Detour: Layout-aware Reroute Attack Vulnerability Assessment and Analysis

Minyan Gao
ECE Department
University of Florida
Gainesville, United States
minyan.gao@ufl.edu

Domenic Forte
ECE Department
University of Florida
Gainesville, United States
dforte@ece.ufl.edu

***Abstract*—Over the past several decades, the rate of innovation and performance enhancement in integrated circuits (ICs) is mind-boggling, making them ubiquitous in a wide spectrum of critical applications ranging from military infrastructure to personal healthcare. Lately, however, physical security has become a prime concern given the valuable assets that ICs process and store. Out of all invasive attack vectors, micro-probing attacks emerge as one of the most threatening because they utilize advanced focused ion beam (FIB) systems for *post-silicon* secret eavesdropping and circuit editing at a negligible footprint. As an evolved variant of micro-probing attacks, *reroute attacks* can effectively abolish built-in shielding countermeasures to access the security-sensitive signals underneath. To mitigate and tackle such challenges, we propose a layout-level framework called *Detour* to automatically evaluate the exploitable vulnerabilities. Specifically, we utilize a linear programming-based scheme to determine the layout-aware added traces length of reroute attempts given target assets. Experimental results show that all of the shielded designs act better than the non-shielded structures against reroute attack, and that the orthogonal two-layer shield structure has better performance than the parallel two-layer shield structure. In addition, we also consider both the independent and dependent scenarios based on whether circuit edit locations are allowed to interfere with each other or not. Our results show that a near 50% increase in attack cost can occur when utilizing our more realistic dependent estimation method.**

I. Introduction

The past few decades have witnessed the rapid scaling of integrated circuit (IC) technology nodes and the proliferation of leading-edge processors and lightweight terminals, enabling strong computational power and smart end-point connectivity. Despite the prosperity of the semiconductor industry, hardware security is becoming increasingly important given a variety of attack vectors such as side-channel attacks [1], [2] and fault injection attacks [3], [4] to compromise the confidentiality and integrity of security assets [5], [6]. As an invasive attack vector, focused ion beam (FIB)-based micro-probing attacks are drawing more and more traction from both academia and industry because of their capability to directly eavesdrop and edit the internals of a fabricated circuit at a negligible impact on the entire system. Specifically, FIB is able to mill and deposit materials at a *nano-scale* level allowing for high-precision post-silicon intrusions and tampering. An exemplary security attack is cloning an SRAM physical unclonable function (PUF) [7] where a FIB was utilized to etch out a portion of the SRAM's transistors to bias cells such that adversaries can predict start-up initialization and force them into predetermined ones. Other attack cases have been discovered for extracting sensitive plaintexts, private keys, and security tokens [8].

Recently, various countermeasures have been proposed to protect security-critical assets against invasive FIB probing attacks. For example, designers can place active *shield nets* at the top metal layers during the design time. As such, potential probing intrusions might compromise the active metal wires that continuously transfer specific-pattern signals; the mismatch between the information from the top-layer metal wires and underneath reference signals can be detected to trigger the subsequent countermeasures against micro-probing attacks [9], [10]. In addition, analog sensors like the probe attempt detector (PAD) [11] can capture the added capacitance and delay imposed by the attached probe in a timely manner. However, these existing solutions either suffer from exorbitant overhead or low reliability, failing to become a silver bullet to address threats. Even worse, an advanced variant of micro-probing attacks, namely *reroute attack*, has been proposed to effectively nullify the shield protection and thus be easier to access the sensitive signals than conventional bypass attacks [12]. The basis of the reroute attack is to destroy a part of the shield while replacing it with FIB at another location in order to gain access to sensitive nets in the design without being detected. In an effort to better understand this threat, in this paper, we propose a layout-aware reroute attack assessment framework called **Detour**. Detour allows designers to efficiently and accurately quantify the vulnerability of an IC at the physical design level. Our contributions are as follows:

- We propose a highly-automated layout-aware security assessment framework that evaluates design layouts' vulnerabilities against reroute attacks according to state-of-the-art FIB precision.
- We develop a new metric, *layout-aware added traces length*, to quantify the required efforts of reroute attacks. A linear programming-based approach is provisioned to automatically identify circuit edit locations on the shield nets to establish the reroute paths.
- We perform extensive experiments on a variety of physical design layouts of a system-on-chip (SoC) design using our Detour framework. The results demonstrate the

