SAT-attack Resilient Sequential Locking

***Abstract*—SAT-attack is known to decrypt a *functionally correct key* of an encrypted combinational circuit. Recently, sequential locking was proposed as a defense to SAT-attack on sequential circuits and subsequently ScanSAT was proposed as an unlocking mechanism for the same. ScanSAT method converts the sequential locking problem to an instance of logic locking problem, and thereby decrypts the entire sequential key using SAT-attack. In this paper, we show that ScanSAT does not guarantee decryption of *functionally correct key*, and a defense against ScanSAT. The proposed defense had $95-100\%$ success rate when tested on sequentialized ISCAS'85 and MCNC benchmarks, across five different encryption schemes, and $1-3$ orders of magnitude increase in decryption time for large circuits.**

I. Introduction

Today, fabless model is the most economical and hence the preferred business model of the semiconductor industry. However, IC counterfeiting, piracy and overbuilding in the untrusted offshore foundry have caused major concerns in electronic and defense industries [1]–[3].

Logic encryption uses a low-overhead combinational chip-locking system, to combat these issues [3]. Several logic encryption have been proposed so far in literature [3]–[7], and each of them have their own merits and demerits. The SAT-attack [8] is shown to successfully decrypt the logic encryption keys in all above cases, with over 95% success rate. Recently, sequential locking [9] was proposed as a defense to SAT-attack on sequential circuits. The flip-flop encryption is done in this paper, by inserting an XOR gate right at the output of selected flip-flops.

A major limitation of this technique is that $58-100\%$ of flip-flops were encrypted, which induces huge are overhead in real designs. Apart from that, the most recently proposed ScanSAT method [10] converts the sequential locking problem to an instance of logic locking problem, and thereby decrypts a *functionally correct key* using SAT-attack. Thus, it is imperative to come up with a new sequential locking scheme that is area-efficient, as well as, and more importantly resilient to the SAT-attack.

In this paper, we propose a sequential locking scheme that addresses the aforementioned challenges. The main contributions of this paper are as follows:

1) We show that ScanSAT does not guarantee decryption of *functionally correct key*, in the presence of encrypted XOR-chains at combinational outputs;
2) We show there is 100% correlation between failure of ScanSAT and presence of encrypted XOR-chains at combinational outputs;
3) Exploiting this property as a defense mechanism, we propose a new low-overhead encrypted scan cell, that consists of an unencrypted functional path and an encrypted scan path;
4) Finally, to increase the success rate of successful functional corruption, we propose an iterative combinational key gate pushing algorithm, that makes the sequential circuit resilient to SAT-attack.

II. Attack model

The attack model used in this paper is same as the one used in [8]. More specfically, we consider a malicious foundry, where the attacker has access to layout and mask information. Hence, the gate-level netlist can be reverse-engineering from this. We also assume that the attacker has access to an activated IC on which to apply input patterns and observe outputs. This could be obtained by purchasing an activated IC from the open market. The components of our attack model are therefore: (i) a gate-level netlist of the encrypted IC and (ii) a means for applying arbitrary input patterns and observing the resultant outputs on an activated IC.

III. Access model

Since the SAT-attacker needs access to every flip-flop in the IC, this can be achieved by operating the IC in scan mode of operation by asserting the test mode signal. In today's SoCs, it is only possible to access the scan architecture through embedded deterministic test (EDT) architecture, the on-chip decompression/compression scheme, which also comes with a bypass mode for debug purposes. Moreover, the SAT-attacker wants to apply very specific values to these flip-flops to force the logic nodes to specific states and capture the exact responses, in order to decrypt one of the correct keys. This is possible only in the EDT-Bypass mode. In summary, the access model is (i) initializing flip-flops during scan shift; and (ii) using EDT bypass mode to have direct access to scan-chains.

