**

    Select a random $un_i(x, s) \Rightarrow path(i, j) \wedge un(s)$;

    Add NAND$(j.Out, un_i(x, s))$ to $i.In$;

    $l = l - 1$;

  Call MUX-encryption $(path(i, j), q)$;

Return $g(X, S, K, S', Y)$;

First, we assume that non-combinational behavior is not detectable which is currently the case in state-of-the-art key-pruning attacks [11–14, 18, 21]. In this case, a random Boolean will be reported by the SAT solver for non-combinational cases.

Theorem 2. *When non-combinational behavior is not detectable and the number of non-combinational cases under any correct key is exponentially large, the probability of finding a correct key by any SAT-based attack is exponentially small.*

Proof. Based on the theorem assumption, the activated IC outputs a random Boolean for exponential number of tested DIPs. The probability for the test circuit to output the same value for each correct key is $\frac{1}{2}$. This probability becomes exponentially smaller by testing more DIPs. So, if the number of tested DIPs is d, the probability of finding a correct key is 2^{-d}. □

Now, we assume that non-combinational behavior of the activated IC can be detected by a ternary (i.e., 0, 1, and $\perp$) simulation [6].

Theorem 3. *When non-combinational behavior is detectable and the number of non-combinational cases under any correct key is exponentially large, the time complexity of any SAT-based attack is exponential.*

Proof. When non-combinational behavior of the activated IC under a DIP is detected, that DIP should be discarded and not be plugged into the SAT solver. Since the number of DIPs in which the activated IC behaves non-combinational is exponential, the attacker also requires to run exponential iterations of the test & discard operation. □

5 EXPERIMENTS

For experiments, we use acyclic combinational circuits of ISCAS'85 [4] and MCNC'91 [19]. First, four state inputs are added to the original benchmarks and considered to be always zero in reachable states. Then, twenty cycles with a random combination of dummy cycles and semi-cycles are inserted each equipped with a MUX-based key bit. In this case, each key bit can choose between a cycle (either real or dummy) or an acyclic path. For the semi-cycles, the correct key bit should choose the real cycle while for the dummy ones, it should avoid the dummy feedback and choose the acyclic path. It is clear that the overhead of the encryption approach is linear to the key size. Here we choose a constant key size (i.e., twenty bits) for each benchmark regardless of the original circuit size.

Since the code for CycSAT-II is not available, the decryption results under CycSAT-I are depicted in Table 1. As can be seen the attack in most of the cases returns no key because the correct key has already been pruned by the "no structural cycle" condition while returns wrong keys for a few cases.

As another experiment, we lock the original benchmarks with the acyclic MUX-based approach [7] with the same key size (i.e., twenty bits) and a random distribution of the key bits. Then, since an obfuscation step is required, we utilize ABC synthesis tool [15] for resynthesis. Figure 5 depicts the percentage increase on the critical path of the original unencrypted circuit after acyclic logic and unreachable states encryptions. The critical path increase in the acyclic method is on average 30% more than the unreachable states one. Not to mention the fact that the original SAT-based attack [14] can easily decrypt the acyclic benchmarks while even CycSAT cannot decrypt the unreachable state ones.

As anticipated in Theorem 1, when there are semi-cycles that behave non-combinational under any correct key, the circuit cannot be decrypted by CycSAT. In addition, cyclic logic encryption using

Table 1: Unreachable states encryption evaluation with semi/dummy cycles

Bench	CycSAT-I attack [21]		
	CPU Time (s)	#Iteration	Note
apex2	-	-	UNSAT
apex4	~~0.4~~	~~3~~	Wrong key
c432	-	-	UNSAT
c499	-	-	UNSAT
c880	-	-	UNSAT
c1355	-	-	UNSAT
c1908	-	-	UNSAT
c2670	-	-	UNSAT
c3540	-	-	UNSAT
c5315	-	-	UNSAT
c7552	-	-	UNSAT
dalu	-	-	UNSAT
des	-	-	UNSAT
ex5	~~0.096~~	~~2~~	Wrong key
ex1010	-	-	UNSAT
i4	-	-	UNSAT
i7	-	-	UNSAT
i8	-	-	UNSAT
i9	~~0.124~~	~~5~~	Wrong key
k2	~~0.168~~	~~3~~	Wrong key
seq	-	-	UNSAT

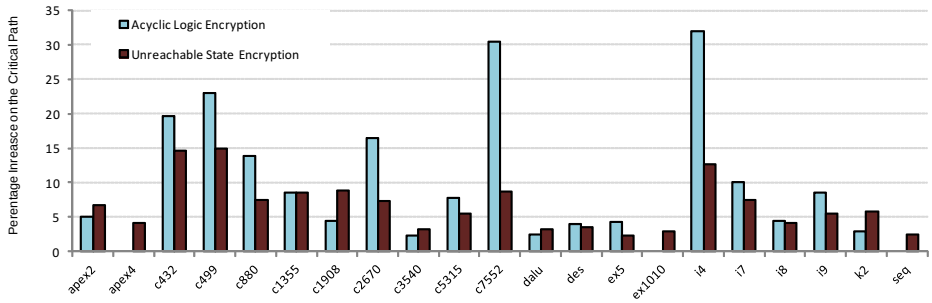

Figure 5: Percentage increase on the critical path in acyclic logic encryption and unreachable state encryption

unreachable states not only secures the chip with a small number of key bits, but also it adds less performance overhead to the circuit in comparison with the traditional acyclic approach.