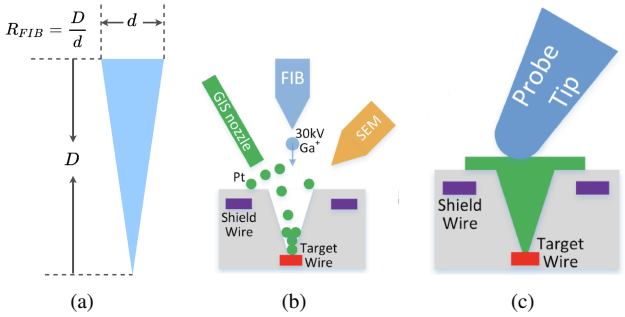


Fig. 1. Basics of FIB-based micro-probing attacks [13]: (a) FIB aspect ratio calculation where d is the diameter while D refers to the depth; (b) Platinum deposition in the milling cavity by FIB to build conducting path from the target wire (red); and (c) probe extracts information from the deposited conducting path.

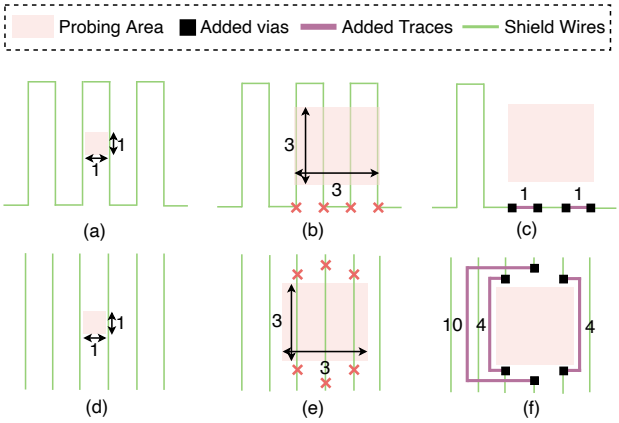


Fig. 2. Shield nets, bypass attack efforts, and reroute attack efforts. (a) possible bypass attack area, (b) opening a 3×3 $pitch^2$ area in reroute attack, and (c) edits needed (4 vias and 2 pitch long traces) for snake-like shield structure. (d) possible bypass attack area, (e) opening a 3×3 $pitch^2$ area in reroute attack, and (f) edits needed (6 vias and 18 pitch-long traces) for single parallel shield structure.

effectiveness of our assessment and suggest that the two-layer shield structure can lead to more resiliency against reroute attacks than its single-layer counterpart. Besides, the orthogonal structure has better resistance than the parallel one in the context of two-layer shield protection.

- We systematically consider both *independent* and *dependent* scenarios where the main difference is whether overlapping the circuit edits from reroute attacks is permitted or not. Results illustrate that a near 50% increase in terms of *layout-aware added traces* is required for the more practical dependent case. Additionally, we develop a graphic utility allowing for intuitive visualizations of exposure of target assets to reroute attacks and corresponding statistics.
- Upon acceptance of this paper, we will make our Detour assessment and visualization tool publicly available to the research community.

The rest of this paper is structured as follows. In Section II, we provide the background on the micro-probing attacks and countermeasures. Section III details our Detour framework, especially the metrics for reroute attack assessment and workflow to compute them for any design layout. Section IV presents the experimental results. Finally, we conclude this paper in Section V.

II. Background

In this section, we will first introduce FIB technology and FIB-based micro-probing attacks. Next, we discuss existing assessment solutions and countermeasures against probing attacks. Finally, we discuss our threat model.

A. Basics of FIB-based Micro-probing

FIB has emerged as a powerful technique for IC editing since it can remove and deposit materials with high precision, allowing for cutting traces or adding metal connections within a fabricated chip [14], [15]. Also, it is useful to create probing points for electrical testing. FIB circuit editing capabilities support fundamental electrical design characterization and/or verification of redesign parameters, as well as diagnosing manufacturing faults and anomalies [16]. Nevertheless, with advanced FIB techniques, adversaries are much more potent to directly eavesdrop and reconstruct partial or full security-sensitive assets, such as messages, decryption keys, and/or device configuration within ICs [8].

