INVITED PAPER

A Primer on Hardware Security: Models, Methods, and Metrics

The paper is a primer on hardware security threat models, metrics, and remedies.

By Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri

ABSTRACT | The multinational, distributed, and multistep nature of integrated circuit (IC) production supply chain has introduced hardware-based vulnerabilities. Existing literature in hardware security assumes *ad hoc* threat models, defenses, and metrics for evaluation, making it difficult to analyze and compare alternate solutions. This paper systematizes the current knowledge in this emerging field, including a classification of threat models, state-of-the-art defenses, and evaluation metrics for important hardware-based attacks.

KEYWORDS | Counterfeiting; hardware Trojans; IP piracy; reverse engineering; side-channel attacks

I. INTRODUCTION

A. Motivation

All algorithmically secure cryptographic primitives and protocols rely on a hardware root of trust to deliver the expected protections when implemented in software. Similarly, critical control and communication functions assume that the hardware platforms that they are implemented on are resilient to attacks. However, this is not the case as the following examples demonstrate.

Quo Vadis Labs has reported backdoors in an integrated circuit (IC) that is used in weapons control systems, nuclear power plants, and public transportation systems [1]. Reports from the U.S. Government indicate that counterfeit electronics are prevalent in computers, communications, automobile, control, and defense systems [2], [3]. Ethical hackers showed that, by spoofing the communication signals between the parking payment smart card and the payment meter, one can add value onto these smart cards [4]. A demonstration at the 2012 Black Hat Conference showed a security vulnerability in hotel keycards [5]. The attacker exploited the small key search space offered by the cryptographic algorithm implemented in the keycard to expose the master key used to unlock all rooms. Microcontrollers are extensively used in embedded systems, and they are equipped with "fuse bits" to prevent unauthorized users from reading or modifying selected sections of its memory. A reverse engineer has been able to electrically reset these fuse bits and thereby gain modify/read access to the contents of its memory [6].

Cost, power consumption, performance, and reliability are considered while designing an IC. Security is an afterthought. An increase in the number and destructive power of hardware-based attacks has highlighted the need for securing the hardware root of trust side by side of power, cost, performance, and reliability optimizations. An emerging body of research in hardware security is addressing these problems [7]–[11]. While the progress in this field has been significant, the approaches taken by researchers has been largely *ad hoc*. Different assumptions are typically made concerning the hardware-based vulnerabilities, threats that exploit them, models for the considered threats, and defenses. Consequently, developed defenses cannot be compared against each other, even when they address the same hardware security problem.

Within this context, this paper systematizes the knowledge for a number of important contemporary problems in hardware security. It classifies hardware-based threats, defenses, and metrics to evaluate the effectiveness of the developed defenses.

B. IC Supply Chain

We start by describing the IC supply chain shown in Fig. 1. This supply chain is distributed worldwide [7], [12],

Manuscript received April 15, 2014; revised June 30, 2014; accepted June 30, 2014. Date of current version July 18, 2014.
This work was supported in part by the U.S. Office of Naval Research (ONR) under Grant R17460 and the National Science Foundation (NSF) under Grants to Rice University (CNS-1059416) and NYU (CNS-1059328 and CCF-1319841), NYU/NYU-AD Center for Research in Information Security Studies and Privacy (CRISSP), and ARO W911NF-13-1-0272.
M. Rostami and **F. Koushanfar** are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: masoud@rice.edu; farinaz@rice.edu).
R. Karri is with the Department of Electrical and Computer Engineering, Polytechnic Institute of New York University, Brooklyn, NY 11201 USA (e-mail: rkarri@nyu.edu).
Digital Object Identifier: 10.1109/JPROC.2014.2335155

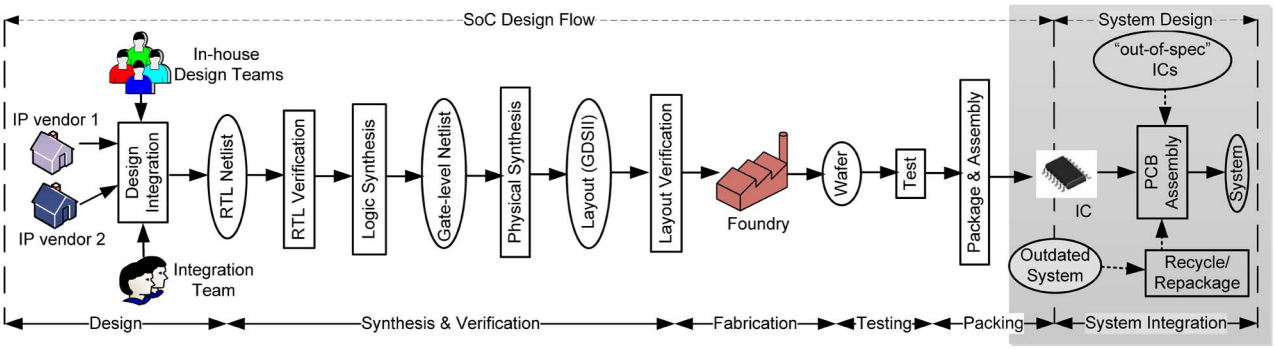

Fig. 1. ***Semiconductor supply chain: IC design flow (only the steps and entities that are relevant to this paper are shown). System design (the dotted lines represent how the fake and low-quality components enter the supply chain). Source: [13].***

and the emerging hardware security problems arise because of the global trends in IC design, manufacturing, and distribution in this supply chain. Designing an IC involves procuring intellectual property (IP) designs from third-party design houses, designing some components inhouse, combining both, and generating the IC layout. A blueprint of the design (e.g., in terms of GDS-II layout format) is then sent to the foundry that develops a costly mask and manufactures the ICs. The ICs are then tested at the manufacturing site and often also at third-party test facilities. Finally, fault-free ICs are packaged and sold. There are multiple points within this supply chain where things can go wrong. The following hardware-based threats are possible.

- Hardware Trojans: An attacker either in the design house or in the foundry may add malicious circuits or modify existing circuits.
- IP piracy and IC overbuilding: An IP user or a rogue foundry may illegally pirate the IP without the knowledge and consent of the designer. A malicious foundry may build more than the required number of ICs and sell the excess ICs in the gray market.
- Reverse engineering (RE): An attacker can reverse engineer the IC/IP design to his/her desired abstraction level. He can then reuse the recovered IP or improve it.
- Side-channel analysis: An attacker can extract the secret information by exploiting a physical modality (power consumption, timing, or electromangnetic emission) of the hardware that executes the target application.
- Counterfeiting: An attacker illegally forges or imitates the original component/design.

C. Systematization of Hardware Security Knowledge

Fig. 2 systematizes the hardware security knowledge centered around the attack method. The left column shows the goals of the attack, and the right column shows the location of the attacker within the IC supply chain.

Fig. 3 presents this hardware security knowledge in terms of the hardware-based attacks, countermeasures, and metrics for evaluation. The left, attack column abstracts the scenarios pertaining to each attack class, the middle column summarizes the countermeasures, and right column shows the metrics for evaluation of the countermeasures. The description of the attack scenarios is application dependent.

A preliminary version of this paper has appeared in [13]. The paper is organized from the perspective of the state of the art in hardware-based attacks. Section II focuses on hardware Trojans. Section III details IP piracy and IC overbuilding. Section IV discusses reverse engineering. Section V explains side channels, and Section VI describes counterfeiting. For each attack, the threat model, the state-of-the-art defenses, and the metrics used to evaluate the defenses are systematized. Section VII concludes the paper.

II. HARDWARE TROJANS

A hardware Trojan is a malicious modification to a circuit. The Trojan may control, modify, disable, or monitor the contents and communications of the underlying computing device [14]–[16]. Trojan detection is difficult for multiple reasons. First, the inherent opaqueness of the IC internals hurdles detection of the modified components; conventional parametric IC testing methods have a limited effectiveness because of the classic observability issues, and destructive tests and IC RE are slow and expensive. Second, technology scaling to the limits of the device physics and mask imprecisions cause a nondeterminism in a chip's characteristics making the distinction between the process variation and Trojans hard. Finally, there is a large (uncharacterized) space in the IC for the possible Trojans.

