The Cat and Mouse in Split Manufacturing

Yujie Wang, *Member, IEEE*, Pu Chen, Jiang Hu, *Fellow, IEEE*, Guofeng Li, and Jeyavijayan Rajendran, *Member, IEEE*

***Abstract*— Split manufacturing of integrated circuits eliminates vulnerabilities introduced by an untrusted foundry by manufacturing only a part of the target design at an untrusted high-end foundry and the remaining part at a trusted low-end foundry. Most researchers have focused on attack and defenses for hierarchical designs and/or use a relatively high-end trusted foundry, leading to high cost. We propose an attack and defense for split manufacturing for flattened designs. Our attack uses a network-flow model and outperforms previous attacks. We also develop two defense techniques using placement perturbation—one using physical design information and the other using logical information—while considering overhead. The effectiveness of our techniques is demonstrated on benchmark circuits.**

***Index Terms*— Hardware security, placement perturbation, split manufacturing.**

I. Introduction

A. Motivation

THE cost of owning and maintaining a state-of-the-art semiconductor manufacturing facility has become enormously expensive, even several billion dollars [1]. Consequently, only high-end commercial foundries now manufacture high-performance, mixed system integrated circuits (ICs), especially at the advanced technology nodes [2]. Without the economies of scale, many of the design companies cannot afford to own and to acquire expensive foundries; hence, outsourcing their fabrication process to these "one-stop-shop" foundries becomes a necessity. Globalization of IC production flow has reduced design complexity and fabrication cost, but it has introduced several security vulnerabilities [3]. An attacker anywhere in the IC supply chain can perform the following attacks: reverse engineering, malicious circuit insertion, counterfeiting, and intellectual property (IP) piracy [2], [4]–[8]. Due to these attacks, the semiconductor industry loses billions of dollars annually [9]. This is because designers have no control over their design in this distributed supply chain, and more importantly, current electronic design automation (EDA) tools do not consider security as a design objective.

Split manufacturing of ICs reduces vulnerabilities introduced by an untrusted foundry by manufacturing only the front-end-of-line (FEOL) layers at an untrusted high-end foundry and the back-end-of-line (BEOL) layers at a trusted low-end foundry [2], [10]–[13]. An attacker in the untrusted foundry has access only to an incomplete design, i.e., the FEOL but not the BEOL. Thus, he can neither pirate nor insert Trojans into it. Recently, researchers have successfully fabricated split-manufactured designs with ∼0% faults and 5% performance overhead [11], [12], [14], [15], including a 1.3-million-transistor asynchronous field-programmable gate array (FPGA) [15]. Moreover, research has shown that split manufacturing can help to improve yield [14]. Although promising and feasible, split manufacturing still faces the following challenges.

Challenge 1: Naïve split manufacturing alone does not ensure security. An attacker can use heuristics of physical design tools to undermine the security offered by split manufacturing, as demonstrated in [10].

Challenge 2: Defense techniques usually incur timing overhead (TO), power overhead (PO), or area overhead. Hence, one needs to minimize overhead while satisfying the security objective. However, current physical design tools do not ensure this criterion.

B. Threat Model

The objective of the attacker is to retrieve the missing BEOL connections from the FEOL connections. Since the attacker is in the FEOL foundry, he has access to the technology library. Consequently, he can obtain the following information about logic gates: layout structure, delay, capacitance load, and wire capacitance. Based on this information, an attacker can reverse engineer the FEOL components and, thereby, obtains the incomplete gate-level netlist (this netlist lacks the BEOL information). For this purpose, he can use existing reverse-engineering tools [6]. The attacker neither knows the functionality implemented by the design nor has access to an IC that performs that function.

C. Related Work

The semiconductor industry proposed split manufacturing in the early 2000s to improve yield by using only defect-free FEOL parts [14]. Recently, Intelligence Advanced Research Project Agency proposed split manufacturing for security [2].

Manuscript received February 6, 2017; revised June 18, 2017 and October 15, 2017; accepted December 5, 2017. Date of publication January 16, 2018; date of current version April 24, 2018. This work was supported in part by NSF under Grant CCF-1618824 and in part by SRC under Grant 2016-TS-2688 and Grant 2016-TS-2689. The work of Y. Wang was supported by the China Scholarship Council. *(Corresponding author: Yujie Wang.)*

Y. Wang is with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: wangyujie@ict.ac.cn).

P. Chen was with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77840 USA. He is now with Alexa Shopping, Amazon Web Service, Inc., Seattle, WA 98109 USA (e-mail: puchen0211@outlook.com).

J. Hu is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77840 USA (e-mail: jianghu@tamu.edu).

G. Li is with the College of Electronic Information and Optical Engineering, Nankai University, Tianjin 300071, China (e-mail: ligf@nankai.edu.cn).

J. Rajendran is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77840 USA (e-mail: jv.rajendran@tamu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2017.2787754

Split manufacturing is feasible, as several research groups have successfully demonstrated fully functional split-manufactured designs: 32-bit multiplier, data encryption standard, and static random access memory (RAM) circuits [11], [12], [16]; asynchronous FPGA [17]; and resistive RAM-based split manufacturing [18]. Split manufacturing for analog designs has been proposed [19].

An attack called proximity attack has been proposed in [10]. This attack aims to recover the missing BEOL connections using the physical proximity of the FEOL components and the heuristics of the physical design tools. To thwart this attack, a pin-swapping technique is proposed to swap the block pins in the layout such that the Hamming distance between the original design and the design recovered by proximity attack is close to 50% [10]. The disadvantages of this paper are: 1) it is applicable only to hierarchical designs, while lots of designs are flattened designs and 2) it incurs a performance overhead of 25%.

A recent work [13] describes several security metrics and defense techniques but fails to provide techniques to reduce overhead. M2 is used as the split layer to increase security, but at the expense the increased cost of the BEOL foundry [11], [12], [16].

Recently, security-driven wire-lifting was proposed to defend against proximity attacks [20]. In [20], the attacker knows the function of the circuit. In other cases, however, the attacker do not know the function and can only get information in physical level, but that work incurred up to 200% overhead.

To improve the pin-swapping defense, a security-driven circuit partitioning algorithm and a simulated annealing-based placement algorithm are proposed in [21]. However, simulated annealing is very slow for large designs and rarely used in modern cell placers.

Security-driven split manufacturing is also embraced in 3-D or 2.5-D IC designs, where different die layers can be manufactured at different foundries [22], but this technique inherently assumes that split manufacturing is secure. Researchers use circuit monitoring techniques to enhance the security of split manufacturing [23]; our technique does not need such circuitry, as we ensure security through design.

Most existing defense techniques assume that one can enhance security by splitting at M1 [11]–[13], [16]. Unfortunately, this increases the cost of the BEOL foundry. If the BEOL foundry can manufacture certain lower metal layers (e.g., M2 and M3), the attacker in the FEOL foundry obtains less information about the design—only M1 and transistors. Unfortunately, in this case, the cost of owning and maintaining the BEOL foundry increases. If the BEOL foundry can manufacture only upper metal layers (e.g., M5 and above), the attacker in the FEOL foundry obtains more information about the design—M1–M4 and transistors. Although the cost of the BEOL foundry decreases in this case, it decreases security. Thus, one needs to find the optimal tradeoff between security and capability/cost of the BEOL foundry. Our framework is compatible with any split layer, unlike the existing work [10]–[13], [16], [21]. Table I summarizes the comparison of this paper and related work.

IC camouflaging [24] and logic encryption [25], [26] are other IP protection techniques. The former protects against malicious users, and the latter protects against both malicious users and foundries but requires secret key storage. Split manufacturing is an orthogonal technique that targets malicious foundries without the requirement of keys.

D. Approach and Contributions

In this paper, we first develop an attack for flattened designs using a network-flow model. In addition to the proximity heuristic, our framework considers load capacitance constraint and dangling wire hint. Note that most of the hints described in [10] are for hierarchical designs and cannot be used for flattened designs. We then develop placement perturbation techniques to defend against proximity attack for flattened designs. These techniques consider both physical proximity and logic structures that may affect the effectiveness of the defense. Finally, we optimize the wirelength overhead (WLO) of our defense technique, though this approach can also optimize for other metrics, such as power, delay, and wire congestion.

The contributions of this paper are as follows.

1) A network-flow based attack model leveraging common design conventions from industry-standard physical design tools geared toward flattened designs (see Section II).
2) Experiments on ISCAS-85 and ITC-99 benchmark circuits to demonstrate that our attack outperforms proximity attack [10] for flattened designs by $\sim 3\times$.
3) A security-driven placement perturbation algorithm within Pareto optimization framework to ensure security while minimizing the overhead (see Section III). This paper enables a designer to control the security versus the overhead tradeoff.
4) Under the placement perturbation framework, different gate selection schemes and optimization objectives are studied.
5) Security assessment using ISCAS-85 and ITC-99 benchmarks to show the effectiveness of our defense (see Section IV).

II. Attack

In a common embodiment of split manufacturing, FEOL layers are manufactured by an offshore high-end foundry, while BEOL manufacturing and the final integration are conducted in a trusted foundry. The security risk in this scenario arises from the attacker in the offshore foundry.