Fig. 1 illustrates the basic concepts of FIB-based micro-probing attacks. In detail, Fig. 1(a) presents an important parameter, i.e., aspect ratio, of FIB systems which is defined as $R_{FIB} = \frac{D}{d}$ where D and d refer to the depth and diameter of the milling hole, respectively. Clearly, the larger a FIB aspect ratio is, the more powerful adversaries could be because the milling hole can be too narrow to affect any shield nets and trigger the alarm. After milling the hole through the package to the sensitive metal wires using FIB, adversaries will follow the steps of metal deposition, dielectric deposition, and imaging of the IC using the scanning electron microscope (SEM) as depicted in Fig. 1(b). FIB systems can image, etch and deposit materials on an IC with a finely focused gallium ion (Ga+) beam at a 4∼5nm resolution. Helium and Neon ion based systems are even more precise.The navigation system coupled with FIB can characterize chip subsurface features to ensure compliant circuit-level edits. The conductors are milled through with the high energy Ga beam; tungsten (W), platinum (Pt) gas, or silicon dioxide would be released from the gas injection system (GIS) nozzle to be precisely deposited using the ion beam of appropriate gas chemistry. The established conducting path from the sensitive signals can be tapped using the external probe tip to extract the security assets as presented in Fig. 1(c).

B. FIB-aware Anti-probing Physical Design Flow

Figure 2 sheds light on the classification of shield structures, i.e., single-layer and multiple-layer in general. Single-layer can be further classified into *snake-like wires* and *parallel wires* as shown in Fig. 2(a) and 2(d), respectively. The snake-like structure has the advantage of fewer driving signals to cover a large sensitive area whereas the parallel shield structure

might have better resiliency against advanced attacks [12]. When it comes to the multiple-layer shield structure, three types are mostly considered, i.e., orthogonal, parallel, and random shield. In order to achieve optimal protection, the minimum spacing between each shield net on the same layer is required. As for the different layer shield nets, in order to increase the overall shield coverage, an additional 50% offset of the pitch size may be added to the lower layer shield of the two-layer parallel shield. The proposal of the FIB-aware anti-probing physical design flow, iPROBE, [7], [12] enables different forms of shield structures including both single and two-layer shield structures.

C. Countermeasures and Limitations

There are two main categories of FIB-based probing attacks, i.e., *bypass attack* and *reroute attack*. Their essential difference lies in the necessity of circuit editing. Specifically, a bypass attack occurs when attackers break through the gap space of shield nets by creating a tiny hole without cutting off shield/alarm wires. In contrast, reroute attack can utilize the circuit editing ability of FIB to reroute a path between the equipotential points on the shield wire to render a large portion of shield protection to no avail.

There are a variety of countermeasures and evaluation approaches being proposed against FIB-based probing attacks. For instance, [17] proposes a FIB-aware anti-probing physical design flow that primarily utilizes internal shield nets within the design layout. It is able to implement single-layer and two-layer parallel shield structures against probing from the top metal layer of the chip. [18] takes a step further to implement the two-layer parallel and orthogonal structure against FIB probing from both the top metal layer and silicon substrate of the ICs. *Exposed area* metric is used to evaluate the attack efforts of the bypass attack, which assesses the gap space between shield wires; the larger the exposed area is, the more vulnerable that design is against probing attacks.

When it comes to reroute attacks, [12] utilizes *added traces length* metric to quantify the required efforts of reroute attacks. For example, to create a 3×3 $pitch^2$ hole area to access the target net as shown in Fig. 2(a) and 2(c) would require 4 vias and 2 pitches long traces, and 4 vias and 18 pitches long traces in total respectively to be added on the shield nets for probing attacks as presented in Fig. 2(b) and 2(d). However, the approach in [12] merely considers fixed shield structures. In addition, the cost for different shield structures is calculated theoretically based on the ideal number of shield nets to be distributed in the design layout. *However, in practice, the routing conditions can vary a lot such that shield nets might not be routed in the layout exactly as expected. For example, the existence of routing congestion due to the limited available space of the protected region might push some shield nets to other non-optimal layers or even somewhere outside the shield nets region. In contrast, our Detour framework takes the actual design layout into consideration to enable more realistic estimation instead of optimistically considering the reroute attack cost of a fixed shield structure.*