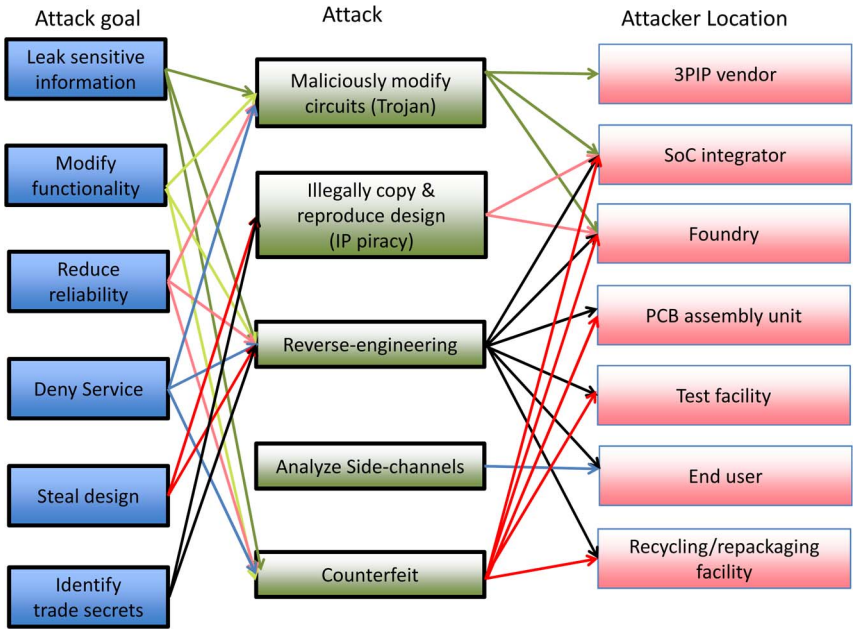

Fig. 2. *Systematization of hardware security around the attack method. The left column shows the goals of the attack, and the right column shows the location of the attacker.*

A. Threat Models

Table 1 illustrates two common scenarios for a hardware Trojan attack. In the first scenario, an attacker in the foundry inserts a Trojan into the design by manipulating the lithographic masks. These Trojans are in the form of addition, deletion or modification of gates [15], [16].

In the second scenario, a malicious IP is designed either by a rogue in the third-party IP (3PIP) design house

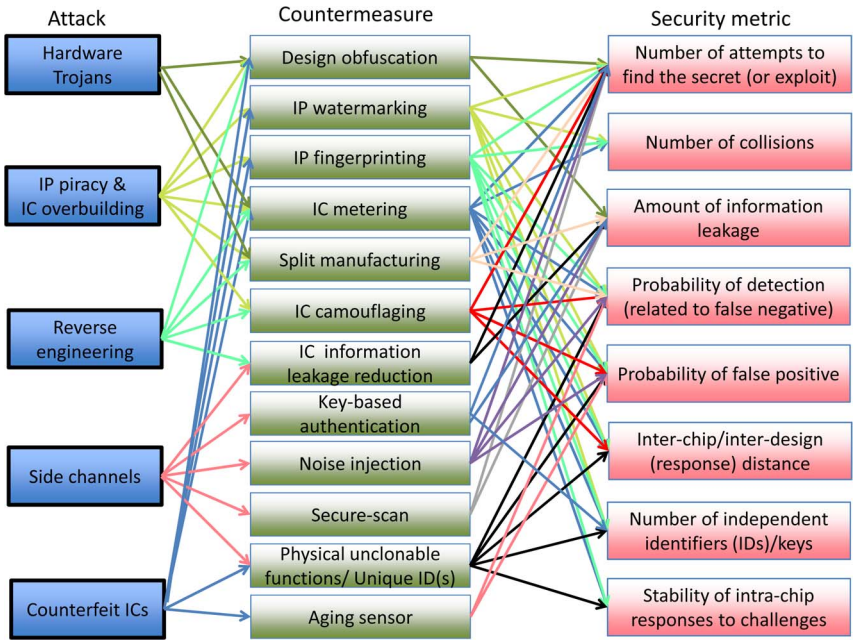

Fig. 3. *Hardware security knowledge in terms of the hardware-based attacks, countermeasures, and metrics for evaluation. The left column abstracts the attack scenarios, the middle column abstracts the countermeasures, and the right column shows the metrics for evaluation.*

Table 1 Two Hardware Trojan Attack Scenarios: (i) by an Attacker in the Foundry; (ii) by a Rogue in the 3PIP Vendor. The Bullet (•) Depicts an Attacker, the Star (⋆) Represents a Defender, and the Dash (—) Indicates an Untrustworthy Entity

Scenario	3PIP Vendor	SoC Integrator	Foundry	User
1	–	⋆	•	–
2	•	⋆	–	–

[15]–[17] or by a rogue in the inhouse design team [18]–[20]. It is unlikely that the malicious insider provides information about the inserted Trojan; without this information, the validation team may not be able to detect it.

B. State-of-the-Art Defenses

Most techniques attempt to detect Trojans inserted in the foundry [15], [16]. There are at least two possible ways to detect this class of Trojans: invasive and noninvasive. Invasive (and semi-invasive) detection methods make the tested components unusable afterwards. These methods require costly, precision measurement equipments that only big silicon companies can afford [15], [16].

Noninvasive detection methods rely on external parametric and functional IC testing. These methods excite the circuit under test (CUT) with input patterns and measure the corresponding output values as well as the side channels, e.g., delay, quiescent leakage, and dynamic leakage [14]. Variants of functional and statistical tests are also used. Examples include transient power analysis [21], [22], path-delay measurements [23], gate-level characterization [24], [25], thermal profiling [26], or combinations of them [14], [26]. All of these techniques assume the availability of the full details of the circuit design, in addition to the statistical distribution of gate characteristics. The expected value of the characteristics of the IC is used as a reference model for detecting Trojans.

Several techniques combining invasive and noninvasive detection techniques have also been proposed. They attempt to model the structure of the IC by invasively testing a few, and then use the models in combination with noninvasive tests to detect Trojans [21].

Defenses against malicious 3PIP and insider attacks include self-monitoring [20] and static verification [19]. Trojans can also be prevented from activation by breaking the sequence/timing of events and by scrambling inputs supplied to the 3PIPs [18]. The integrator and the 3PIP vendor can also agree on a set of security properties which the integrator can verify [17].

C. Metrics

1) Probability of detection: It is defined as the ratio of the number of Trojans detected by the technique to the total number of Trojans in the design. This metric equals to one minus the false positive rate [15], [16]. 2) Probability of false alarm: It is defined as the ratio of the number of Trojan-free designs that are incorrectly classified as Trojan to the number of Trojan-free designs [15], [16]. 3) The amount of time required to detect Trojans is important. In the case of Trojans inserted in the foundry, this time is often reported in terms of the number of applied test patterns. For 3PIP Trojans, this time is reported as the number of required clock cycles.

Table 2 Scenarios for IP Piracy and IC Overbuilding. Obfuscation (O), Watermarking (W), Fingerprinting (F), and Metering (M) Are the Defenses

Scenario	3PIP Vendor	SoC Integrator	Foundry	User
1	O+W+F+M ⋆	•	–	–
2	O+W+F+M ⋆	–	•	–
3	–	O+W+F+M ⋆	•	–

III. IP PIRACY AND IC OVERBUILDING

An attacker with access to an IP or an IC can steal and claim ownership and/or can overbuild and sell them illegally [8], [27].

A. Threat Models

Table 2 illustrates the threat. In scenario 1, the attacker in the integration house may pirate the 3PIP or use more than the licensed number of 3PIP instances. In scenario 2, the attacker in the foundry may pirate the 3PIP after extracting it from the layout of the design. In scenario 3, the attacker in the foundry may pirate the IC design and/or overbuild.

B. State-of-the-Art Defenses

Five methods have been developed to thwart piracy and overbuilding: obfuscation, watermarking, fingerprinting, metering, and split manufacturing. In scenarios 1 and 2, the 3PIP vendor may protect his IP by obfuscating it, or by embedding his watermark, or by inserting a separate watermark in each instance of the IP (also called a fingerprint). In scenario 3, the integrator may obfuscate or embed his watermark or fingerprint the design before delivering it to the foundry.

1) Watermarking: A designer's signature is embedded into the design artifact [28]. The designer can later reveal the watermark and claim ownership of an IC/IP. Watermarks may include addition of black-hole states to the finite state machine (FSM) [29], addition of secret constraints during high level [30], logic and physical synthesis [31], and field-programmable gate array (FPGA) design [32].