The objective of the attacker is to retrieve the missing BEOL connections from the FEOL connections. Since the attacker is in the foundry, he has access to the technology library. Consequently, he can obtain the following information about logic gates: layout structure, delay, capacitance load, and wire capacitance. Based on this information, an attacker can reverse engineer the FEOL components and, thereby, obtains the incomplete gate-level netlist (this netlist lacks the BEOL information). For this purpose, he can use existing tools [27].

TABLE I

RELEVANCE TO RELATED WORK. "–" INDICATES THAT INFORMATION IS UNAVAILABLE. "H" AND "F" INDICATE APPLICABILITY TO HIERARCHICAL AND FLATTENED DESIGNS, RESPECTIVELY. "L" INDICATES WHETHER THE ATTACKER KNOWS CIRCUITS FUNCTION OR LOGIC. (RES. MEANS RESILIENCY TO PROXIMITY ATTACK)

Technique	Split layer	H/F	L	Res.	Security metric	Overhead
[2]	M4	F	No	No	–	–
[10]	M4	H	No	Yes	Hamming distance	25% (delay)
[13]	M1	F	No	Yes	# of known functions	77% (area)
[11], [12], [16]	M1	F	No	Yes	# of wrong connections	5% (delay)
[20]	–	F	Yes	Yes	# of known functions	200% (delay)
[21]	–	H	No	Yes	Hamming distance	14% (wirelength)
This work	Any layer	F	No	Yes	# of correct connections	3.3% (wirelength)
						0.27%(delay)

The attacker neither knows the functionality implemented by the design nor has access to an IC that performs the function.

An attacker has the disadvantage that the solution space can be astronomically large. If k gate output pins miss their connections, there are 2^{k^2} possible connections in the worst case. An attacker can tremendously reduce this large solution space based on the knowledge that the designer used conventional physical design tools to design the target IC, which has been proposed in [10]. An attacker can take the advantage of the following hints, which are public knowledge.

Hint 1 (Physical Proximity): Physical design tools aim to minimize wirelength, thereby improving performance and reducing power consumption. Therefore, a connection between two pins is rarely very long. Hence, an attacker will prefer to connect two pins that are close to each other rather than the ones that are far apart.

Hint 2 (Acyclic Combinational Logic Circuit): With the exception of ring oscillators, flip-flops, and latches, combinational loops are rare in a design.

Hint 3 (Load Capacitance Constraint): A gate can drive only a limited load capacitance to honor slew constraints. The maximum load capacitance of a gate can be obtained from the physical design library, which is public information. Hence, an attacker will consider only connections that will not violate the load capacitance constraints.

Hint 4 (Directionality of Dangling Wires): Physical design tools route wires from a source gate to the sink node along the latter's direction. Hence, the directionality dangling wires at lower metal layers indicate the direction of their destination cell. An attacker can disregard components in the other directions. Consider the example in Fig. 1. There is a dangling metal pointing toward gate A in the FEOL design available to the attacker. Intuitively, the missing upper metal is most likely to be connected with gate A instead of gate B.

Hint 5 (Timing Constraint): If a connection violates the timing constraints, then this connection can be excluded. An attacker can at least obtain a conservative estimate on timing constraints through educated guess on clock period.

A. Greedy Attack–Proximity Attack for Flattened Designs

The greedy attack mainly follows the proximity hint (*Hint #1*) and the acyclic combinational logic hint (*Hint #2*) [10]. Unlike in hierarchical designs [10], where each missing net has only two pins, the net in flattened designs may have multiple fan-outs, i.e., more than two pins. In the greedy attack, we iteratively connect a gate input pin to its nearest gate output pin. After each connection, we check if that connection results in a combinational loop. If a loop is found, this connection is reverted, the input pin is tried to connect with the next nearest output pin that does not result in a combinational loop. This procedure is repeated until all gate input pins are connected. At the end, if there is a dangling output (i.e., the output of a gate that is not connected to any input), we find its nearest multifan-out net and connect the nearest input pin in this net to the dangling output pin.

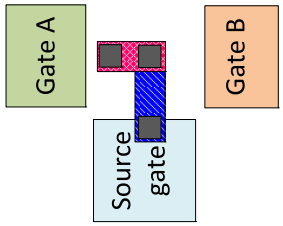


Fig. 1. Dangling wire points potential connection from the source gate toward gate A.

B. Network-Flow Attack

We describe a network-flow based attack framework that considers all the aforementioned hints (*Hints #1–#5*) in a holistic manner. This is shown by an example in Fig. 2, where the attack needs to infer the connections between output pins $\{a, b\}$ and input pins $\{1, 2, 3\}$.

The network is a directed graph $G = (V, E)$, where V is a set of vertices and E is a set of edges. The set V is composed of a set of vertices corresponding to the output pins (V_o), a set of vertices corresponding to the input pins (V_i), the source vertex (S), and the target vertex (T). The set E consists of E_{So}, edges from S to every output pin vertex, E_{oi}, edges from output pin vertices to input pin vertices, and E_{iT}, which includes edges from every input pin vertex to the target vertex. The network for Fig. 2(a) is shown in Fig. 2(b). In a network-flow solution, a certain amount of flow emerges from S, goes through the network edges, and finally arrives T. The flow through the edge $(a, i) \in E_{oi}$ infers wire connection between output pin a and input pin i.

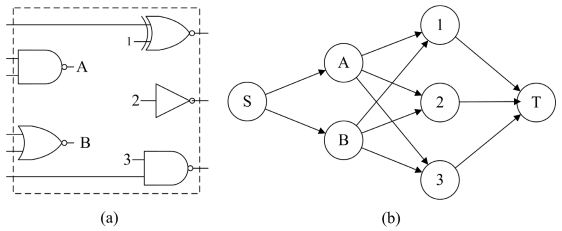


Fig. 2. (a) Circuit with missing connections. (b) Network-flow model for inferring the missing connections.

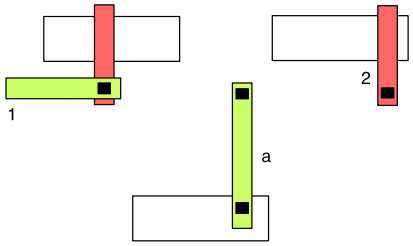


Fig. 3. Input pins 1 and 2 are in output pin a values' dangling direction. Output pin a is in pin 2 values' dangling direction but not in pin 1 values' dangling direction.

The five hints are addressed by edge construction for E_{oi}, edge capacities, edge costs, and dynamic use of the network-flow model. A necessary condition for including an edge $(a, i) \in E_{oi}$ is that output pin a is along the direction of input pin i values' dangling wire and vice versa. For the example shown in Fig. 3, edge $(a, 2)$ is included in E_{oi}, but $(a, 1)$ is not. Another condition is that the connection between a and i would not result in timing violation. We can estimate the slack at a by subtracting the arrival time (AT) at a from the required AT (RAT) at i.[1] This is an optimistic estimation without considering the delay from a to i. If this optimistic slack is less than zero, then including the delay from a to i would make the violation even worse. Then, the connection between a and i is disallowed, i.e., there is no (a, i) in E_{oi}. Sometimes the AT and RAT are not available due to wire disconnections; then, we replace AT with lower bound, which is the AT at the primary input, and replace RAT with upper bound, which is the RAT at the primary output. The estimate obtained as such provides an upper bound for the slack. By constructing E_{oi} as such, the hint of the directionality of dangling wire (*Hint #4*) and timing constraints (*Hint #5*) is followed.

The capacity $c_{S,a}$ for each edge in E_{So} is defined as the load capacitance constraint for output pin a. The capacity $c_{a,i}$ for each edge in E_{oi} is infinity. The capacity $c_{i,T}$ for each edge in E_{iT} is the input capacitance for pin i. A flow solution that satisfies the edge capacity constraints follows the hint of load capacitance constraint (*Hint #3*).

The cost $w_{a,i}$ for each edge in E_{oi} is the wirelength in connecting pins a and i. The other edge costs are set to 0. If we run min-cost flow algorithm on this network, the solution minimizes the total flow cost, which is the total wirelength for all connections. This edge cost definition addresses the proximity hint (*Hint #1*).

[1]An attacker can determine AT and RAT from the minimum operating frequency available in the design specification.

The hint of an acyclic combinational logic circuit (*Hint #2*) is difficult, if not impossible, to be handled in a one-shot network-flow solution, because a loop can be detected only after the connection solution is obtained. To solve this issue, we used an iterative network-flow approach. After connections are inferred from a network-flow solution, a circuit traversal is performed to check if any loop exists. If so, the longest inferred connection is picked. This connection must correspond to an edge in E_{oi}. Then, this edge is removed from the network, and the min-cost flow algorithm is conducted again. This procedure is repeated until no loop is detected.