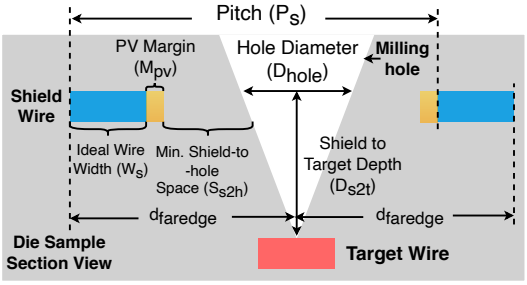


Fig. 3. Calculations for $d_{faredge}$.

.

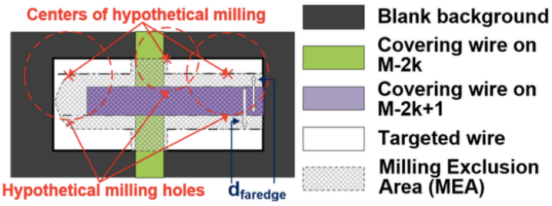


Fig. 4. Exposed area (EA) calculation [17].

D. Exposed Area

To assess a design's vulnerability to bypass probing attacks, we rely on the *exposed area* metric proposed in [13], which assumes that a complete cut on the shield wire is required for the detection of the attack. As such, it generates a corresponding probing area given the structure of surrounding shield nets and specified FIB aspect ratios. In other words, [13] assumes a model that one can detect the probing intrusions once the center of the FIB milling hole falls within the distance of $d_{faredge}$ from the far edge of the shield wire as defined below. Fig. 3 presents an equivalent intersectional view of the probing region regarding $d_{faredge}$ calculation by labeling important parameters.

$$d_{faredge} = \frac{D_{s2t}}{2R_{FIB}} + W_s + S_{s2h} + M_{PV} \quad (1)$$

where

- D_{s2t} is the depth or distance from the shield layer to the target layer in the IC layout. This depth should be available in the process design kit (PDK) for the IC's technology node.
- R_{FIB} denotes the FIB aspect ratio (see Fig. 1(a)), which can be found in FIB datasheets and in the case of probing represents the attacker's capability.
- W_s represents the nominal width of shield wires. The minimum wire width is a parameter that can be found in the PDK.
- M_{PV} is the process variation margin of shield wires.
- S_{s2h} is the space required between shield and hole to avoid shorts created by operator/FIB localization error. This parameter can be estimated by the FIB's datasheets and empirical studies.

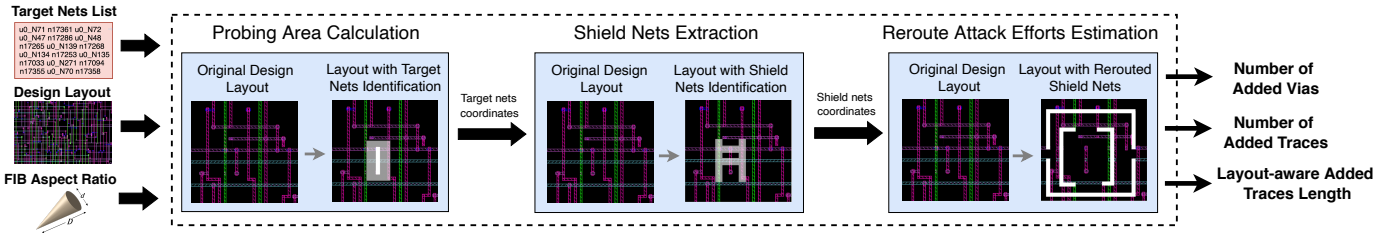


Fig. 5. Overview of our Detour framework for assessing the vulnerabilities of security-sensitive signals in the physical layout against FIB reroute attacks.

After identifying the $d_{faredge}$, Fig. 4 shows the determination of the exposed area for any target wire in a design layout. In detail, the covering wires (green and purple regions) at higher metal layers above the target wire layer (white area) can project a so-called milling exclusion area (MEA), i.e., the shaded region in Fig. 4, indicating that the attack will be detected if the milling center falls in this area. Next, the complement area of the MEA on the target wire is the exposed area (EA), which varies with the different FIB aspect ratios. The larger the exposed area of a design layout is, the more vulnerable it is to probing attacks.