Graph partitioning has found many applications in the IC design process: system design, behavioral synthesis, gate-level synthesis, physical design, packaging, and testing [33], [34]. One may encode the watermark as constraints during graph partitioning. For instance, one case

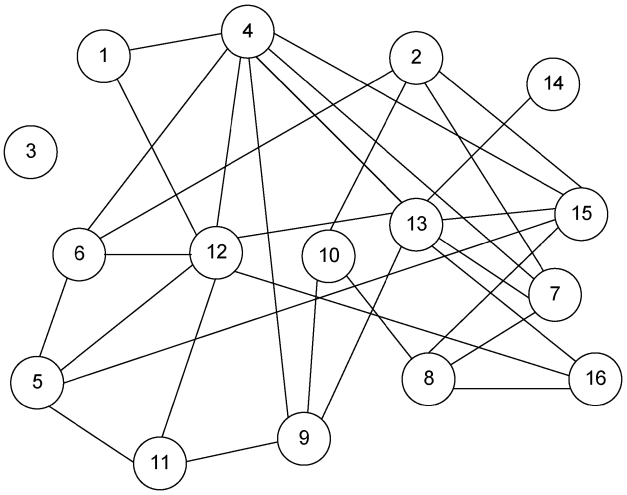

Fig. 4. *Motivational example for IP watermarking based on graph partitioning. Source: [33].*

constrains a set of nodes to be in the same partition. Alternately, a watermark can constrain the number of edges (edge cuts) spanning the partitions.

Consider embedding a watermark in the graph shown in Fig. 4. This graph has 16 nodes and 31 edges. One watermark may constrain pairs of vertices to be in the same partition. The number of possible watermarking solutions for different number of pairs and the quality of the corresponding solutions are depicted in Fig. 5. While there is only one solution for an edge-cut value of 9 (and hence this is not a good watermark constraint), there are 37 different solutions for an edge-cut value of 13. There is a delicate tradeoff between the number of possible solutions and the output quality that should be carefully considered.

A watermark should be: 1) unobtrusive, i.e., it should be oblivious to the functionality of the circuit; 2) robust, i.e., it should be extremely difficult to remove; 3) unambiguous, i.e., it should yield conclusive proof of ownership; and 4) universal, i.e., it should be applicable to all designs [30], [35].

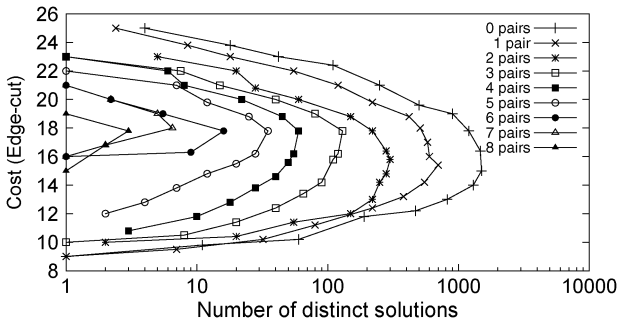

Fig. 5. *Watermarking: Number of possible watermarks versus quality of solutions for the graph when the following pairs of vertices are merged together: (16, 14), (6, 2), (16, 4), (9, 8), (5, 16), (9, 4), (11, 10), (9, 4). Source: [33].*

2) Fingerprinting: It helps the defender to track the source of piracy by embedding the signature of the buyer (for instance, his public key) along with the watermark of the designer [36]. When challenged, the designer can reveal the watermark to claim the ownership and the buyer's signature to reveal the source of piracy. For example, the power, timing, or thermal fingerprint of an IC is revealed on applying a set of input vectors.

Similar to watermarking, fingerprinting can also be applied during high-level, logic, and physical synthesis [36]. A technique that employs Kolmogorov–Smirnov statistical test for matching two probability distribution to identify whether a particular chip is fabricated at a particular foundry has been recently proposed in [37]. Another possibility is to use fingerprints derived from the static random access memory cells in the IC [38]. The recent Defense Advanced Research Projects Agency (DARPA) Supply Chain Hardware Integrity for Electronics Defense (SHIELD) program aims at uniquely (and irremovably) identifying chips, but the effectiveness of the approach is yet to be seen [39], [40].

A possible preventive measure of piracy is to register the authentic IC fingerprints using physical random functions (PUFs) and then match the dubious ICs against the PUF fingerprint database using efficient security protocols [41]. PUFs are physical functions that map the unique variations of IC's parameters to a digital output. This approach, which also works for legacy designs without any added overhead, was first proposed in [42]. Another possibility is to use the existing fingerprints from a chip's SRAM, e.g., [38]. A comprehensive survey of PUFs can be found in [43].

3) Obfuscation: Obfuscation hides the functionality and implementation of a design by inserting additional gates into it. In one type of obfuscation, XOR/XNOR gates [8], [44] and memory elements [45] are added. The obfuscated design will function correctly only on applying the correct value to these gates and memory elements.

In another type of obfuscation, the FSM of the design is obfuscated. An FSM can be obfuscated by adding extra states and/or transitions into it. Some states in the original FSM may be replicated [46], invalid transitions between states may be added [27], [47]–[49], unused states can be utilized [29], [50], [51], or additional states with no outward transitions, referred to as black hole states, can be added [29], [50], [51]. In all these techniques, only a valid key leads to the correct functionality; an invalid key leads the design into invalid states or transitions, and maybe into black hole states where the design will be stuck. Fig. 6 shows the obfuscated controller of an example FSM.

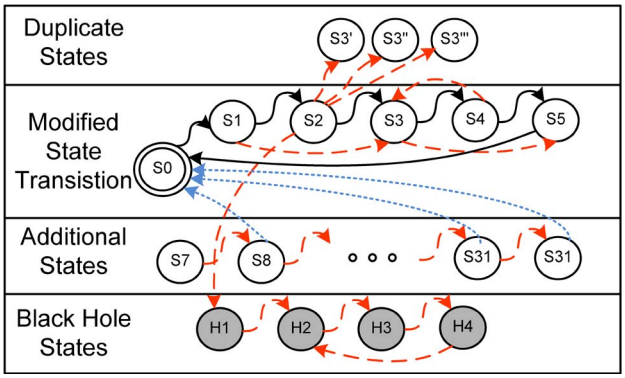

Fig. 6. ***Obfuscating a controller. Approach 1: Existing states are replicated [46]. Approach 2: State transitions are modified [27], [47]–[49]. Approach 3: Additional states are added [29], [50], [51]. Approach 4: Black-hole states are added [29], [50], [51]. S0 through S6 are the states in the original FSM. All the other states are added for obfuscation. Solid edges are the state transitions in the original FSM. Dashed edges are state transitions from an invalid state to a valid state, on applying the valid key. Dotted edges are the state transitions from a valid state to an invalid state, on applying an invalid key or when key is withdrawn.***

4) Metering: It is a set of tools, methodologies, and protocols used to track a manufactured IC. In passive metering, part of an IC's functionality is used for metering [52]. The identified ICs are matched against their record in a database. This will reveal unregistered ICs or overbuilt ICs. In active metering, parts of the IC's functionality can be only accessed, locked, or unlocked by the designer and/or IP rights owners [29]. The difference between metering and obfuscation is that while metering uses a unique unlock key per IC, obfuscation just locks the IC.

5) Split Manufacturing: The layout of the design is split into the front-end-of-line (FEOL) layers and back-end-of-line (BEOL) layers. They are then fabricated separately in different foundries. The FEOL layers consist of transistors and other lower metal layers (say $\leq$ M4) and the BEOL layers consist of the top metal layers (say $>$ M4). Post-fabrication, the FEOL and BEOL wafers are aligned and integrated together using either electrical, mechanical, or optical alignment techniques. The final ICs are tested upon integration of the FEOL and BEOL layers [53], [54]. The asymmetric nature of the metal layers facilitates split manufacturing. The top BEOL metal layers are thicker and have a larger pitch than the bottom FEOL metal layers. Hence, a designer can integrate the BEOL and FEOL layers.

Split manufacturing is practical [55]. Ideally, an attacker should not be able to retrieve the missing BEOL connections by knowing the FEOL layers [56].