In the min-cost network-flow problem, the decision variables are the flow $x_{i,j}$ going through each edge $(i, j) \in E$. Then, the problem is formally formulated as follows:

$$\text{Min} \sum_{(i,j)\in E} w_{i,j} \cdot x_{i,j} \tag{1}$$

$$\text{s.t.} \sum_{i|\ (i,j)\in E} x_{i,j} = \sum_{k|(j,k)\in E} x_{j,k}, \quad j \in V_o \cup V_i \tag{2}$$

$$\sum_{(i,T)\in E_{iT}} x_{i,T} = \sum_{(i,T)\in E_{iT}} c_{i,T} \tag{3}$$

$$\sum_{(S,i)\in E_{So}} x_{S,i} = \sum_{(i,T)\in E_{iT}} c_{i,T} \tag{4}$$

$$x_{i,j} \le c_{i,j}, \quad \forall\ (i, j) \in E. \tag{5}$$

This problem can be solved by off-the-shelf algorithms, e.g., the Edmonds–Karp algorithm [28], which can obtain the optimal solution in polynomial time.

The complexity of the Edmonds–Karp algorithm is $O(VE^2)$ [28], where V is the number of disconnected BEOL pins and E is the number of wires, as known as edges in the graph. We execute this algorithm for V times in the worst case. The run-time of network-flow attack is $O(E^2V^2)$. The proposed five hints are considered when constructing the graph to limit E. Before starting next iteration, the connected BEOL pins will be removed from the graph to reduce V. By this way, the actual run-time is much higher than the theoretical value.

III. Placement-Based Defense

A. Motivation Example and Overview

To develop provably secure split manufacturing, one needs to reinforce physical design techniques with security. Unfortunately, such approaches may have high overhead. Hence, we propose a defense technique, placement perturbation, while explicitly optimizing the overhead. This technique is compatible with conventional physical design tools.

Fig. 4 shows an example of overhead control in defense. Fig. 4(a) shows the original layout. If we perform pin swapping [10], we obtain the layout in Fig. 4(b) where the dashed lines indicate the upper metal wires, which the attackers are missing. By using the proximity hint, an attack may restore the connection as the red lines, which are wrong. Thus, the swapping, indeed, improves security. However, it increases wirelength by 70%, which is quite significant. Alternatively, one can make placement perturbations to gates A and B like in Fig. 4(c), which also causes the attacker to fail with only

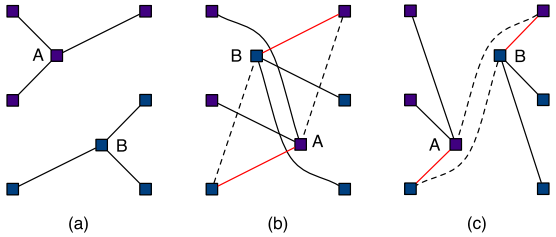


Fig. 4. (a) Original layout with small squares indicating logic gates. (b) After swapping gates A and B, dashed lines are wires missing to attackers, and red lines are connections determined by the network-flow attack, which are wrong; wirelength increases by 70%. (c) Smaller perturbation to gates A and B thwarts the attack but also decreases the WLO to 30%.

30% wirelength increase. In placement perturbation, we aim for a secure design with minimum overhead.

We describe a placement perturbation-based defense. Given a circuit design after global routing and wire layer assignment, gate locations are changed such that the proximity hint is no longer effective. The perturbation may affect the conventional design objectives, such as wirelength, timing, and power. Thus, the overhead needs to be minimized. The algorithm consists of two phases. Phase I is to select which gates to be perturbed, and Phase II is to make small placement changes to the selected gates.

B. Phase I: Gate Selection

The proposed procedure iteratively extracts a set of trees from the circuit and perturbs the locations of the gates in the trees. The reason to extract tree topology is because of its compatibility with the Pareto optimization-based placement perturbation algorithm. We propose and study two gate selection techniques: one is BEOL-driven, and the other considers logical differences among gates.

1) BEOL-Driven Gate Selection: The BEOL-driven gate selection, or tree extraction, mainly follows one principle. That is, the selected gates should be incident with BEOL wires, i.e., wires missing to attackers. If a gate only involves FEOL wires, its related wire connections are visible to attackers, and therefore, perturbation of its location does not help improve security. The BEOL-driven gate selection is an iterative procedure. At each iteration, a gate with BEOL net as its fan-in is chosen as a root node. Then, a reverse topological order traversal is performed starting from this root node. All gates incidents to BEOL wires are included in the tree during the traversal. The traversal terminates when either the primary inputs are reached, or none of its traversal front nodes are incident to BEOL wires. We make sure that the selected gates conform to tree topology. For a multifan-out gate, we only include the fan-out gate that is first encountered in the traversal. This tree extraction procedure is repeated until all gates involving BEOL wires are included.

2) Logic-Aware Gate Selection: BEOL-driven gate selection can be improved by exploiting the following two observations. First, the potential change due to incremental routing is neglected. After a placement perturbation, incremental routing must be performed to connect the gates with changed locations. It is likely that a gate that is incident to only FEOL wires becomes incident to BEOL wires after the incremental routing. Second, not all gates have the same impact on security. For some gates, a wrong connection by an attacker may have a very limited impact on the circuit output, while for some other gates the impact can be much larger.

Based on the two observations, we suggest a logic-aware gate selection method. First, we additionally take gate incident to top FEOL layers into consideration. When gates not incident to BEOL wires are considered, the number of selected gates tends to be large and so is the WLO. To overcome this drawback, we extract a treelike in the BEOL-driven gate selection but only allow a subset of the gates in the tree to be perturbed. These gates are movable gates, while the others are fixed gates in the tree.

Second, among the gates being considered, we select movable gates according to the logical difference. More specifically, we choose a gate that has the remarkable difference from its neighbor gates. In an attack, a gate is mostly likely to be confused with its neighbor gates. A wrong connection to one of its neighbor gates will cause more errors if the gate and its neighbor have a large logical difference.

Here, we specify what is counted as neighbor gate with significant logical difference (NGSLD). For two gates g_i and g_j, we can run logic simulations to estimate the probability $\Delta(g_i, g_j)$ that their output pins have opposite logic values. For a gate g_i, we look at how many of its nearby gates have a significant logic difference. More specifically, for each gate g_j within distance D from g_i, if $\Delta(g_i, g_j)$ is greater than a threshold Φ, then g_j is counted as a neighbor with significant logical difference.

Since a gate incident to BEOL wires is more likely to involve BEOL wires again after incremental routing, we choose a relatively low threshold Φ for them, which allows more of such nets being selected. For the same reason, the chance a gate originally incident to only FEOL wires has smaller chance to be incident to BEOL wires after incremental routing; we use a bigger Φ for them. This means that we select such gates only if their potential impact is very large. All gates being considered are sorted in nonincreasing order of number of NGSLD. Then, we select top $\rho\%$ of them to be as moveable gates. If a gate has a large difference from its neighbors, a wrong connection among them tends to produce big impact to the circuit outputs. Please note that $\rho\%$ affects the tradeoff between security and wirelength. A large $\rho\%$ implies that more gates are perturbed, and hence, security is improved more. On the other hand, more perturbation generally causes more WLO.

Like the BEOL-driven gate selection, the tree extraction, here, is a reverse topological order traversal from a root node. If a gate g_j is selected to be a movable gate in the tree, its fan-in gate g_i is also included even if g_i is a fixed gate. However, if all fan-in gates of g_i are not selected, then the tree traversal at g_i terminates.

The run-time complexity of tree placement is $O(nm^2 f_{max})$, where n is the number of gates in the tree and m is the candidate location for one gate and f_{max} [29]. We set the limit of tree size to less than 30 gates to reduce n. Since we aim

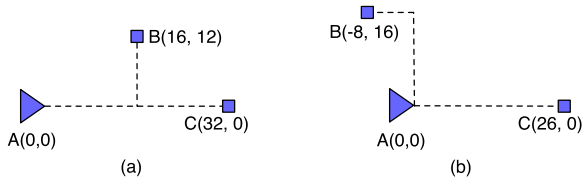


Fig. 5. Example for placement perturbation. (a) Original layout. (b) Layout after perturbation. The numbers within the parentheses indicate the X–Y coordinates of the gates.

to minimize the wirelength, the candidate location of one gate is limited within a certain, making m is also not high. In a logic-aware gate selection method, if a gate in the tree has many fan-in gates, we prefer the one that is highly movable.

C. Phase II: Placement Perturbation

1) Physical-Driven Placement Perturbation: For each extracted tree, we perturb its gate locations using a Pareto optimization approach similar to [29]. It is a bottom-up procedure from the leaf nodes to the root node of the tree. At a leaf gate g_i, we vary its location to obtain a set of candidate solutions. Please note that the candidate location must be in an empty space that can accommodate this gate. Then, the candidate solutions at leaf nodes are propagated toward their parent nodes and merged there. During this step, the clock trees are not generated, and the designer leaved some space for clock trees or other gates. Our algorithm may use that part space as candidate location. On the other hand, the algorithm places gates in a tree at one time, and so the location of each gate may be used by other gates. In the worst case, the gates in the same tree may be swapped and do not use any other space.

Each candidate solution is evaluated by its WLO and a perturbation metric. The perturbation metric dictates the placement difference from the original layout with the consideration of security. A placement has two underlying factors. The first is the spatial order, e.g., a gate is to the left or right of another gate. The second is the distance between gates or pins. From the security point of view, a relatively large pin distance in a net implies a large solution space for the attacker. A spatial order change is a more disruptive perturbation to the original design and can enhance security, considering that the spatial order in the original layout is highly optimized by conventional placement tools.