E. Threat Model

In this paper, we assume that the electrical probing intrusions come from the top metal layer of the IC *perpendicularly*. The attacker aims to extract asset information through the probing attack with the possession of the entire layout information obtained through reverse engineering or unauthorized access to the database of a foundry or design house. Adversaries are supposed to be capable of performing both bypass attacks (milling a hole in the shield-free area directly) and reroute attacks (cutting and reconnecting shielding wires) and subsequently building the conducting path via the milling hole probing at the pad to extract asset information. To the best of our knowledge, our Detour framework is a first-of-its-kind solution focusing on the security assessment of reroute micro-probing vulnerabilities for real layouts.

III. Detour Framework

In this section, we shall detail our methodology for enabling the layout-aware assessment of anti-probing designs against reroute attacks. Also, we consider the overlap of circuit edits, which might complicate the steps in which a reroute attack is executed and therefore how its effort should be estimated.

A. Overview

We aim to develop a layout-aware assessment framework that takes floorplanning, cell placement, and routing of the target implementation into consideration to assess the vulnerabilities of security-critical nets against FIB reroute attacks. The overall workflow of the Detour framework against reroute attack is illustrated in Fig. 5. Detour accepts the design layout and a list of target nets (i.e., the nets carrying security assets) as inputs. Besides, the FIB aspect ratio (as introduced in Section II-A) is a user-defined input that is crucial to evaluate reroute attacks according to adversarial capabilities. There are three main stages in the Detour framework, i.e., *probing area calculation*, *shield nets extraction*, and *reroute attack efforts estimation* to yield the assessment results of required efforts of reroute attacks (e.g., number of added vias, number of added traces, and layout-aware added trace length). Specifically, Detour first extracts layout information, i.e., the location of metal wire instances of target nets, and calculates their exposed area (see Section II-D) to identify weak points given the FIB aspect ratio. Also, a collection of protected shield nets for each target net can be resolved accordingly. Next, shield nets within the probing area are analyzed by Detour using linear programming to report locations of circuit edits to be added for each shield net and the total reroute attack efforts.

TABLE I
Notations of Constraints

Notation	Definition
D_{VT}	Distance between vias to probing area
D_{VV}	Distance between vias to vias
D_{TP}	Distance between traces to probing area
D_{TT}	Distance between traces to traces

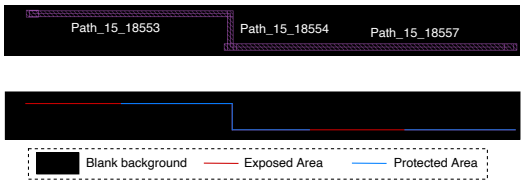


Fig. 6. Constituent shapes of the net $n8998$ and their exposed (red) and protected (blue) area.

B. Probing Area Calculation

The probing area calculation step will take the design layout, list of target nets, and FIB aspect ratio as the input, and will report the target nets wires as the candidate for the reroute attackers. In detail, it will first identify the metal layer of each target wire, and perform an estimation of the exposed area projected on the topmost layer with $d_{faredge}$ as shown in Equation (1). Nets in the layout designs always consist of a number of wires, usually called *shapes* by layout design tools, which have different names and might be located at different

TABLE II
EXPOSED AREA AND RATIO FOR DIFFERENT METAL WIRES.

Wire Name	Path_15_18553	Path_15_18554	Path_15_18557
Exposed Area (μm^2)	10.086	0	12.722
Ratio	49%	0	40%

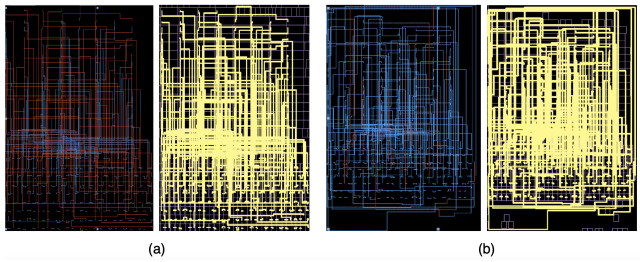


Fig. 7. The percentage of exposed area (red) on the target nets (yellow) in (a) and (b) is 62.28% and 8.77% respectively.