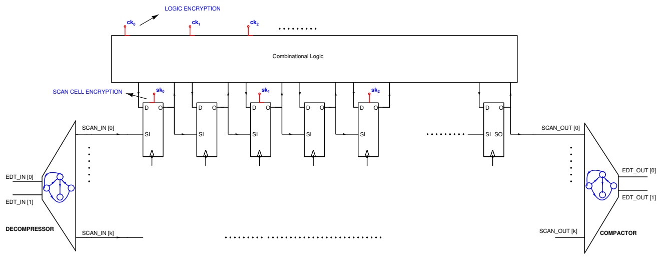


Figure 1. DFT Encryption

Figure 1 shows the encrypted design-for-testability (DFT) architecture, where both the combinational logic and flip-flops are encrypted, while the decompressor/compactor are

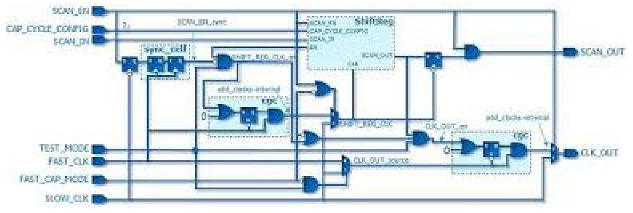
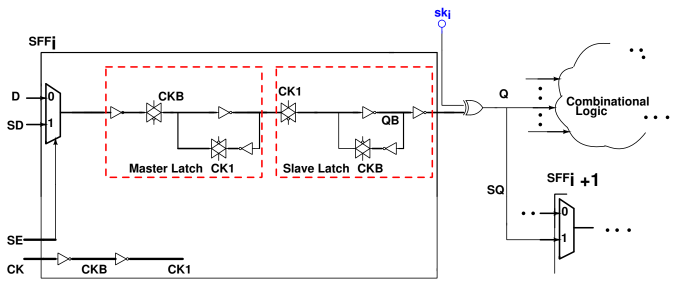


Figure 2. On-chip clock control

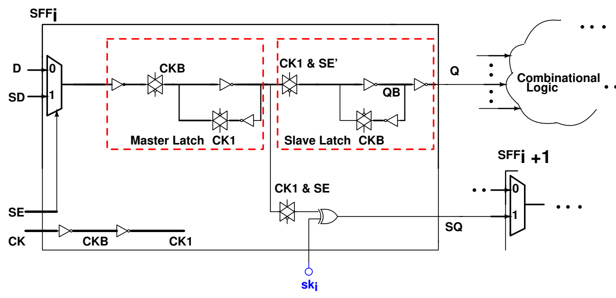
unencrypted. Hence, the decompressor and compactor should be bypassed, to directly access the scan chains. Figure 2 shows the on-chip clock control scheme used in modern SoCs. The attacker places the IC in scan mode of operation by asserting *TEST_MODE* signal shown in this figure. To apply the desired input vector to the combinational portion of the IC, the attacker assigns the flip-flops to desired values by placing the IC in scan-shift mode, by further asserting *SCAN_EN* signal. After initialization, the *SCAN_EN* signal is deasserted and *SLOW_CLK* is pulsed, to capture the response back into the flip-flops. The *SCAN_EN* signal is asserted again to read-out the captured response is read-out, while parallelly reading-in next attack vector. The attacker repeats this procedure until all distinguishing input patterns are applied to the IC.

IV. Proposed Encrypted Flip-Flop

Existing and proposed encrypted flip-flops are shown in figures 3(a) and 3(b) respectively.



(a) Existing Encrypted Flip-Flop



(b) Proposed Encrypted Flip-Flop

Figure 3. Proposed Vs. Existing Encrypted Flip-Flop

V. Results

Running time comparisons are shown in Figure 4.

References

[1] M. Pecht and S. Tiku, "Bogus: electronic manufacturing and consumers confront a rising tide of counterfeit electronics," *IEEE Spectrum*, vol. 43, no. 5, pp. 37–46, 2006.

[2] S. Trimberger, "Trusted design in fpgas," in *IEEE Design Automation Conference*, pp. 5–8, 2007.

[3] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending Piracy of Integrated Circuits," in *IEEE/ACM Design, Automation and Test in Europe*, pp. 1069–1074, 2008.