Based on these rationales, we develop a perturbation metric, which is described using the example in Fig. 5. First, we consider each source-sink pair along one axis. For source A and sink B along the y-axis, the perturbation $\pi^y_{A,B} = 4$, as the distance is increased from 12 to 16. Along the x-axis, the spatial order between A and B is flipped. Thus, we scale the distance change with a gain factor, e.g., 2. Hence, $\pi^x_{A,B} = 2 \cdot |-8-16| = 48$. For source A and sink C, there is no spatial order change, and the distance is decreased. Since such distance decrease does not change the proximity of gates, its effect is nullified in the perturbation $\pi^x_{A,C} = 0 \cdot |26-32| = 0$. The overall perturbation for this net is the summation of all these factors: $\pi_A = \pi^y_{A,B} + \pi^x_{A,B} + \pi^x_{A,C} = 52$. In general, a solution with a large perturbation is more difficult to attack.

Algorithm 1: Physical-Driven Placement Perturbation

Input : Tree extracted from layout
wirelength increase budget α
Output: Location of each gate in tree

$w_{ini} \leftarrow$ wirelength of the original tree;
Initialize every fan-in v_i of all tree leaf nodes with solution $(w_i = 0, \pi_i = 0)$;
for each *gate* v_i *in the tree in topological order* **do**
  if v_i *has fan-in* **then**
    if v_i *has 2 fan-in* v_j *and* v_k **then**
      $S \leftarrow \emptyset$;
      for each *solution* s_j *from* v_j *and* s_k *from* v_k **do**
        $S \leftarrow \{(w_j + w_k, \pi_j + \pi_k)\} \cup S$;
      end
    end
    else
      if v_i *has 1 fan-in* v_j **then**
        $S \leftarrow$ solutions from v_j;
      end
    end
  end
  $S_i \leftarrow \emptyset$;
  for each *candidate location* (x_k, y_k) **do**
    Temporarily place v_i at (x_k, y_k);
    $S_{i,k} \leftarrow \emptyset$;
    for each *solution* s_j *in* S **do**
      Obtain w_k based on (x_k, y_k) and s_j;
      Obtain π_k based on (x_k, y_k) and s_j;
      $S_{i,k} \leftarrow \{(w_k, \pi_k)\} \cup S_{i,k}$;
    end
    Prune $S_{i,k}$;
    $S_i \leftarrow S_i \cup S_{i,k}$;
  end
end
$S_{root} \leftarrow$ solutions at the root gate;
Find $s_i \in S_{root}$ that has max π_i and $w_i \leq (1 + \alpha\%) \cdot w_{ini}$;
Return Location of each gate in tree according to s_i;

Each candidate solution ψ_i is characterized by its wirelength w_i and perturbation π_i. Although the perturbation increases wirelength, they are not perfectly correlated. For the example shown in Fig. 5, the perturbation according to our metric is 52, while the wirelength increase is only 6. To avoid enumerating all cases like in a brute force approach, some inferior candidate solutions are pruned out without further propagation. For two solutions ψ_i and ψ_j, ψ_i is inferior if $w_i \geq w_j$ and $\pi_i \leq \pi_j$. At the root node, we choose the solution with the maximum perturbation while the wirelength increase is not more than $\alpha\%$, where α is the given budget. As such, α affects the tradeoff between WLO and security. The pseudocode of the defense algorithm is shown in the following. Without loss of generality, we assume that the tree is binary. If a tree is not binary, it can be converted to one by inserting pseudonodes.

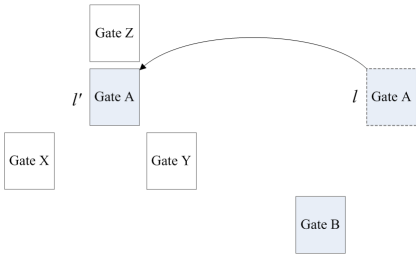


Fig. 6. Gate B is fan-out gate of gate A. Move gate A from location l to location l' where there are three critical gates.

2) Logic-Driven Placement Perturbation: Although the perturbation described in Section III-C1 strives to enhance security by making designs to be different from conventions, the geometric-based difference alone may be insufficient. For example, consider that a gate g_i is moved away from its ideal location, which is according to common design conventions. However, it is likely that the BEOL connection for this gate does not have other BEOL wires nearby. As such, this BEOL connection can be easily restored, as it is a unique option for an attacker. Even if it has another BEOL wire nearby, which is incident to another gate g_j, it is possible that the logical difference between g_i and g_j is small. Hence, even a wrong connection does not affect the circuit functionality much.

To make an improvement over the physical-driven perturbation, we proposed a logic-driven perturbation, which replaces the perturbation metric with a new metric, called weighted logical difference (WLD). If a gate g_i is temporarily placed at location l, then the corresponding WLD is defined as

$$WLD(g_i, l) = \lambda(g_i, l) \sum_{g_j \in V_l} \frac{\Delta(g_i, g_j)}{dist(g_i, g_j)} \tag{6}$$

where $\lambda(g_i, l)$ is the ratio of wirelength related to g_i at location l versus the original wirelength, V_l is the set of critical gates around l, $\Delta(g_i, g_j)$ is the logical difference between g_i and g_j, and $dist(l, g_j)$ is the Manhattan distance between location l and g_j. A candidate perturbation solution with a large WLD value is retained for exploration. The centerpiece of WLD is the logical difference $\Delta g_i, g_j$, which encourages solutions with large logical difference from its neighbors. This difference is weighted by the inverse of distance. This is because gates nearby matter more than gates that are far away.

In (6), $\lambda(g_i, l)$ is a factor to increase the chance that the wires incident to g_i to be routed through BEOL layers. In general, routers tend to use BEOL layers for long wires. A large $\lambda(g_i, l)$ value implies longer wires, which have a greater chance to be routed through BEOL layers.

The algorithm of the logic-driven perturbation is very similar to that of the physical-driven perturbation, except that the perturbation metric is replaced by WLD. The pseudocode of the logic-driven perturbation is shown in Algorithm 2.

Fig. 6 shows an example of the logic-driven placement perturbation. In this example, there is a BEOL net, for which gate A is the source, and gate B is the sink. Three other gates, X, Y, and Z, have their output nets in the BEOL layers. The original location of gate A is at l, and therefore, gate B is its nearest sink gate. Hence, an attacker can easily figure out that gate A is connected with gate B. If gate A is moved to location l', there are three other source gates close to gate A, and hence, the attack becomes much more difficult. In this case, WLD of gate A in location l' is

$$\text{WLD}(A, l') = \frac{D(B, l')}{D(B, l)} \left(\frac{\Delta(A, X)}{D(X, l')} + \frac{\Delta(A, Y)}{D(Y, l')} + \frac{\Delta(A, Z)}{D(Z, l')} \right)$$

where D indicates the distance.

Algorithm 2: Logic-Driven Placement Perturbation

Input : Tree extracted from layout
wirelength increase budget α
Output: Location of each gate in tree

$w_{ini} \leftarrow$ wirelength of the original tree;
Initialize every fan-in v_i of all tree leaf nodes with solution $(w_i = 0, \theta_i = 0)$;
for each *gate* v_i *in the tree in topological order* **do**
  if v_i *has fan-in* **then**
    if v_i *has 2 fan-in* v_j *and* v_k **then**
      $S \leftarrow \emptyset$;
      for each *solution* s_j *from* v_j *and* s_k *from* v_k **do**
        $S \leftarrow \{(w_j + w_k, \theta_j + \theta_k)\} \cup S$;
      end
    end
    else
      if v_i *has 1 fan-in* v_j **then**
        $S \leftarrow$ solutions from v_j;
      end
    end
  end
  $S_i \leftarrow \emptyset$;
  for each *candidate location* (x_k, y_k) **do**
    Temporarily place v_i at (x_k, y_k);
    $S_{i,k} \leftarrow \emptyset$;
    for each *solution* s_j *in* S **do**
      Obtain w_k based on (x_k, y_k) and s_j;
      if v_i *is driven by critical gate* **then**
        Obtain θ_k based on (x_k, y_k) and s_j;
      else
        $\theta_k = \theta_j$ in s_j
      end
      $S_{i,k} \leftarrow \{(w_k, \theta_k)\} \cup S_{i,k}$;
    end
    Prune $S_{i,k}$;
    $S_i \leftarrow S_i \cup S_{i,k}$;
  end
end
$S_{root} \leftarrow$ solutions at the root gate;
Find $s_i \in S_{root}$ that has max θ_i and $w_i \leq (1 + \alpha\%) \cdot w_{ini}$;
Return Location of each gate in tree according to s_i;

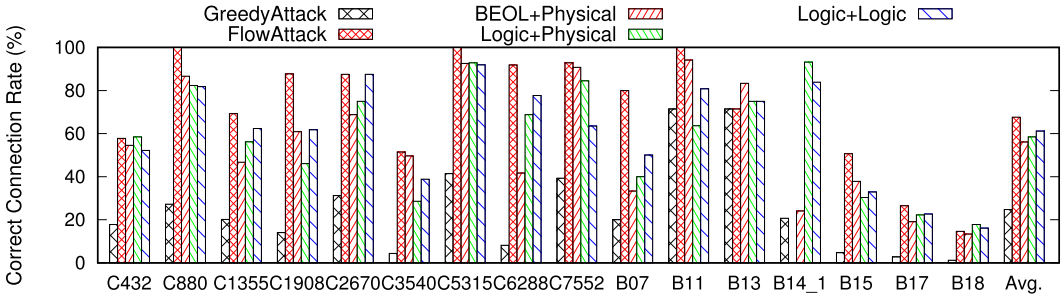


Fig. 7. Correct connection rate on performing greedy attack and the proposed network-flow model-based attack with and without defense.