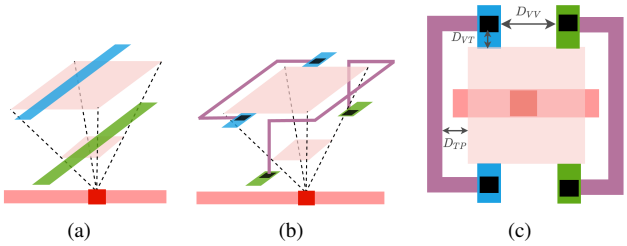


Fig. 8. (a) Shield nets extraction; (b) Reroute effort estimation; (c) Cross-sectional view of signals rerouted by FIB.

metal layers. For example, Fig. 6 shows three constituent wires for the target net *n8998* in the design layout.

In order to get the exposed area of the target nets, Detour will iterate each shape of the target nets and report the size of the exposed area and the ratio of the exposed area on the target net. As for the wires in Fig. 6, we can acquire the size of the exposed area and ratio as shown in Table II. Note that Detour chooses the wire with the most exposed area instead of the ratio as it is possible that the metal wire with a larger exposed ratio has a relatively smaller exposed area. Furthermore, the most exposed region will be reported as the best candidate for the reroute adversaries.

Fig. 7 presents an example regarding the exposed area of two AES designs where the milling exclusion area (blue), exposed area (red), and target nets area (yellow) are colored accordingly. The AES implementation in Fig. 7(a) is considerably more vulnerable than the one in Fig. 7(b) since the percent of its exposed area is 62.28% which is much higher than 8.77% of the other, indicating more exploitable space for probing intrusions.

C. Shield and Other Obscuring Nets Extraction

For all the metal wires that obstruct the attacker's path to the target net, they can be classified into two categories. The first type are a set of special internal nets that is utilized to be *shield nets*, which can be identified and constructed as described in Section II-B. The other group of obscuring nets are some *other design wires* that are routed on the layer above the target nets.

Algorithm 1: Shield Nets Extraction

Input: **Layout** - Physical design layout
Input: **Tar** - Coordinates of target wires
Input: R_{FIB} - FIB aspect ratio
Input: $Tech_{para}$ - Technology parameters
Output: $d_{faredge}$ of the target wire
Output: **MEA, EA** - MEA and EA of the target wire
Output: $Coor_{shield}$ - Coordinates of the shield nets
Output: $Layer_{shield}$ - Metal layer of the shield nets
1 Load the physical design layout **Layout**
2 Input R_{FIB}, $Tech_{para}$, **Tar** and identify the $d_{faredge}$
3 Apply the $d_{faredge}$ of the target wire and identify its **MEA**
4 **EA** $= \{\ Area \mid Area \in Tar\ and\ Area \notin MEA\}$
5 $\{Coor_{shield}, Layer_{shield}\}$=get_objects_by_location -intersect *EA*

For the wire with the most exposed area chosen for each target net, its probing area will be identified on the topmost metal layer, where the vias and traces for re-routing all obscuring nets are to be added when a reroute attack is performed. The physical design tool takes the physical design layout, FIB aspect ratio, and technology information as the inputs and will first identify the exposed area of a target wire. Then, the tool will report all the obscuring nets that cross the current probing area. This whole process is elaborated in the pseudocode as shown in **Algorithm 1**. Algorithm 1 starts with the target wires which are the input to the algorithm. Given the FIB aspect ratio, technology library dependent parameters, such as the wire width, distance between each metal layer and process variation margin, as shown in Equation (1), a value of $d_{faredge}$ can be obtained, which determines the size of the milling exclusion area (MEA) as shown in Fig. 4. Then, the EA can be acquired by getting the complement area on the target wire area projected onto the topmost metal layer. Finally, it will report all the obscuring nets and location in the upper metal layer that cross the EA of the current target wire, including their coordinates and metal layers in the design layout.

As shown in Fig. 8, after the identification of the target net, (red rectangle in the lower layer), then, blue and green shield nets will be recognized to cross with the light red probing area from different upper metal layers in Fig. 8(a). The extracted shield nets will then be used to estimate reroute efforts, i.e., the black vias and purple lines to be added by FIB to retain design/shield net continuity (see Figs. 8(b-c)).