C. Metrics

Metrics for watermarking include [30]: 1) collision, defined as the probability that a watermarking algorithm generates the same solution for two different signatures; and 2) degradation in the quality of the solution. Ideally, degradation should have zero effect. For example, in the graph shown in Fig. 4, though merging more number of vertices reduces the probability of collision, it increases the edge cut, degrading the quality of the solution. Fingerprinting has the same metrics as for watermarking.

Metrics for obfuscation are: 1) the number of brute force attempts required to unlock the FSM or to determine the secret key [29], [44]; 2) the Hamming distance between the outputs of an obfuscated netlist on applying an incorrect key (or configuration) and the original netlist [45], [57]; 3) the number of input patterns that produce an incorrect output on applying an incorrect key to the design [27]; and 4) the strength of the generalized point function for provable obfuscation [51].

In addition to those used for obfuscation, metering uses the following metrics: 1) the average Hamming distance between the responses to the same challenge obtained from two different ICs; ideally, this value has to be 50%; 2) the average Hamming distance between the responses to the same challenge (or a repeatedly measured fingerprint) applied at different times and environmental conditions to the same IC; ideally, this value has to be 0%; 3) nondigital measures of distances 1)/2); and 4) the number of independent IDs that can be generated.

Metrics for split manufacturing include: 1) the number of BEOL connections predicted by an attacker; and 2) the Hamming distance between the outputs of a netlist with BEOL connections predicted by an attacker and the original netlist.

IV. REVERSE ENGINEERING

RE of an IC involves 1) identifying the device technology used in it [58]; 2) extracting its gate-level netlist [9]; and/or 3) inferring its functionality [59], [60]. Several techniques and tools have been developed to reverse engineer[1] ICs [61], [62]. RE can be misused to steal and/or pirate a design, identify the device technology, or illegally fabricate the target IC. The objective of the attacker is to successfully reverse engineer a design to a desired abstraction level. He can use the known input–output pairs to verify the functional correctness of the reverse-engineered design and/or to guide RE to extract the gate-level netlist of a competitor's IP and use it in one's own IC or illegally sell it as an IP.

The objective of the attacker is to successfully reverse engineer a design to its target abstraction level. The target level can vary depending on the objective of the attacker. If the objective is to pirate the design, the target abstraction level can be either the physical design level, the gate level, or the RT level. If the goal is to insert Trojans, the

[1]These tools enable RE to collect competitive intelligence, to verify a design, to check for commercial piracy, to determine patent infringements, and to detect hardware Trojans [59], [61], [62].

Table 3 Threat Scenarios for RE. Obfuscation (O) and Camouflaging (C) Are the Defenses

Scenario	3PIP Vendor	SoC Integrator	Foundry	User
1	O ⋆	•	–	–
2	O ⋆	–	•	–
3	–	O ⋆	•	–
4	O ⋆	–	–	•
5	–	O ⋆	–	•
6	–	–	C ⋆	•
7	O ⋆	–	C ⋆	•
8	–	O ⋆	C ⋆	•

target abstraction level can be either the gate level or the RT level.

A. Threat Models

Table 3 illustrates the threat models for RE. In scenario 1, the attacker in the integration house can reverse engineer the 3PIP. The 3PIP vendor can protect his IP by obfuscating it. The foundry and the user are assumed to be untrustworthy. In scenario 2, the attacker in the foundry can extract the 3PIP from the layout of the IC. Similar to RE scenario 1, the vendor can obfuscate his IP before delivering it to the untrustworthy system-on-chip (SoC) integrator. In scenario 3, the attacker in the foundry can reverse engineer the IC. He can extract the transistor-level netlist from the layout [62], and then the gate-level netlist from it [63]. The integrator can protect the design by obfuscating it.

In scenarios 4–8, the user is the reverse engineer. He may depackage the IC, delayer it, image the layers, stitch those images, and extract the netlist. While a 3PIP vendor may obfuscate his IP (RE scenario 4), an integrator may obfuscate the layout (RE scenario 5). A trusted foundry might camouflage the layout (RE scenarios 6–8). This will provide an additional layer of defense beyond obfuscation (RE scenarios 7 and 8).

An algorithm to extract a gate-level netlist from transistors has been presented in [63]. Structural isomorphism can be used to extract the functionality of datapath modules [64]. Functionality of unknown units can be reverse engineered by performing behavioral matching against a library of components with known functionality such as adders, counters, register files, and subtractors [65]. The functionality of unknown modules can be identified by performing Boolean satisfiability analysis against a library of components with known functionality [66].

B. State-of-the-Art Defenses

Obfuscation (see Section III-B) and camouflaging can thwart RE. In scenarios 1, 2, 4, and 7, a 3PIP vendor can obfuscate his IP. In scenarios 3, 5, and 6, an SoC integrator can obfuscate his design. A trusted foundry can camouflage the layout (scenarios 6–8) and add a layer of defense beyond obfuscation.

1) Camouflaging: This is a layout-level technique to hamper image-processing-based extraction of gate-level netlist. In one embodiment of camouflaging, the layouts of standard cells are designed to look alike, resulting in incorrect extraction of the netlist. The layout of NAND cell and the layout of NOR cell look different and hence their functionality can be extracted. However, the layout of a camouflaged NAND cell and the layout of camouflaged NOR cell can be made to look identical[2] and hence an attacker cannot unambiguously extract their functionality [67]–[70].

IC camouflaging can leverage unused spaces in a chip by filling them with filler cells [71]. One can camouflage a design by using programmable standard cells [69]. Postfabrication, these cells may be programmed using a control input. One can also use dummy contacts—a dummy contact has a gap in the middle and fakes a connection between two metal layers—for camouflaging [67]. TSMC, a leading foundry, can manufacture dummy-contact-based camouflaging cells [67].

C. Metrics

Metrics for RE include: 1) percentage of gates correctly extracted from a layout [9]; 2) percentage of gates whose functionality is correctly inferred [66]; and 3) the number of signals correctly matched between the signals in the component with known functionality and the signals in the target design [65].

Metrics for camouflaging include: 1) the number of brute force attempts required to identify the functionality of camouflaged gates [72], [73]; and 2) the Hamming distance between the outputs of the original netlist and the netlist in which the functionality of camouflaged gates is assigned by the attacker [72].

V. SIDE-CHANNEL ATTACKS

Side-channel attacks exploit the leakage of secret information through a physical modality when an application is being executed on a system [10]. Side-channel attacks are powerful and have been able to break most existing important cryptographic algorithms [74].

Consider the RSA encryption algorithm which uses modular exponentiation with large exponents. An essential step in RSA encryption and decryption is computing m^e, where m is the message and e is either the pubic or private key. For an acceptable security level, m and e are required to be at least 1024-b numbers [75]. A naive approach to calculate m^e involves multiplying m by itself $e-1$ times. This approach requires $e-1$ multiplications, which is prohibitive.

To reduce the overhead, cryptographers use the square-and-multiply [75] algorithm. The pseudocode of the algorithm is shown in Algorithm 1.

[2]The contacts and vias are opaque to the RE tool as it processes the image templates for NAND and NOR standard cells.

Algorithm 1: Square and multiply algorithm for RSA exponentiation that calculates m^e

1: Input: $e = \sum_{i=0}^{t} x_i \times 2^i, b$;
2: Output = 1;
3: **for** $i = t$ DOWNTO 0 **do**
4: Output = Output^2;
5: **if** $x_i = 1$ **then**
6: Output = Output $\times$ m;
7: **end if**
8: **end for**

This algorithm first assigns 1 to the output. It then takes a pass through the bits of the exponent (e) starting from the most significant bit. For each bit in e, the output is squared; if and only if the exponent bit is equal to "1," a multiplication operation with the base value (b) is performed. For example, while the naive approach for calculating x^{1026} takes 1025 multiplications, the square-and-multiply algorithm requires only 11 multiplications. Thus, for every logic 1 in the binary representation of the exponent, the square-and-multiply algorithm takes more cycles to finish the multiplication than the one in the naive approach. An adversary can guess the exponent by measuring the amount of time that a system takes to calculate an exponentiation (at the output). This type of adversarial key extraction relies on the execution delay analysis, and is called the timing side-channel attack [76].