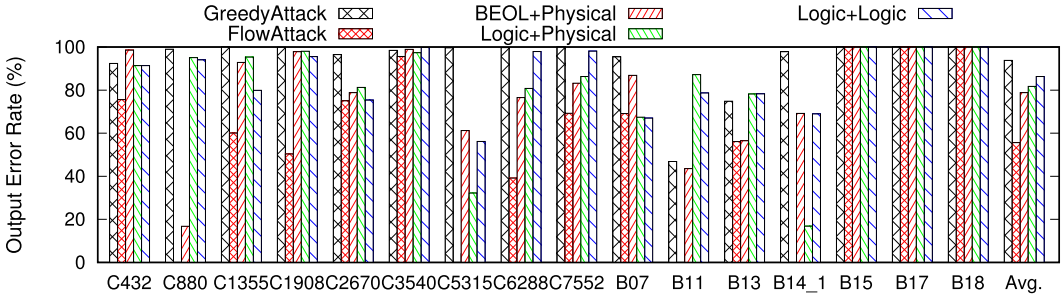


Fig. 8. Output error rate on performing greedy attack and the proposed network-flow model-based attack with and without defense.

IV. Result

A. Experimental Setup

We evaluate our techniques using ISCAS-85 combinational benchmark circuits [30] and ITC-99 benchmark [31]. Each circuit was synthesized by Synopsys Design Compiler tool. Placement and routing were performed using Cadence SoC Encounter tool for 45-nm CMOS technology. Synopsys Prime-Time static timing analysis tool was used to measure the TO for the defense techniques.

We assess the effectiveness of the attack model by identifying the number of correct connections that it makes. An attacker always tries to make as many correct connections as possible. In order to prevent an attacker from reconstructing the design correctly, the defender perturbs "enough" gates in the design while minimizing the overhead. In this paper, we measure overhead in terms of wirelength, which may affect delay, power, and congestion. In addition, we can evaluate the performance of attack and defense techniques through error rate, the number of wrong outputs produced upon applying a specific number of inputs [11]–[13], [16]. The objective of the defender is to ensure that the recovered design has a high error rate through placement perturbation. Contrarily, the objective of the attacker is to minimize the error rate of the recovered design. The error rate between the outputs of the original design and the design reconstructed using the attack was determined by applying 50 000 random input patterns.

In order to validate the effectiveness of our techniques, the following methods are compared.

1) *GreedyAttack:* Greedy attack [10] to circuits without defense.
2) *FlowAttack:* Our network-flow-based attack to circuits without defense.
3) *BEOL + Physical:* Network-flow attack to circuits with the defense of BEOL-driven gate selection and physical-driven placement perturbation.
4) *Logic + Physical:* Network-flow attack to circuits with the defense of logic-aware gate selection and physical-driven placement perturbation.
5) *Logic + Logic:* Network-flow attack to circuits with the defense of logic-aware gate selection and logic-driven placement perturbation.

B. Effectiveness of Attack and Defense

1) Effectiveness of Attack: In Figs. 7 and 8, the left two bars in each cluster compare the results of "GreedyAttack" [10] and "FlowAttack" to circuits without defense. Fig. 7 shows the results of correct connection rate, which is the percentage of BEOL connections that are successfully restored by the attacks. One can see that the correct connection rate from the "GreedyAttack" is less than 25% on average while the "FlowAttack" can improve it to 67%.

Fig. 8 shows output error rates the circuits restored by the attacks compared with the original designs. The average error rate from the "GreedyAttack" is around 90% while the "FlowAttack" can reduce it to less than 50%. Overall, the proposed "FlowAttack" is much more effective than the previous work of "GreeyAttack" [10].

2) Effectiveness of Different Defense Techniques: Figs. 7 and 8 also demonstrate the effectiveness of different defense techniques under the network-flow attack. These correspond to the rightmost three bars in each cluster. Compared with "FlowAttack" without defense, all the three proposed defense techniques can remarkably reduce correct connection rate and increase the error rate. The average correct connection

TABLE II

WLO, TO IN TERM OF PERCENTAGE INCREASE OF CRITICAL PATH DELAY, CRITICAL PATH DELAY INCREASE (Δd), AND PO AFTER DEFENSE OF "BEOL + PHYSICAL," "LOGIC + PHYSICAL," AND "LOGIC + LOGIC"

benchmarks	BEOL+Physical				Logic+Physical				Logic+Logic			
	WLO (%)	TO (%)	Δd (ps)	PO (%)	WLO (%)	TO (%)	Δd (ps)	PO (%)	WLO (%)	TO (%)	Δd (ps)	PO (%)
C432	4.5	0.49	3.9	0.17	5.57	0.24	1.9	0.44	1.68	0.21	1.7	0.17
C880	9	0.05	0.3	0.25	7.92	0.03	0.2	0.35	2.61	-0.09	-0.6	-0.05
C1355	3.76	0.57	4.1	0.52	6.5	0.42	3	0.75	1.2	0.01	0.1	0.03
C1908	10.08	1.3	13	1.1	7.99	0.23	2.3	1.1	5.71	0.39	3.9	0.45
C2670	6.2	0.27	2.1	0.29	2.85	0.27	2.1	0.29	0.25	0.03	0.2	0.05
C3540	4.79	0.28	3.2	0.53	6.64	0.02	0.2	0.36	1.86	-0.02	-0.2	0.14
C5315	2.38	-0.01	-0.1	0.19	5.96	0.08	0.7	0.67	2.53	-0.01	-0.1	0.29
C6288	4.99	0.19	5.3	0.29	3.38	0	0	0	4.35	0.67	19.1	0.1
C7552	4.54	-0.36	-4.5	0.28	3.54	-0.05	-0.6	0.35	5.41	1.77	21.9	0.56
B07	6.08	0.27	2.2	0.82	9.64	0.24	2	0.55	6.03	0	0	0.27
B11	4.9	-0.06	-0.5	0.42	7.04	-0.04	-0.3	0.34	7.58	0.06	0.5	0.56
B13	3.57	0.08	0.3	0.29	2.13	-0.37	-1.3	0.29	0.59	-0.39	-1.4	0.29
B14_1	0.79	0.22	6.6	0.14	8.39	3.16	94	0.89	2.23	0.48	14.2	0.02
B15	5.44	1.01	27.3	1.2	7.2	0.04	1	0.35	3.29	0.3	8.1	0.31
B17	1.87	0.52	21.1	0.58	4.09	0.51	20.7	0.8	4.52	0.69	28.2	0.9
B18	8.06	3.15	0.13	2.68	1.7	0.85	0.04	0.49	0.61	0	0	0.21
Avg.	5.06	0.50	5.28	0.61	5.66	0.35	7.87	0.50	3.15	0.26	5.98	0.27

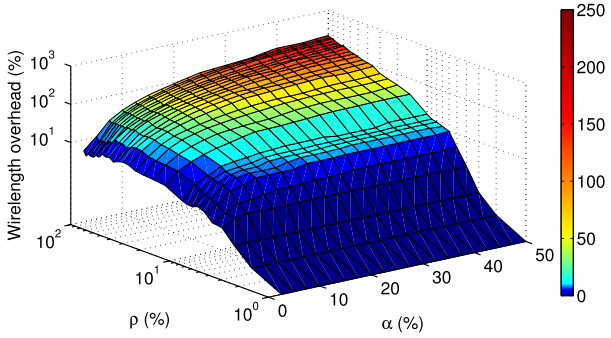


Fig. 9. Overhead of performing "Logic + Logic" on benchmarks B11.

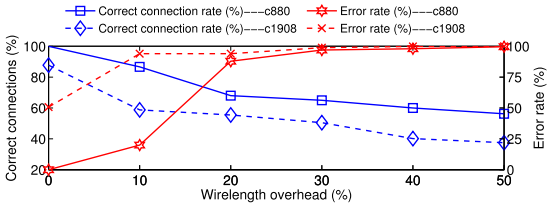


Fig. 10. Correct connection rate and error rate versus WLO for "BEOL + Physical" on circuits C880 and C1908.

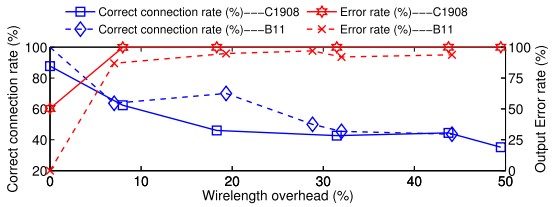


Fig. 11. Correct connection rate and error rate versus WLO for "Logic + Physical" on circuits C1908 and B11.