[4] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *IEEE/ACM Design, Automation and Test in Europe*, pp. 1069–1074, 2008.

[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. 21, no. 5, pp. 410–424, 2015.

[6] 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 *IEEE International On-Line Testing Symposium*, pp. 49–54, 2014.

[7] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing ic piracy using reconfigurable logic barriers," *IEEE Design and Test of Computers*, vol. 27, no. 1, pp. 66–75, 2010.

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

[9] R. Karmakar, S. Chattopadhyay, and R. Kapur, "Encrypt flip-flop: A novel logic encryption technique for sequential circuits," *CoRR*, vol. abs/1801.04961, 2018.

[10] L. Alrahis, M. Yasin, H. Saleh, B. Mohammad, M. Al-Qutayri, and O. Sinanoglu, "ScanSAT: Unlocking obfuscated scan chains," in *IEEE Asia South Pacific Design Automation Conference*, 2019.

Table I
SUCCESS IN FUNCTIONAL OUTPUT CORRUPTION

Bench.	RND (5%)		DAC12 (5%)		IOLTS14 (5%)		TOC13'XOR (5%)		TOC13'MUX (5%)	
	ScanSAT [10]	Proposed	ScanSAT [10]	Proposed	ScanSAT [10]	Proposed	ScanSAT [10]	Proposed	ScanSAT [10]	Proposed
apex2	✗	✓	✗	✗	✗	✓	✓	✓	✗	✓
i4	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
c432	✗	✓	✗	✓	✗	✓	✗	✓	✗	✓
ex1010	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
dalu	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
apex4	✗	✓	✗	✓	✗	✓	✓	✓	✓	✓
c3540	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓
c1908	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓
c880	✓	✓	✓	✓	✗	✓	✗	✓	✗	✓
c1355	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
c499	✗	✓	✗	✓	✗	✓	✗	✓	✗	✓
seq	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
k2	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
ex5	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
i9	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓
i7	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
i8	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
c7552	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
c5315	✓	✓	✓	✓	✗	✓	✓	✓	✗	✓
des	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
Success rate	80%	100%	10%	95%	0%	100%	70%	100%	5%	100%

Table II
INCREASE IN DECRYPTION TIME

Ben.	#POs	RND [3]			DAC12 [4]			IOLTS14 [6]			TOC13'XOR [5]			TOC13'MUX [5]		
		[6]	Prop.	Inc.	[6]	Prop.	Inc.	[6]	Prop.	Inc.	[6]	Prop.	Inc.	[6]	Prop.	Inc.
apex2	3	0.18s	0.26s		0.13s	0.75s	5.8×	0.05s	0.05s	1×	0.14s			0.38s	0.07s	0.2×
i4	6	0.05s			0.05s	0.08s	1.6×	0.03s	0.07s	2.3×	0.11s			0.04s	0.07s	1.8×
c432	7	0.02s	0.04s	2×	0.02s	0.04s	2×	0.03s	0.04s	1.3×	0.04s	0.07s		0.03s	0.04s	1.3×
ex1010	10	7.44s			13.6s	16.6s	1.2×	1.20s	3.49s	2.9×	23.75s			3.48s	5.45s	1.6×
dalu	16	1.74s			2.35s	5.8s	2.5×	0.31s	7.67s	24.7×	49.59s			1.15s	7.47s	6.5×
apex4	19	5.2s	14.6s		19.8s	24s	1.2×	0.75s	3.04s	4.1×	53.47s			6.21s	25.37s	4.1×
c3540	22	1.94s			4.87s	11.9s	2.4×	0.26s	0.34s	1.3×	2.30s	3.86s		2.63s	3.10s	1.2×
c1908	25	0.54s			0.88s	4.6s	5.2×	0.33s	1.54s	4.7×	19.25s	12.01s		0.18s	1.60s	8.9×
c880	26	0.14s			0.13s	0.89s	6.9×	0.07s	1.14s	16.3×	0.59s	3.96s		0.10s	1.81s	18.1×
c1355	32	3.75s			0.39s	3.6s	9.2×	0.05s	0.83s	16.6×	26.05s			0.13s	2.53s	19.5×
c499	32	0.13s	2.72s		0.27s	3.2s	11.9×	1.13s	1.41s	1.3×	0.27s	1.25s		0.09s	2.72s	30.2×
seq	35	3.6s			5.8s	6.2s	1.1×	1.82s	3.27s	1.8×	281s			3.22s	16.96s	5.3×
k2	45	2.6s			2.37s	17.8s	7.5×	0.34s	9.24s	27.2×	3.29s			0.49s	6.40s	13.1×
ex5	63	1.68s			0.69s	36.1s	52.3×	8.6s	174s	20.2×	6.99s			1.88s	6.52s	3.5×
i9	63	2.6s			1.1s	26.1s	23.7×	0.29s	16.7s	57.6×	1.56s	23.27s		0.78s	17.81s	22.8×
i7	67	2.68s			1.02s	13.6s	13.3×	0.11s	5.64s	51.3×	185s			0.52s	3.01s	5.8×
i8	81	7.37s			7.6s	79.7s	10.5×	0.22s	8.18s	37.2×	8.84s			2.40s	3.69s	1.5×
c7552	108	11.7s			50.5s	1 hr	71.3×	3.20s	27.3s	8.5×				5.48s	≈ 11 hrs.	7226×
c5315	123	12.2s			14.4s	146s	10.1×	4.64s	515s	111×	714s			2.91s	62.79s	21.6×
des	245	66.6s			408s	2668s	6.5×	2.52s	375s	149×	67.0s	3247s		24.97s	≈ 2 hrs.	288.4×