Power consumption [10], electromagnetic (EM) emanations [77], photonic emissions [78], and acoustic noise of the system [79] are all correlated with the exponent, and can be used to extract the secret. Another side-channel attack against RSA exploits the Chinese reminder theorem (CRT) that is typically used to speed up its computation. If an adversary induces a fault during the CRT computation, the secret information can be obtained. Fault attacks can be launched using lasers, glitches in power supplies and clocks, and X-rays [80].

An attacker can scan out the secret key, when the key storing registers are connected as a scan chain[3] [81]–[83].

It has been shown that the power/timing consumption of PUF circuits is directly correlated with the process variation that PUF secrets are based upon. Therefore, PUFs are also shown to be susceptible to side-channel attacks [84]–[87].

A. Threat Models

A realistic threat model must be developed first, and the defense should then vary depending upon the capabilities of the attacker in collecting the side-channel measurements. For example, securing a smart card is harder than securing the hardware of an offsite server against side-channel attacks; the adversary can manipulate the power and clock signals of a smart card, while he does not usually have access to power and clock systems of remote servers.

We recommend a variant of the threat model in [88]: Consider cryptographic functions of type $F : K \times M \rightarrow D$, where K is a finite set of keys, M is a finite set of messages, and D is an arbitrary set of ciphertext. The attacker is assumed to have no access to the values of k and $F(k, m)$, but he can measure/observe the characteristics of the physical implementation of F, I_F. The objective of the side-channel attack is to find the value of the secret key(s). Even if the key(s) cannot be directly found, this attack reduces the search space for the key k.

This model assumes that the attacker has the full details of the implementation of I_F and can make one side-channel measurement per invocation of function F. If more than one observations are made for each invocation, it can be easily integrated within the model by adding variables to the output space. A side channel is a function $f_{I_F} : K \times M \rightarrow O$, where O is the set of possible observations, and f_{I_F} is known to the attacker.

B. State-of-the-Art Defenses

1) Leakage Reduction: These techniques decrease the dependency between the side-channel traces of I_F and the secret information k [89]. For instance, consider the timing attack against RSA. The dependency between the timing information and the secret exponent can be reduced by performing "dummy" multiplication operations in Algorithm 1. This countermeasure incurs a 33% overhead and eliminates the leakage on timing channel.

The above countermeasure does not completely remove the threat of side-channel attacks. This is because other possible side-channel measurements (e.g., power consumption, EM, and acoustic noise of the computation unit) are still dependent on the secret multiplicands, although to a lesser extent. The dependence of side-channel information on system's inputs is a systematic property of conventional complementary metal–oxide–semiconductor (CMOS) implementations. Several CMOS implementations have been proposed to mitigate this systematic leakage. For example, information leakage from power traces can be reduced by "smoothing" the power consumption using dynamic and differential logic [90], asynchronous logic [91], current-mode logic [92], or dual-rail with precharge logic [93]. The aforementioned circuit techniques cannot fully eliminate the side-channel leakage, because perfect symmetry in power and timing traces cannot be achieved due to inevitable process variations in CMOS process. However, these defenses can effectively reduce the signal-to-noise ratio (SNR) of the side channel.

2) Noise Injection: The SNR of the measurable side-channel information can be reduced by injecting artificial noise. This makes it more difficult for an attacker to

[3]A scan chain is a design-for-test structure that connects a set of flip-flops and makes them as a shift register. During test mode, scan chains are used to convert a sequential design to a combinational design.

retrieve the secret key from the noisy side channel. In noise injection techniques, dummy circuits that consume random amount of power for each transaction, or by performing random operation independent of the secret information, are added [94] to the system.

The effects of added noise can be reduced by averaging over several samples or by applying advanced signal processing [95]. Therefore, noise injection does not provide a theoretical security but it does increase the required work of an attacker to extract the secret keys. It can be shown that decreasing the SNR of the side-channel information by a factor of K increases the number of required side-channel samples by a factor of K^2 [96]. Temporal noise is an exception to this rule and increases the required samples by a factor of K [96].

3) Key Update: Frequently updating the secret key prevents the accumulation of side-channel information by the adversary [97]. This method uses a predefined sequence of keys (e.g., the output of a pseudorandom number generator) plus synchronized timings to ensure that the sequence of keys is consistent for both communicating parties. Several methods of key update and derivation, such as key tree [97], have been proposed.

If an estimate of the maximum information leakage rate per transaction (L_{MAX}) were given, the keys should be updated before the amount of leaked information breaches a predefined level [95]. Unfortunately, to the best of our knowledge, there is still no reliable method of directly estimating L_{MAX} [98]. Instead, researchers explore new cryptographic primitives that have a theoretically proven bounded side-channel leakage per iteration [99]–[101].

4) Side-Channel-Resistant PUFs: Due to effectiveness of side-channel attacks against PUFs, it is imperative that circuit countermeasures, as proposed in [85], be used in future implementations. These countermeasures mitigate the correlation between the secret information and the measurable circuit delay/power consumption.

5) Secure Scan Chains: In a secure scan approach, mirror key registers are used in sensitive parts of the circuits [102]. These registers block unauthorized access to value of sensitive registers in the test mode of operation. In another approach, scan chains are divided into smaller subchains and access to them for regular users is randomized [82].

An adversary can combine the information leaked from several side channels to increase the effectiveness of the attack [74]. Likewise, several countermeasures, either in software or hardware, may be combined to increase the resiliency of the system against these attacks. In general, security experts advise against implementing security applications from scratch, and recommend leveraging open-source hardware or software implementations [103].

C. Metrics

The two key metrics of side-channel attacks are: 1) the amount of secret information that is vulnerable; and 2) the number of samples from side channels needed to extract the secret information. Several metrics exist to quantify the amount of information leaked. An information-theoretic measure of side-channel leakage is quite desirable [88]. However, calculating a tight upper bound of information leakage may not be practical. Metrics have been proposed [98], [104], [105] to indirectly model and estimate the side-channel information leakage. Side-channel vulnerability factor (SVF) gauges the difficulty of finding the secret information from side channels [98]. SVF quantifies the correlation between the secret information and the patterns in the side-channel time trace. SVF is a system-level metric that can be applied to all types of side channels.

VI. COUNTERFEITING

A counterfeit semiconductor component is an illegal forgery or imitation of the original component.[4] Counterfeiting is often performed by one of the many entities in the semiconductor supply chain, including new product vendors or secondary (recycled) IC vendors. In recent years, because of technological advances in 3-D packaging, fake ICs are hard to distinguish from the real ones.

Because of counterfeiting, the suppliers of the original components suffer loss. The poor performance of fake products, which are commonly lower quality or older generations of the original product, adversely impacts the overall system performance/reliability. It also harms the reputation of the authentic provider. Such fake products could potentially tamper the performance of weapons, airplanes, cars, or other critical applications that use them [11]. Although the common incentive for selling fake ICs is financial, the ease of inserting intentional hardware Trojans or spyware in fake ICs makes them a real security threat for the whole system which would eventually integrate the fake components.

A. Threat Models

Table 4 illustrates the counterfeit IC threat models. In scenario 1, defective ICs, i.e., those which failed the manufacture-time testing and have been discarded, are used in consumer products [11]. An untrustworthy entity at the test facility can be the source of leaking defective ICs. In scenario 2, a dishonest entity in the IC supply chain mislabels a product and sells it as another IC potentially through a vendor [11]. Scenario 3 is similar to scenario 2 except for the following difference: While the designer

[4]Note that we make a distinction between the pirated/overbuilt ICs and fake ICs (although a clear distinction may be blurry in certain scenarios). IC piracy and overbuilding entail making ICs by illegally copying or stealing an authentic blueprint/IC during one of the design, synthesis, or production phases.

Table 4 Threat Scenarios for Counterfeiting. The Shield Represents a Defender, and "—" Indicates an Untrustworthy Entity

Scenario	Design	Test	Recycling/ Repackaging	PCB Assembly
1	–	●	–	⋆
2	⋆	–	●	–
3	–	–	●	⋆

employs proactive techniques to prevent counterfeiting in scenario 2, the assembly use reactive techniques to detect counterfeiting in scenario 3 [11].