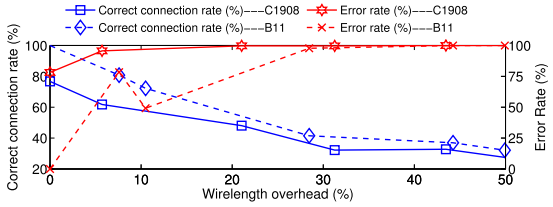


Fig. 12. Correct connection rate and error rate versus WLO for "Logic + Logic" on circuits C1908 and B11.

rates among the three defense techniques are close to each other. However, the error rate is progressively increased by considering logic effects in the defense. The best technique, "Logic + Logic," increases the average error rate from less 50% to over 80%. Please note that we constrain the WLO to be less than 10%, so that the correct connection rate cannot be suppressed to very low. The utilization of core area is 50%–60% in our test cases.

C. Overhead of Defense

One of the goals of this paper is to control the overhead due to defense, especially the WLO. Tabel II lists the WLO, TO, and PO of the proposed defense techniques on the benchmark circuits. In the placement perturbation-based defense techniques, we explicitly restrict the WLO to be within 10% of the original design. However, the subsequent routing by SoC Encounter usually causes discrepancy from the wirelength estimate in placement. Despite this, we can almost always restrict the wirelength increase to be less than 10%. The average WLO is only $3\% - -5\%$.

The average WLO of "BEOL + Physical" is 5.1%, which is lower than 5.7% from "Logic + Physical." This is because the logic-aware gate selection tends to perturb more gates. By considering the logical difference in "Logic + Logic," the average WLO is reduced to 3.2%. The simultaneous consideration of logic difference with location change steers the solution search to more secure options with less perturbation movement.

Table II also shows the TO in terms of critical path delay increase. In general, the overhead is very small, less than 0.5% and a few *ps*, as TO usually correlates with WLO. In a few cases, the TO is significant, as timing is not explicitly controlled in our placement perturbation. This problem will be solved in our future work.

The PO is shown in Table II. The PO highly correlates with WLO, since the WLO increases the load capacity. In general, the PO is less than 0.5%. Table III indicates the detailed

TABLE III

INTERNAL POWER (IP), SP, LP, AND TOTAL POWER O AFTER DEFENSE OF "BEOL + PHYSICAL," "LOGIC + PHYSICAL," AND "LOGIC + LOGIC"

benchmarks	BEOL+Physical				Logic+Physical				Logic+Logic			
	IP (%)	SP (%)	LP (%)	TP (%)	IP (%)	SP (%)	LP (%)	TP (%)	IP (%)	SP (%)	LP (%)	TP (%)
C432	0	0.39	0	0.17	0.05	0.83	0	0.44	0.03	0.41	0	0.17
C880	0.06	0.61	0	0.25	0.04	0.84	0	0.35	0.01	-0.13	0	-0.05
C1355	0.1	1.03	0	0.52	0	1.51	0	0.75	0	0.07	0	0.03
C1908	0.07	2.3	0	1.1	0.01	2.39	0	1.1	0.02	0.88	0	0.45
C2670	0.06	0.74	0	0.29	0.06	0.74	0	0.29	0	0.12	0	0.05
C3540	0.03	1.07	0	0.53	0	0.75	0	0.36	0	0.29	0	0.14
C5315	0.03	0.48	0	0.19	0.08	1.48	0	0.67	0.03	0.56	0	0.29
C6288	0.03	0.5	0	0.29	0	0	0	0	0.03	0.17	0	0.1
C7552	0.02	0.57	0	0.28	0.02	0.78	0	0.35	0.04	1.13	0	0.56
B07	0.1	1.77	0	0.82	0	1.23	0	0.55	0	0.54	0	0.27
B11	0.08	0.9	0	0.42	0.08	0.72	0	0.34	0.08	1.26	0	0.56
B13	0.02	0.83	0	0.29	0.02	0.7	0	0.29	0.02	0.63	0	0.29
B14_1	0.07	0.23	0	0.14	0.14	1.66	0	0.89	0	0.05	0	0.02
B15	0.24	2.3	0	1.2	0.06	0.67	0	0.35	0.06	0.6	0	0.31
B17	0.1	1.05	0	0.58	0.1	1.45	0	0.8	0.12	1.55	0	0.9
B18	0.33	4.63	0	2.68	0.14	0.83	0	0.49	0.04	0.37	0	0.22
Avg.	0.08	1.21	0	0.61	0.05	1.04	0	0.50	0.03	0.53	0	0.27

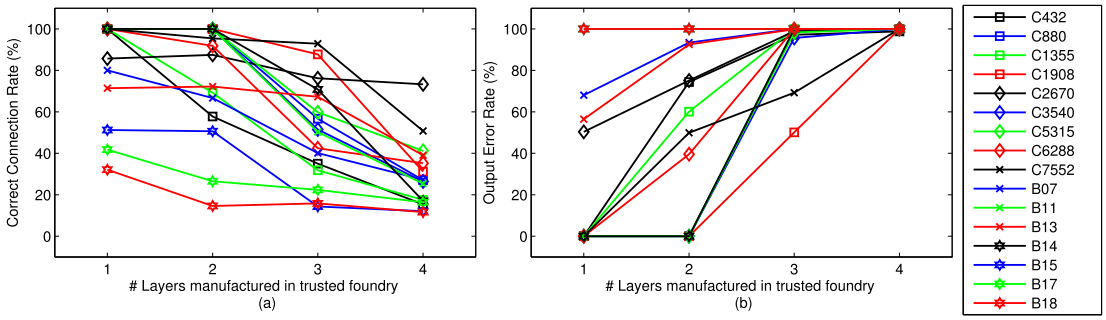


Fig. 13. Effectiveness of network-flow attacker with different layers in BEOL. (a) Correct connection rate. (b) Output error rate.

information on PO. As defined in SoC Encounter, the internal power is the activity power lead by switching inside the cell, and the switching power (SP) is the power cost in wires. The leakage power (LP) does not have any change, since we did not change the layout of the cells.

D. Security Versus Overhead Tradeoff

Besides restricting WLO, our defense techniques obtain a tradeoff between security and the overhead. In selecting the gates to be perturbed (see Section III-B2), parameter ρ decides how many gates are selected. A large ρ value implies improved security with relatively large overhead. In the placement perturbation algorithm (see Section III-C1), among the multiple candidate solutions at the root node of each tree, the one with the maximum security subject to $\alpha\%$ WLO is chosen. As such, parameter α also directly affects the tradeoff. In Fig. 9, we show the impact of these two parameters on the tradeoff for "Logic + Logic" on B11.

The tradeoff curves for the three defense techniques are shown in Figs. 10–12. The "Logic + Physical" and "Logic + Logic" are superior to "BEOL + Physical," as they can decrease the correct connection rate lower and increase the error rate with lower WLO.

E. Effect of Split Layer on Security

The layer at which the BEOL and FEOL split occur is called the split layer. If the split layer is M2 or M3, it may guarantee security, but it demands a relatively high-end BOEL facility, thus increasing its cost. Fig. 13 shows the effectiveness of the network-flow attack model with the different split layer. Since we need to align cases with different top metal layers, we indicate the split layer by the layers in BEOL. The correct connection rate is decreasing, and the output error rate is increasing while more layers in BEOL, which the proposed attack is ineffective. In Fig. 13(b), the output error rate on most cases reaches 100%. In this case, the defense is not needed.

We also test the effectiveness of the defense model with the different split layer, as shown in Fig. 14. On average, the defense mode is almost ineffective when the split layer is the top layer. That is because the attacker gets a lot of information from BEOL and it is hard to defend. When FEOL part contains two layers, the defense mode is most effective. In the case of C880 and C3540, the security achieved by our defense for a higher split layers reaches the security level achieved by a lower split layer without any defense. Thus, users who cannot support manufacturing many lower layers can obtain security using our technique.