6 CONCLUSION

We studied the influential role of unreachable states on chip protection by proposing a CycSAT-proof cyclic logic encryption approach. If the designer knows the safety properties and existing unreachable states, she utilizes them; otherwise she intentionally adds some unreachable states to the original circuit. Then, she chooses a combination of semi-cycles and dummy cycles to be added into the circuit each equipped with at least one MUX-based key bit. If the number of non-combinational cases under any correct key is exponentially large, any SAT-based attack is impractically impossible. Also, a linear increase to the number of FFs has an exponential effect to the number of states; therefore, an exponential gain can be provided in terms of chip protection by spending a linear area overhead. This makes unreachable state encryption a promising logic encryption solution.

ACKNOWLEDGMENT

This work is partially supported by NSF under CNS-1441695, CNS-1651695, and CCF-1533656.

REFERENCES

[1] J. Backes, B. Fett, and M. D. Riedel. 2008. The analysis of cyclic circuits with Boolean satisfiability. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 143–148.

[2] A. Baumgarten, A. Tyagi, and J. Zambreno. 2010. Preventing IC piracy using reconfigurable logic barriers. In *IEEE Design Test of Computers*, Vol. 27, Issue 1. 66–75.

[3] A. R. Bradley. 2011. SAT-based model checking without unrolling. In *International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*. 70–87.

[4] F. Brglez and H. Fujiwara. 1985. A neutral netlist of 10 combinational benchmark circuits and a target translator in Fortran. In *IEEE International Symposium on Circuits and Systems (ISCAS)*. 677–692.

[5] M. Fujita. 2015. Detection of test patterns with unreachable states through efficient inductive-invariant identification. In *IEEE Asian Test Symposium (ATS)*. 31–36.

[6] S. Malik. 2006. Analysis of cyclic combinational circuits. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 13, Issue 7. 950–956.

[7] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri. 2015. Fault analysis-based logic encryption. In *IEEE Transactions on Computers*, Vol. 64, Issue 2. 410–424.

[8] A. Rezaei, Y. Shen, S. Kong, J. Gu, and H. Zhou. 2018. Cyclic locking and memristor-based obfuscation against CycSAT and inside foundry attacks. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 85–90.

[9] J. A. Roy, F. Koushanfar, and I. L. Markov. 2008. EPIC: Ending piracy of integrated circuits. In *Design, Automation and Test in Europe (DATE)*. 1069–1074.

[10] K. Shamsi, M. Li, T. Maede, Z. Zhao, D. Z. Pan, and Y. Jin. 2017. Cyclic obfuscation for creating SAT-unresolvable circuits. In *ACM Great Lakes Symposium on VLSI (GLSVLSI)*. 173–178.

[11] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin. 2017. AppSAT: Approximately deobfuscating integrated circuits. In *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 95–100.

[12] Y. Shen, A. Rezaei, and H. Zhou. 2018. SAT-based bit-flipping attack on logic encryptions. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 629–632.

[13] Y. Shen and H. Zhou. 2017. Double DIP: Re-evaluating security of logic encryption algorithms. In *ACM Great Lakes Symposium on VLSI (GLSVLSI)*. 179–184.

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

[15] Berkeley Logic Synthesis and Verification Group. ABC: A system for sequential synthesis and verification. In *http://www.eecs.berkeley.edu/ alanmi/abc/*.

[16] A. Valmari. 1998. The state explosion problem. In *Lectures on Petri Nets I: Basic Models, Lecture Notes in Computer Science*, Vol. 1491. 429–528.

[17] Y. Xie and A. Srivastava. 2016. Mitigating SAT attack on logic locking. In *Cryptographic Hardware and Embedded Systems (CHES), Lecture Notes in Computer Science*, Vol. 9813. Springer, 127–146.

[18] X. Xu, B. Shakya, M. Tehranipoor, and D. Forte. 2017. Novel bypass attack and BDD-based tradeoff analysis against all known logic locking attacks. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*. Springer, 189–210.

[19] S. Yang. 1991. Logic synthesis and optimization benchmarks user guide version 3.0. In *Microelectronics Center of North Carolina (MCNC) International Workshop on Logic Synthesis*.

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

[21] H. Zhou, R. Jiang, and S. Kong. 2017. CycSAT: SAT-based attack on cyclic logic encryptions. In *International Conference on Computer-Aided Design (ICCAD)*. 49–56.