B. State-of-the-Art Defenses

1) Hardware Metering and Auditing: Hardware metering is a set of tools, methodologies, and protocols that enable postfabrication tracking of the manufactured ICs. Hardware metering may be passive, or active. In passive metering, part of the functionality of each IC can be specifically identified and used for metering, even for the ICs coming from the same mask [52]. The identified ICs may be matched against their record in a preformed database that could reveal unregistered ICs or overbuilt ICs (in case of collisions). In active metering, parts of the chip's functionality can be only accessed, locked (disabled), or unlocked (enabled) by the designer and/or IP rights owners, using a high level knowledge of the design. Such knowledge is typically not accessible by the foundry or other supply chain entities [106].

2) IC Fingerprints or PUFs: See Section III.

3) Device Aging Models/Sensors: IC lifetime is influenced by a variety of phenomena [11], [42], [107], [108], such as negative temperature bias instability (NBTI), hot carrier injection, and electromagnetic migration. By employing sensors in ICs to measure these phenomena, an estimate of chip lifetime can be found which would prevent counterfeiters from selling used chips as new ones. Measuring the previous usage of a device, while also detecting its authenticity, has been discussed quantitatively in [109].

4) IP Watermarking: See Section III.

C. Metrics

The metrics for hardware metering, PUF fingerprinting, and watermarking are discussed in Section III. The two new metrics relevant to counterfeiting are: 1) probability of detection is the ratio of the number of counterfeit ICs detected by the technique to the total number of counterfeit ICs [110], [111]; and 2) probability of false positive is the ratio of the number of genuine ICs that are incorrectly classified as counterfeit ICs to the number of genuine ICs [42], [110].

VII. CONCLUSION

In this paper, the threat models, state-of-the-art countermeasures, and metrics used to evaluate the defenses against hardware Trojans, IC and IP piracy, RE, side channels, and counterfeiting have been introduced. Until now, most evaluations of defenses have been informal and anecdotal. The authors believe that the metrics are an important first step in formalizing the evaluation of the strengths of defenses. Similarly, a consistent classification of threat models was not available. By organizing the threat/defense scenarios, we hope the countermeasures can be compared against each other based on the target threat model and the corresponding metrics. ■

Acknowledgment

The authors would like to thank J. Rajendran (New York University, New York, NY, USA) for his input.

REFERENCES

[1] S. Skorobogatov, "Hardware assurance and its importance to national security," 2012. [Online]. Available: http://www.cl.cam.ac.uk/sps32/secnews.html

[2] U.S. Department of Commerce, "Defense industrial base assessment: Counterfeit electronics," 2010.

[3] 112th Congress, "Inquiry into counterfeit electronic parts in the department of defense supply chain," Senate Report of the Committee on Armed Services, 2012.

[4] J. Grand, J. Applebaum, and C. Tarnovsky, "'Smart' parking meter implementations, globalism, you aka meter maids eat their young," 2009. [Online]. Available: https://www.defcon.org/images/defcon-17/dc-17-presentations/defcon-17-grand-appelbaum-tarnovsky-smart_parking.pdf

[5] "My Arduino can beat up your hotel room lock," 2012. [Online]. Available: http://demoseen.com/bhpaper.html

[6] A. Huang, "Hacking the PIC 18F1320," 2007. [Online]. Available: http://www.bunniestudios.com/blog/?page_id=40

[7] Office of the Under Secretary of Defense For Acquisition, Technology, Logistics, "Defense Science Board (DSB) study on high performance microchip supply," 2005. [Online]. Available: www.acq.osd.mil/dsb/reports/ADA435563.pdf

[8] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending piracy of integrated circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, Oct. 2010.

[9] R. Torrance and D. James, "The state-of-the-art in semiconductor reverse engineering," in *Proc. IEEE/ACM Design Autom. Conf.*, 2011, pp. 333–338.

[10] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," *Adv. Cryptol.*, pp. 388–397, 1999.

[11] F. Koushanfar *et al.*, "Can EDA combat the rise of electronic counterfeiting?" in *Proc. IEEE/ACM Design Autom. Conf.*, 2012, pp. 133–138.

[12] SEMI, "Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to IP infringement," 2008. [Online]. Available: www.semi.org/en/Press/P043775

[13] M. Rostami, F. Koushanfar, J. Rajendran, and R. Karri, "Hardware security: Threat models and metrics," in *Proc. Int. Conf. Comput.-Aided Design*, 2013, pp. 819–823.

[14] F. Koushanfar and A. Mirhoseini, "A unified framework for multimodal submodular integrated circuits trojan detection," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 1, pp. 162–174, Mar. 2011.

[15] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *IEEE Computer*, vol. 43, no. 10, pp. 39–46, Oct. 2010.

[16] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 10–25, Jan./Feb. 2010.

[17] E. Love, Y. Jin, and Y. Makris, "Proof-carrying hardware intellectual property: A pathway to trusted module acquisition," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 25–40, Mar. 2012.

[18] A. Waksman and S. Sethumadhavan, "Silencing hardware backdoors," in *Proc. IEEE Symp. Security Privacy*, 2011, pp. 49–63.

[19] M. Hicks, M. Finnicum, S. T. King, M. M. Martin, and J. M. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 159–172.

[20] C. Sturton, M. Hicks, D. Wagner, and S. T. King, "Defeating UCI: Building stealthy and malicious hardware," in *Proc. IEEE Symp. Security Privacy*, 2011, pp. 64–77.

[21] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 296–310.

[22] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic, "Power supply signal calibration techniques for improving detection resolution to hardware trojans," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2008, pp. 632–639.

[23] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," in *Proc. IEEE Int. Workshop Hardware-Oriented Security Trust*, 2008, pp. 51–57.

[24] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware Trojan horse detection using gate-level characterization," in *Proc. IEEE/ACM Design Autom. Conf.*, 2009, pp. 688–693.

[25] Y. Alkabani and F. Koushanfar, "Consistency-based characterization for IC Trojan detection," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2009, pp. 123–127.

[26] K. Hu, A. N. Nowroz, S. Reda, and F. Koushanfar, "High-sensitivity hardware trojan detection using multimodal characterization," in *Proc. IEEE Design Autom. Test Eur. Conf. Exhibit.*, 2013, pp. 1271–1276.

[27] R. Chakraborty and S. Bhunia, "HARPOON: An obfuscation-based SoC design methodology for hardware protection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 10, pp. 1493–1502, Oct. 2009.

[28] A. Kahng *et al.*, "Watermarking techniques for intellectual property protection," in *Proc. IEEE/ACM Design Autom. Conf.*, 1998, pp. 776–781.

[29] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. 16th USENIX Security Symp.*, 2007, pp. 291–306.

[30] F. Koushanfar, I. Hong, and M. Potkonjak, "Behavioral synthesis techniques for intellectual property protection," *ACM Trans. Design Autom. Electron. Syst.*, vol. 10, no. 3, pp. 523–545, 2005.

[31] A. Kahng *et al.*, "Robust IP watermarking methodologies for physical design," in Proc. IEEE/ACM Design Autom. Conf., 1998, pp. 782–787.

[32] J. Lach, W. Mangione-Smith, and M. Potkonjak, "FPGA fingerprinting techniques for protecting intellectual property," in *Proc. IEEE Custom Integr. Circuits Conf.*, 1998, pp. 299–302.

[33] G. Wolfe, J. L. Wong, and M. Potkonjak, "Watermarking graph partitioning solutions," in *Proc. IEEE/ACM Design Autom. Conf.*, 2001, pp. 486–489.

[34] C. Alpert and A. Kahng, "Recent directions in netlist partitioning," *Integration, VLSI J.*, vol. 19, no. 1–2, pp. 1–81, 1995.

[35] F. Koushanfar and Y. Alkabani, "Provably secure obfuscation of diverse watermarks for sequential circuits," in *Proc. IEEE Int. Symp. Hardware-Oriented Security Trust*, 2010, pp. 42–47.

[36] A. Caldwell *et al.*, "Effective iterative techniques for fingerprinting design IP," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 2, pp. 208–215, Feb. 2004.

[37] J. B. Wendt, F. Koushanfar, and M. Potkonjak, "Techniques for foundry identification," in *Proc. Design Autom. Conf.*, 2014, DOI: 10.1145/2593069.2593228.

[38] D. Holcomb, W. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Trans. Comput.*, vol. 58, no. 9, pp. 1198–1210, Sep. 2009.

[39] Defense Advanced Research Projects Agency (DARPA), "Supply Chain Hardware Integrity for Electronics Defense (SHIELD)," Microsystems Technology Office/MTO Broad Agency Announcement, 2014.