F. Sensitivity of Parameters

Section IV-D explains how parameters α and ρ influence the overhead. In this section, we will discuss the sensitivity of parameters α and ρ to the security. The effectiveness of the defense with different parameters is shown in Fig. 15. When α is in larger number, for example in Fig. 15(e) and (f), one can easily observe that the incorrect connection rate and the

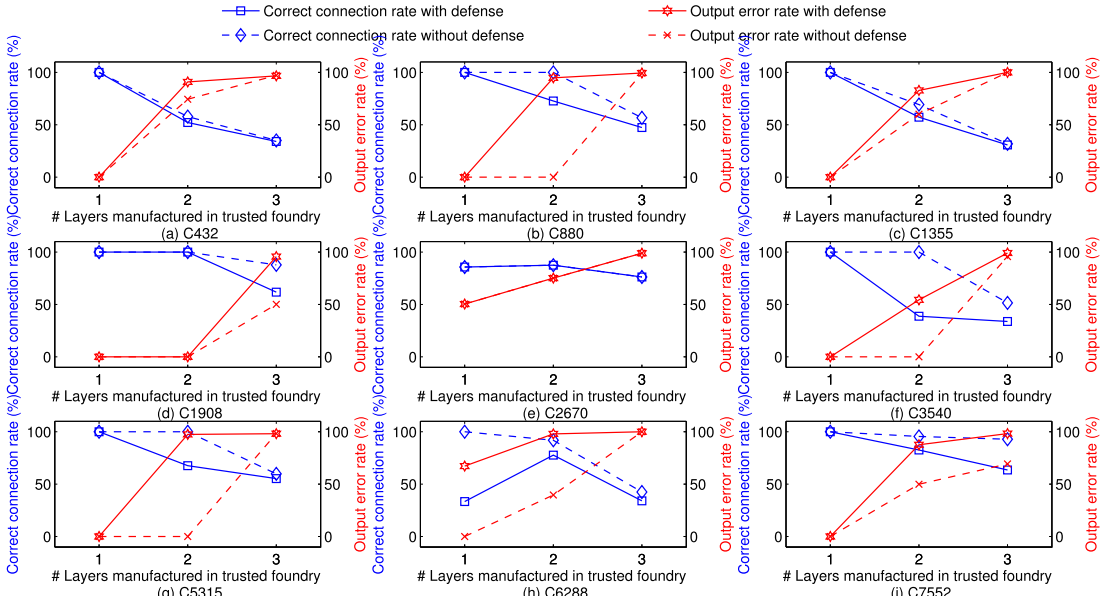


Fig. 14. Effectiveness of "Logic + Logic" with different layers in BEOL. (a) C432. (b) C880. (c) C1355. (d) C1908. (e) C2670. (f) C3540. (g) C5315. (h) C6288. (i) C7552.

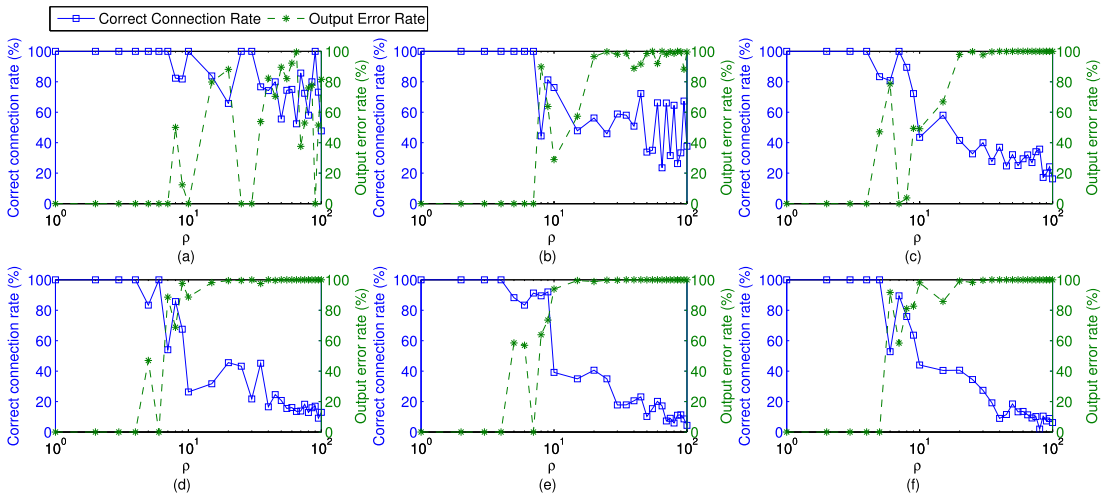


Fig. 15. Correct connection rate and output error rate on benchmark b17 when running "Logic + Logic" defense with different values of α and ρ. (a) $\alpha = 5\%$. (b) $\alpha = 10\%$. (c) $\alpha = 15\%$. (d) $\alpha = 20\%$. (e) $\alpha = 25\%$. (f) $\alpha = 30\%$.

output error rate increase as ρ increases. In Fig. 15(a) and (b), on other hand, this trend is not that clear. The parameter α indicates WLO budget when placement perturbation defense is performed. If the value of α is lower, the critical gate will not be allowed to be moved too far from its original location, because long moving distance would cause large WLO; this decreases the number of candidate placement solutions, thus impacting security. Hence, the trend is not clear when α is in lower value. This is also shown in Fig. 9, in which the surface in lower α part rises slower than that in higher α part.

We also discuss whether the gain factor in physical-driven perturbation will influence the result. Fig. 16 shows the security with a different gain factor in four test cases. In the previous result, the gain factor is 2. If we change this value, the result in Fig. 16(a)–(c) does not change obviously, only the B14_1 have some change. The value of gain factor does not affect security significantly. As shown in Fig. 17, the TO almost remains constant.

Fig. 18 shows the attack and defense result with different utilization ratios. In the lower core area utilization, the gates will be sparsely distributed, making it easy to move gates with less overhead. While, the result shows that there is no impact on security, because the high core area utilization easily results in placement perturbation.

G. Limitations and Discussion

1) Is 67% Correct Connections Enough? The greedy attack has a correct connection rate of 24% [10]. Our attack has almost tripled this rate to 68%. This is still not 100%, even though our attack deduces all the connections in C880 and C5315. This rate can be further increased by using hints at the logic level; we considered only physical design heuristics.

2) Optimizing for Other Security Metrics, e.g., Hamming Distance: In this paper, we chose correct connection rate and output error rate to measure the effectiveness of our attack and defense technique, as they are the most commonly

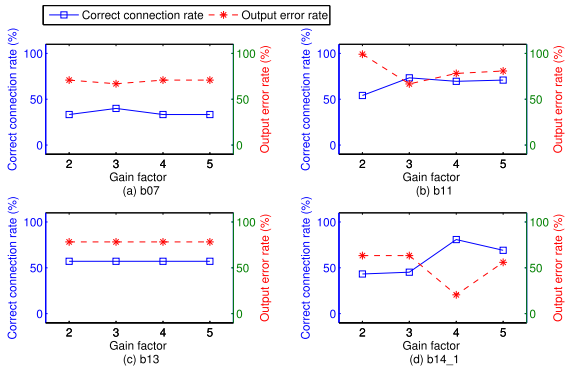


Fig. 16. Effectiveness of "Logic + Physical" with different gain factors. (a) b07. (b) b11. (c) b13. (d) b14_1.

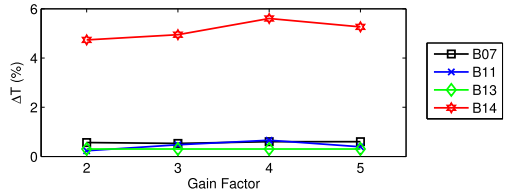


Fig. 17. Delay overhead "Logic + Physical" with different gain factors.

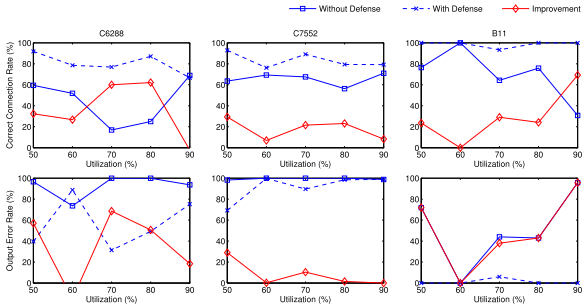


Fig. 18. Effectiveness of "Logic + Physical" with different core area utilization values.

used metrics [10]–[12], [16]. However, researchers have also used another metric, namely Hamming distance between the outputs of the original design and the design recovered by the attack [10], [21]. Our future work involves using this metric to quantify attack and defense. Furthermore, our defense framework is amenable to incorporate controllability and observability metrics into account [21]. For instance, in Algorithm 1, one can assign weights based on move distance, observability, controllability, or a combination thereof.

3) Extension to Other Threat Models: In this paper, we assumed that the attacker does not know the functionality implemented by the target design. This assumption is congruent with most work in the literature [10]–[12], [16], [21]. In an orthogonal threat model, an attacker knows the functionality implemented by the design, and he tries to identify "safe places" to insert Trojans [20]. Our framework can be adapted to this threat model: the proposed Pareto optimization approach can identify, which wires to "lift" to the BEOL connection such that it meets the required security level while minimizing overhead.

4) Lack of Formal Proof: Our attack and defense techniques hinge on the list of hints available to the attacker. The metrics we used are based on empirical results, rather than theoretical, similar to most of the existing work in the literature [10]–[12], [16], [21]. As part of future work, we aim to provide a theoretical framework for our attack and defense.

V. Conclusion

Split manufacturing, though not a universal solution for all security problems, it can protect commercial designs from rogue elements in the FEOL foundry. While the state-of-the-art attack is applicable only to hierarchical designs [10], we have proposed an attack for flattened designs, using the heuristics of physical designs tools. Our attack success rate is $\sim 3\times$ that of the state-of-the-art algorithm [10]; our attack predicts 68% of the missing BEOL connections correctly, while the state of the art predicts only 24% for flattened designs. While the placement perturbation-based defense can increase the wirelength (3.2% on average), delay (0.26% on average), and power (0.27% on average), we showed that it makes the network-flow-based attack less effective. For high-performance designs, one can easily constrain the proposed algorithm not to consider pins on the critical path. Also, our framework allows a designer to control the security versus the performance tradeoff. Apart from placement perturbation, one can also reroute the wires and use dummy wires. This paper is focused on the placement perturbation as a postprocessing for global placement. Alternatively, one can incorporate the security-driven perturbation into global placement in a similar manner, which may provide improved solution quality at the expense of increased complexity. The similar approach will be included in our future research.