D. LP-based Reroute Attack Effort Estimation

To account for the physical layout when assessing the design's vulnerabilities against reroute attack, we propose a new metric, *layout-aware added traces length*, i.e., the length of added traces a successful reroute attack requires given the specified layout. We present our linear programming-based reroute attack estimation methodology in **Algorithm 2**. The algorithm takes five inputs, i.e., the input physical design layout *Layout*, the set of technology library constraints for linear programming **C**, the set of all target nets carrying security assets **Tar** = $\{Tar_1, Tar_2, ..., Tar_M\}$ where *M* is the number of target nets, the set of exploitable probing area $\mathbf{A}_{prob}$ = $\{A^1_{prob}, A^2_{prob}, ..., A^M_{prob}\}$ for each of target nets, and the set

Algorithm 2: Linear Programming in Estimating Reroute Paths

Input: *Layout* - input physical design layout
Input: **C** - technology library constraints for linear programming
Input: **Tar** = $\{Tar_1, Tar_2, ..., Tar_M\}$ - set of all target nets
Input: $\mathbf{A}_{prob}$ = $\{\mathbf{A}^1_{prob}, \mathbf{A}^2_{prob}, ..., \mathbf{A}^M_{prob}\}$ - set of probing area
Input: **Shield** - set of all shield nets for each target net in **Tar**
Output: **Vertices** - set of vertices at the ends of reroute added traces
Output: L - Total length of added traces length

1 Load the physical design layout *Layout*
2 Initialize $l \leftarrow 0$, $Num \leftarrow$ |**Shield**|
3 **for** i = 1: M **do**
4 **while** $l \leq Num$ **do**
5 Initialize $L_i \leftarrow 0$ and $\mathbf{C}_i \leftarrow \emptyset$
6 $Tar_i \leftarrow$ the i^{th} target net in **Tar**
7 $\mathbf{A}^i_{prob} \leftarrow$ the i^{th} set probing area in $\mathbf{A}_{prob}$
8 $A^i_{prob,l} \leftarrow$ the l^{th} probing area in $\mathbf{A}^i_{prob}$
9 **Shield**$_i \leftarrow$ shield nets of Tar_i from **Shield**
10 **for** j = 1: N **do**
11 $Shield_{i,j} \leftarrow$ the j^{th} shield net from **Shield**$_i$
12 **Vertices**$_{i,j} \leftarrow$ the set of vertices of $Shield_{i,j}$
13 $L_{i,j} = [d(V_1, V_2) + d(V_2, V_3) + d(V_3, V_4)]_{i,j}$
14 **for** k = 1:3 **do**
15 $V_{i,j,k} \leftarrow$ the k^{th} vertex of $Shield_{i,j}$
16 $C1 : Dist(V_{i,j,k}, V_{i,j,(k+1)}) \geq D_{VV}$
17 $C2 : Dist(V_{i,j,k}, A^i_{prob,l}) \geq D_{VT}$
18 $\mathbf{C}_i$ adds $C1$ and $C2$
19 end
20 $\overline{L_i} = \overline{L_i} + L_{i,j}$
21 **end**
22 {**Vertices**$_i$, L_i} $\leftarrow$ *Linear_Prog.*($\overline{L_i}$, $\mathbf{C}_i$)
23 **if Vertices**$_i \cap$ **Vertices** $= \emptyset$ **then**
24 **break**
25 **else**
26 $l = l + 1$
27 **end**
28 **end**
29 L = L + L_i
30 **Vertices** adds **Vertices**$_i$
31 $l = 0$
32 **end**

of obscuring/shield nets also for each target nets in **Tar**. With **Algorithm 2** and the linear programming (LP) engine, we are able to figure out the minimum total length of added traces length L for feasible reroute attacks and how the vertices of each added reroute trace should be placed. The main flow of **Algorithm 2** is as follows.