[40] F. Koushanfar and R. Karri, "Can the shield protect our integrated circuits?" in *Proc. Midwest Symp. Circuits Syst.*, 2014, pp. 51–57.

[41] M. Rostami, M. Majzoobi, F. Koushanfar, D. Wallach, and S. Devadas, "Robust and reverse-engineering resilient puf authentication and key-exchange by substring matching," *IEEE Trans. Emerging Topics Comput.*, vol. 2, no. 1, pp. 37–49, Mar. 2014.

[42] Y. Alkabani, F. Koushanfar, N. Kiyavash, and M. Potkonjak, "Trusted integrated circuits: A nondestructive hidden characteristics extraction approach, *Information Hiding*, ser. Lecture Notes in Computer Science, Berlin, Germany: Springer-Verlag, 2008, vol. 5284, pp. 102–117.

[43] U. Ruhrmair, S. Devadas, and F. Koushanfar, "Security based on physical unclonability and disorder *Introduction to Hardware Security and Trust*. New York, NY, USA: Springer-Verlag, 2011.

[44] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. IEEE/ACM Design Autom. Conf.*, 2012, pp. 83–89.

[45] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC piracy using reconfigurable logic barriers," *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 66–75, Jan./Feb. 2010.

[46] Y. Alkabani, F. Koushanfar, and M. Potkonjak, "Remote activation of ICs for piracy prevention and digital right management," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2007, pp. 674–677.

[47] R. Chakraborty and S. Bhunia, "RTL hardware IP protection using key-based control and data flow obfuscation," in *Proc. IEEE Int. Conf. VLSI Design*, 2010, pp. 405–410.

[48] R. Chakraborty and S. Bhunia, "Hardware protection and authentication through netlist level obfuscation," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2008, pp. 674–677.

[49] R. Chakraborty and S. Bhunia, "Security against hardware trojan through a novel application of design obfuscation," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, 2009, pp. 113–116.

[50] F. Koushanfar and G. Qu, "Hardware metering," in *Proc. IEEE/ACM Design Autom. Conf.*, 2001, pp. 490–493.

[51] F. Koushanfar, "Provably secure active IC metering techniques for piracy avoidance and digital rights management," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 51–63, Feb. 2012.

[52] F. Koushanfar, G. Qu, and M. Potkonjak, "Intellectual property metering," in *Proc. Inf. Hiding Workshop*, 2001, pp. 81–95.

[53] Intelligence Advanced Research Projects Activity (IARPA), "Trusted integrated circuits program," 2011. [Online]. Available: https://www.fbo.gov/utils/view?id=b8be3d2c5d5babbdffc6975c370247a6

[54] R. Jarvis and M. G. McIntyre, "Split manufacturing method for advanced semiconductor circuits," U.S. Patent 7 195 931, 2004.

[55] B. Hill, R. Karmazin, C. Otero, J. Tse, and R. Manohar, "A split-foundry asynchronous FPGA," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2013, DOI: 10.1109/CICC.2013.6658536.

[56] J. Rajendran, O. Sinanoglu, and R. Karri, "Is split manufacturing secure?" in *Proc. IEEE Design Autom. Test Eur. Conf. Exhibit.*, 2013, pp. 1259–1264.

[57] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Logic encryption: A fault analysis perspective," in *Proc. IEEE Design Autom. Test Eur. Conf. Exhibit.*, 2012, pp. 953–958.

[58] Chipworks, "Intel's 22-nm tri-gate transistors exposed," 2012. [Online]. Available: http://www.chipworks.com/blog/technologyblog/2012/04/23/intels-22-nm-tri-gate-transistors-exposed/

[59] Defense Advanced Research Projects Agency (DARPA), "Integrity and reliability of integrated circuits (IRIS)," 2012. [Online]. Available: http://www.darpa.mil/Our_Work/MTO/Programs/Integrity_and_ReliabilityofI ntegratedCircuits

[60] ExtremeTech, "iPhone 5 A6 SoC reverse engineered, reveals rare hand-made custom CPU, tri-core GPU." [Online]. Available: http://tinyurl.com/9yn23he

[61] Chipworks, "Reverse engineering software." [Online]. Available: http://www.chipworks.com/en/technical-competitive-analysis/resources/reverse-engineering-software

[62] Degate. [Online]. Available: http://www.degate.org/documentation/

[63] W. M. V. Fleet and M. R. Dransfield, "Method of recovering a gate-level netlist from a transistor-level," U.S. Patent 6 190 433, 1998.

[64] M. Hansen, H. Yalcin, and J. Hayes, "Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering," *IEEE Design Test Comput.*, vol. 16, no. 3, pp. 72–80, May/Jun. 1999.

[65] W. Li, Z. Wasson, and S. Seshia, "Reverse engineering circuits using behavioral pattern mining," in *Proc. IEEE Int. Symp. Hardware-Oriented Security Trust*, 2012, pp. 83–88.

[66] P. Subramanyan *et al.*, "Reverse engineering digital circuits using functional analysis," in *Proc. IEEE Design Autom. Test Eur. Conf. Exhibit.*, 2013, pp. 1277–1280.

[67] SypherMedia, "Syphermedia library circuit camouflage technology." [Online]. Available: http://www.smi.tv/solutions.htm

[68] J. P. Baukus, L. W. Chow, R. P. Cocchi, and B. J. Wang, "Method and apparatus for camouflaging a standard cell based integrated circuit with micro circuits and post processing," U.S. Patent 2012 0 139 582, 2012.

[69] J. P. Baukus, L. W. Chow, R. P. Cocchi, P. Ouyang, and B. J. Wang, "Building block for a secure CMOS logic cell library," U.S. Patent 8 111 089, 2012.

[70] J. P. Baukus, L. W. Chow, and W. Clark, "Integrated circuits protected against reverse engineering and method for fabricating the same using an apparent metal contact line terminating on field oxide," U.S. Patent 2002 0 096 776, 2002.

[71] J. P. Baukus, L. W. Chow, R. P. Cocchi, P. Ouyang, and B. J. Wang, "Camouflaging a standard cell based integrated circuit," U.S. Patent 8 151 235, 2012.

[72] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proc. ACM Conf. Comput. Commun. Security*, 2013, pp. 709–720.

[73] J. Rajendran, O. Sinanoglu, and R. Karri, "VLSI testing based security metric for IC camouflaging," in *Proc. IEEE Int. Test Conf.*, 2013, DOI: 10.1109/TEST.2013.6651879.

[74] P. Rohatgi, "Improved techniques for side-channel analysis, *Cryptographic Engineering*. New York, NY, USA: Springer-Verlag, 2009, pp. 381–406.

[75] C. Paar, J. Pelzl, and B. Preneel, *Understanding Cryptography: A Textbook for Students and Practitioners*. New York, NY, USA: Springer-Verlag, 2010.

[76] F. Koeune and F.-X. Standaert, "A tutorial on physical security and side-channel attacks" *Foundations of Security Analysis and Design III*. Berlin, Germany: Springer-Verlag, 2005, pp. 78–108.

[77] P. Rohatgi, "Electromagnetic attacks and countermeasures," *Cryptographic Engineering*. Berlin, Germany: Springer-Verlag, 2009, pp. 407–430.

[78] A. Schlösser, D. Nedospasov, J. Krämer, S. Orlic, and J.-P. Seifert, "Simple photonic emission analysis of AES," *J. Cryptogr. Eng.*, vol. 3, no. 1, pp. 3–15, 2013.

[79] D. Genkin, A. Shamir, and E. Tromer, "RSA key extraction via low-bandwidth acoustic cryptanalysis," Cryptology ePrint Archive, Rep. 2013/857, 2013.

[80] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan, "The sorcerer's apprentice guide to fault attacks," *Proc. IEEE*, vol. 94, no. 2, pp. 370–382, Feb. 2006.

[81] B. Yang, K. Wu, and R. Karri, "Scan based side channel attack on dedicated hardware implementations of data encryption standard," in *Proc. Int. Test Conf.*, 2004, pp. 339–344.

[82] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing designs against scan-based side-channel attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 4, no. 4, pp. 325–336, Oct.-Dec. 2007.