References

[1] DIGITIMES Research. (2012). *Trends in the Global IC Design Service Market*. [Online]. Available: http://www.digitimes.com/news/a20120313RS400.html?chid=2

[2] Intelligence Advanced Research Projects Activity. (2011). *Trusted Integrated Circuits Program*. [Online]. Available: https://www.fbo.gov/utils/view?id=b8be3d2c5d5babbdffc6975c370247a6

[3] DARPA. (2005). *Defense Science Board (DSB) Study on High Performance Microchip Supply*. [Online]. Available: http://www.acq.osd.mil/dsb/reports/ADA435563.pdf

[4] 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.

[5] J. Rajendran, E. Gavas, J. Jimenez, V. Padman, and R. Karri, "Towards a comprehensive and systematic classification of hardware Trojans," in *Proc. IEEE Int. Symp. Circuits Syst.*, May/Jun. 2010, pp. 1871–1874.

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

[7] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," *Proc. IEEE*, vol. 102, no. 8, pp. 1207–1228, Aug. 2014.

[8] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware Trojan attacks: Threat analysis and countermeasures," *Proc. IEEE*, vol. 102, no. 8, pp. 1229–1247, Aug. 2014.

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

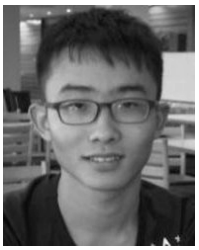
[10] J. Rajendran, O. Sinanoglu, and R. Karri, "Is split manufacturing secure?" in *Proc. IEEE/ACM Design Autom. Test Eur.*, Mar. 2013, pp. 1259–1264.

[11] K. Vaidyanathan, B. P. Das, E. Sumbul, R. Liu, and L. Pileggi, "Building trusted ICs using split fabrication," in *Proc. IEEE Symp. Hardw. Oriented Secur. Trust*, May 2014, pp. 1–6.

[12] K. Vaidyanathan, R. Liu, E. Sumbul, Q. Zhu, F. Franchetti, and L. Pileggi, "Efficient and secure intellectual property (IP) design with split fabrication," in *Proc. IEEE Symp. Hardw. Oriented Secur. Trust*, May 2014, pp. 13–18.
[13] M. Jagasivamani, P. Gadfort, M. Sika, M. Bajura, and M. Fritze, "Split-fabrication obfuscation: Metrics and techniques," in *Proc. IEEE Symp. Hardw. Oriented Secur. Trust*, 2014, pp. 7–12.
[14] R. Jarvis and M. G. McIntyre, "Split manufacturing method for advanced semiconductor circuits," U.S. Patent 7 195 931, Mar. 27, 2004.
[15] B. Hill, R. Karmazin, C. Otero, J. Tse, and R. Manohar, "A split-foundry asynchronous FPGA," in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2013, pp. 1–4.
[16] K. Vaidyanathan, B. P. Das, and L. Pileggi, "Detecting reliability attacks during split fabrication using test-only BEOL stack," in *Proc. IEEE/ACM Design Autom. Conf.*, Jun. 2014, pp. 156:1–156:6.
[17] C. T. O. Otero, J. Tse, R. Karmazin, B. Hill, and R. Manohar, "Automatic obfuscated cell layout for trusted split-foundry design," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust*, May 2015, pp. 56–61.
[18] S. Mitra, H.-S. Wong, and S. Wong. (2015). "Stopping hardware trojans in their tracks." [Online]. Available: http://spectrum.ieee.org/semiconductors/design/stopping-hardware-trojans-in-their-tracks
[19] Y. Bi, J. S. Yuan, and Y. Jin, "Beyond the interconnections: Split manufacturing in RF designs," *MDPI Electron.*, vol. 4, no. 3, pp. 541–564, 2015.
[20] F. Imeson, A. Emtenan, S. Garg, and M. V. Tripunitara, "Securing computer hardware using 3D integrated circuit (IC) technology and split manufacturing for obfuscation," in *Proc. USENIX Conf. Secur.*, 2013, pp. 495–510.
[21] Y. Xie, C. Bao, and A. Srivastava, "Security-aware design flow for 2.5D IC technology," in *Proc. ACM Int. Workshop Trustworthy Embedded Devices*, 2015, pp. 31–38.
[22] J. Valamehr *et al.*, "A 3-D split manufacturing approach to trustworthy system development," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 4, pp. 611–615, Apr. 2013.
[23] K. Xiao, D. Forte, and M. M. Tehranipoor, "Efficient and secure split manufacturing via obfuscated built-in self-authentication," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust*, May 2015, pp. 14–19.
[24] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2013, pp. 709–720. [Online]. Available: http://doi.acm.org/10.1145/2508859.2516656
[25] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," *Computer*, vol. 43, no. 10, pp. 30–38, 2010.
[26] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. IEEE/ACM Design Autom. Conf.*, Jun. 2012, pp. 83–89.
[27] *Degate*. Accessed: 2015. [Online]. Available: http://www.degate.org/documentation/
[28] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.
[29] Y. Liu, R. S. Shelar, and J. Hu, "Simultaneous technology mapping and placement for delay minimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 3, pp. 416–426, Mar. 2011.
[30] M. C. Hansen, H. Yalcin, and J. P. Hayes, "Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering," *IEEE Design Test Comput.*, vol. 16, no. 3, pp. 72–80, Jul./Sep. 1999.
[31] F. Corno, M. S. Reorda, and G. Squillero, "RT-level ITC'99 benchmarks and first ATPG results," *IEEE Design Test Comput.*, vol. 17, no. 3, pp. 44–53, Jul./Sep. 2000.

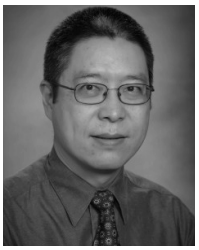
Yujie Wang (M'17) received the B.E. degree in electrical science and technology from Nankai University, Tianjin, China and the Ph.D. degree from the College of Electronic Information and Optical Engineering, Nankai University, in 2017.

He was a Visiting Scholar with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA, as a joint Ph.D. Student. He is currently an Assistant Professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His current research interests include digital VLSI design and hardware security.



Pu Chen received the B.Eng. degree in automation from the Huazhong University of Science and Technology, Wuhan, China, in 2014 and the M.S. degree from the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA.

He joined Amazon Web Service, Inc., Seattle, WA, USA, in 2017.



Jiang Hu (F'16) received the B.S. degree in optical engineering from Zhejiang University, Hangzhou, China, in 1990 and the M.S. degree in physics and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, MN, USA, in 1997 and 2001, respectively.

He was with IBM Microelectronics, New York, NY, USA, from 2001 to 2002. In 2002, he joined the Electrical Engineering Faculty, Texas A&M University, College Station, TX, USA. His current research interests include optimization for large-scale computing systems, especially VLSI circuit optimization, adaptive design, hardware security, resource allocation, and power management in computing systems.

Dr. Hu has served as a Technical Program Committee Member for the Design Automation Conference, the International Conference On Computer Aided Design, the International Symposium on Physical Design, the International Symposium on Quality Electronic Design, ICCD, the Design, Automation and Test in Europe, the International Symposium on Circuits and Systems, the Asia and South Pacific Design Automation Conference, and the International Symposium on Low Power Electronics and Design. He received the Best Paper Award at the ACM/IEEE Design Automation Conference in 2001, the IBM Invention Achievement Award in 2003, and the Best Paper Award at the IEEE/ACM International Conference on Computer-Aided Design in 2011. He received the Humboldt Research Fellowship in 2012. He is the General Chair of the 2012 ACM International Symposium on Physical Design. He served as an Associate Editor for the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS from 2006 to 2011. He is currently an Associate Editor for the *ACM Transactions on Design Automation of Electronic Systems*.

Guofeng Li received the B.S. degree in physics and the M.S. and Ph.D. degrees in electronic science from Nankai University, Tianjin, China, in 1982, 1985, and 2002, respectively.

He is currently a Professor at Nankai University, where he serves as the Director of the Center of Experimentation. His current research interests include VLSI design, analog IC design, and system integration.

Dr. Li is an Executive Council Member of the China Physics Society.

Jeyavijayan (JV) Rajendran (S'09–M'15) received the Ph.D. degree from the Electrical and Computer Engineering Department, New York University (NYU), New York, NY, USA.

He is currently an Assistant Professor at the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA. His current research interests include hardware security and emerging technologies.

Dr. Rajendran was a recipient of three student paper awards at the ACM Computer and Communications Security in 2013, the IEEE Defect and Fault-tolerance Symposium in 2013, and the IEEE VLSI Design in 2012; four ACM student research competition awards at the Design Automation Conference in 2012 and 2014, the International Conference on Computer-Aided Design in 2013, and the Grand Finals in 2013; the Service Recognition Award from Intel, Mountain View, CA, USA; a third place at Kaspersky American Cup in 2011; and the Myron M. Rosenthal Award for the Best Academic Performance in the M.S. degree from NYU in 2011. He organizes the annual Embedded Security Challenge, a red-team/blue-team hardware security competition.