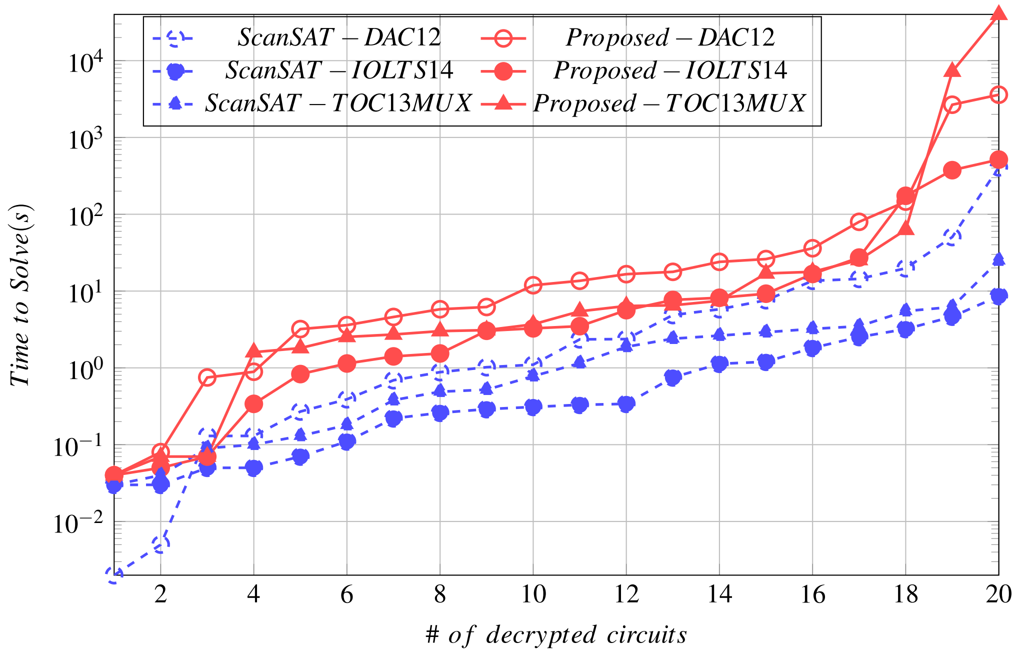


Figure 4. Running Time Comparisons