[83] M. Agrawal, S. Karmakar, D. Saha, and D. Mukhopadhyay, "Scan based side channel attacks on stream ciphers and their counter-measures," in *Proc. INDOCRYPT*, 2008, pp. 226–238.

[84] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Side-channel analysis of PUFs and fuzzy extractors," *Trust and Trustworthy Computing*. New York, NY, USA: Springer-Verlag, 2011, pp. 33–47.

[85] U. Rührmair *et al.*, "Power and timing side channels for PUFs and their efficient exploitation," Cryptology ePrint Archive, Rep. 2013/851, 2013.

[86] D. Karakoyunlu and B. Sunar, "Differential template attacks on PUF enabled cryptographic devices," in *Proc. Int. Workshop Inf. Forensics Security*, 2010, DOI: 10.1109/WIFS.2010.5711445.

[87] A. Mahmoud, U. Rührmair, M. Majzoobi, and F. Koushanfar, "Combined modeling and side channel attacks on strong PUFs," Cryptology ePrint Archivetx, Rep. 2013/632, 2013. [Online]. Available: http://eprint.iacr.org/

[88] B. Kopf and D. Basin, "An information-theoretic model for adaptive side-channel attacks," in *Proc. ACM Conf. Comput. Commun. Security*, 2007, pp. 286–296.

[89] P. Rakers, L. Connell, T. Collins, and D. Russell, "Secure contactless smartcard ASIC with DPA protection," *J. Solid-State Circuits*, vol. 36, no. 3, pp. 559–565, Mar. 2001.

[90] K. Tiri, M. Akmal, and I. Verbauwhede, "A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards," in *Proc. Eur. Solid-State Circuits Conf.*, 2002, pp. 403–406.

[91] S. Moore, R. Anderson, R. Mullins, G. Taylor, and J. J. Fournier, "Balanced self-checking asynchronous logic for smart card applications," *Microprocess. Microsyst.*, vol. 27, no. 9, pp. 421–430, 2003.

[92] F. Mace, F.-X. Standaert, I. Hassoune, J.-D. Legat, and J.-J. Quisquater, "A dynamic current mode logic to counteract power analysis attacks," in *Proc. Int. Conf. Design Circuits Integr. Syst.*, 2004, pp. 186–191.

[93] M. Stanojlovic and P. Petkovic, "Strategies against side-channel-attack," in *Proc. Small Syst. Simul. Symp.*, 2010, pp. 86–89.

[94] M. Joye, "Basics of side-channel analysis," *Cryptographic Engineering*. Berlin, Germany: Springer-Verlag, 2009, pp.365–380.

[95] P. Kocher, J. Jaffe, B. Jun, and P. Rohatgi, "Introduction to differential power analysis," *J. Cryptogr. Eng.*, vol. 1, no. 1, pp. 5–27, 2011.

[96] C. Clavier, J.-S. Coron, and N. Dabbous, "Differential power analysis in the presence of hardware countermeasures," *Cryptographic Hardware and Embedded Systems*, vol. 1965. Berlin, Germany: Springer-Verlag, 2000, pp. 252–263, ser. Lecture Notes in Computer Science.

[97] P. C. Kocher, "Leak-resistant cryptographic indexed key update, " U.S. Patent 6 539 092, 2003.

[98] J. Demme, R. Martin, A. Waksman, and S. Sethumadhavan, "Side-channel vulnerability factor: A metric for measuring information leakage," in *Proc. IEEE Int. Symp. Comput. Architect.*, 2012, pp. 106–117.

[99] J. Katz and V. Vaikuntanathan, "Signature schemes with bounded leakage resilience," *Advances in Cryptology*, vol. 5912. Berlin, Germany: Springer-Verlag, 2009, pp. 703–720, ser. Lecture Notes in Computer Science.

[100] Y. Yu, F.-X. Standaert, O. Pereira, and M. Yung, "Practical leakage-resilient pseudorandom generators," in *Proc. ACM Conf. Comput. Commun. Security*, 2010, pp. 141–151.

[101] F.-X. Standaert, O. Pereira, Y. Yu, J.-J. Quisquater, M. Yung, and E. Oswald, "Leakage resilient cryptography in practice," *Towards Hardware-Intrinsic Security*. Berlin, Germany: Springer-Verlag, 2010, pp. 99–134, ser. Information Security and Cryptography.

[102] B. Yang, K. Wu, and R. Karri, "Secure scan: A design-for-test architecture for crypto chips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 10, pp. 2287–2293, Oct. 2006.

[103] D. Boneh, "Cryptography I," 2013. [Online]. Available: https://class.coursera.org/crypto-007/class/index.

[104] L. Domnitser, N. Abu-Ghazaleh, and D. Ponomarev, "A predictive model for cache-based side channels in multicore and multithreaded microprocessors," *Computer Network Security*, vol. 6258. Berlin, Germany: Springer-Verlag, 2010, pp. 70–85, ser. Lecture Notes in Computer Science.

[105] J.-S. Coron, P. Kocher, and D. Naccache, "Statistics and secret leakage," *Financial Cryptography*, vol. 1962. Berlin, Germany: Springer-Verlag, 2001, pp. 157–173, ser. Lecture Notes in Computer Science.

[106] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. USENIX Security Symp.*, 2007, pp. 291–306.

[107] V. Huard, M. Denais, and C. Parthasarathy, "NBTI degradation: From physical mechanisms to modelling," *Microelectron. Reliab.*, vol. 46, no. 1, pp. 1–23, 2006.

[108] K. Chatterjee and D. Das, "Semiconductor manufacturers' efforts to improve trust in the electronic part supply chain," *IEEE Trans. Compon. Packag. Technol.*, vol. 30, no. 3, pp. 547–549, Sep. 2007.

[109] S. Wei, A. Nahapetian, and M. Potkonjak, "Quantitative intellectual property protection using physical-level characterization," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1722–1730, Nov. 2013.

[110] K. Huang, J. Carulli, and Y. Makris, "Parametric counterfeit IC detection via support vector machines," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, 2012, pp. 7–12.

[111] X. Zhang, K. Xiao, and M. Tehranipoor, "Path-delay fingerprinting for identification of recovered ICs," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, 2012, pp. 13–18.

ABOUT THE AUTHORS

Masoud Rostami received the M.S. degree in electrical engineering from Rice University, Houston, TX, USA, in 2010, where he is currently working toward the Ph.D. degree in computer engineering.

His research interests include double-gate devices, hardware security, and security of implanted medical devices.

Farinaz Koushanfar received the Ph.D. degree in electrical engineering and computer science and the M.A. degree in statistics, both from University of California Berkeley, Berkeley, CA, USA, in 2005.

She is currently an Associate Professor with the Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA. Her research interests include adaptive and low-power embedded systems design, hardware security, and design intellectual property protection.

Prof. Koushanfar is a recipient of several awards and honors including the Presidential Early Career Award for Scientists and Engineers (PECASE), the ACM SIGDA Outstanding New Faculty Award, the NAS Kavli Foundation fellowship, and the young faculty (or CAREER) awards from the U.S. Army Research Office (ARO), the U.S. Office of Naval research (ONR), the Defense Advanced Research Projects Agency (DARPA), and the National Science Foundation (NSF).

Ramesh Karri received the Ph.D. degree in computer science from the University of California at San Diego, La Jolla, CA, USA, in 1993.

He is a Professor of Electrical and Computer Engineering at the Polytechnic Institute of New York University, Brooklyn, NY, USA. He is the Area Director for cyber security of the NY State Center for Advanced Telecommunications Technologies at NYU-Poly; Hardware Security Lead of the Center for Research in Interdisciplinary Studies in Security and Privacy (CRISSP, http://crissp.poly.edu/), cofounder of TrustHub (http://trust-hub.org/), and organizer of the annual red team blue team event at NYU, the Embedded Systems Security Challenge (http://esc.isis.poly.edu). His research interests include trustworthy integrated circuits (ICs) and processors, high-assurance nanoscale IC architectures and systems, very large scale integration (VLSI) design and test, and interaction between security and reliability. He has over 150 journal and conference publications in these areas.

Prof. Karri was the recipient of the Humboldt Fellowship and the National Science Foundation (NSF) CAREER Award, and Best Student Paper Awards at the 2013 ACM Conference on Computer and Communications Security, the 2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, and the 2011 IEEE VLSI Design Conference.