Stage 1: Initialization and Processing (lines 1-9). Algorithm 2 first reads the layout-level placement and routing information from *Layout*. Then, it focuses on the set of M target nets carrying the security assets and thus becoming the probing targets. The variable $\overline{L_i}$ and set $\mathbf{C}_i$ are initialized for representing the added trace length and the optimization constraints, respectively. The i^{th} target net Tar_i is accessed from **Tar** along with its associated information such as the probing area A^i_{prob} and relevant shield nets **Shield**$_i$.

Stage 2: Added Trace Length Formulation and Constraints (lines 7-21). For the set of shield nets **Shield**$_i$, on each of the shielding net $Shield_{i,j}$, there are a couple of vertices for reroute attack added trace **Vertices**$_{i,j}$. As illustrated in Fig. 8(c), each reroute path is determined by the locations of *four* vertices. Therefore, the length of the added trace can be formulated as $L_{i,j} = [d(V_1, V_2) + d(V_2, V_3) + d(V_3, V_4)]_{i,j}$. Note that, $L_{i,j}$ is a *linear function* to be resolved by the linear programming technique later under the constraints. As for the linear constraints for linear programming, we store the constraints C_1 and C_2 which have been introduced in Table I in the set **C**. Specifically, C_1 refers to the minimum distance between two adjacent reroute vertices while C_2 stands for the minimum distance between any reroute vertex and the nearest boundary of the corresponding probing area A^i_{prob}. By iterating each of the vertices $V_{i,j,k}$ given the shield nets $Shield_{i,j}$, we can generate a set of constraints for setting up linear programming optimization in the next stage.

Stage 3: Linear Programming for Reroute Attack Efforts Estimation (lines 22, 29, 30). According to the linear function and constraints, we can formulate the linear programming problem as Equation 2 (**line 22**).

$$\{\textbf{Vertices}_i, L_i\} \leftarrow Min\ (\overline{L_i})\ subject\ to\ \mathbf{C}_i \tag{2}$$

All the optimization constraints considered in our framework, $\mathbf{C}_i$, are elaborated below. Note that the minimum distance between each segment of the metal wire varies with technology libraries and Table I list all the notations and their definitions.

- The first set of constraints enforces that a certain distance between each segment of the added traces in the layout must be maintained to ensure the signals extracted from the target nets to be reliable, which are expressed as,

$$D_{VT} > d_{VT,min} \tag{3}$$
$$D_{VV} > d_{VV,min} \tag{4}$$
$$D_{TT} > d_{TT,min} \tag{5}$$

  Here, we include the distance requirements between vias to vias, vias to metal wires, and wires to wires, to avoid the consequences such as the short of the signals.
- The next constraint enforces that no traces cross in the same layer, and is incorporated for the same reason as the first constraint, It can be stated as,

$$Trace_i \cap Trace_j = \emptyset \tag{6}$$

- To avoid affecting the normal signal transmission of shield wires, a minimum space will be reserved between traces to the probing area of the target net, expressed as,

$$D_{TP} > d_{TP,min} \tag{7}$$

Our linear programming will then automatically identify the best case where the added trace length of reroute attacks can be minimized within the constraint space of $\mathbf{C}_i$. In addition to the numeric value of L_i, the specific locations of **Vertices** of reroute traces can be resolved for analysis. By accumulating the L_i and **Vertices** for each of target nets Tar_i, we can generate the global layout-aware outputs, L and **Vertices** using **Algorithm 2**.

An example can be seen in Fig. 8 that, after identifying the probing area and the shield nets (blue and green wire) that need to be cut and reconnected in the reroute attack, the linear programming tool will reveal the optimal probing point on the current target wire (red point), the location of vias (black) and the added traces path (purple).

TABLE III
DESIGN TYPES USED FOR COMPARISON.

No.	Shield Type	Description
1	Original Design (No Shield)	Conventional physical design
2	One-layer Single Shield	Shield on M6
3	Two-layer Orthogonal Shield	Shield on M6 and M7
4	Two-layer Parallel Shield	Shield on M6 and M8

E. Dependence: Considering Overlap of Edits

The analysis performed by Algorithm 2 considers the situation where the probing of each target net is independent of all others. However, in reality, attackers tend to possess a limited number of FIB probe tips, while there tend to be more than hundreds of target nets, and then, attackers cannot perform the probing on all of the target nets at the same time. Therefore, if they did the probing on one target net after another, it is possible that the location of circuit edits on the topmost layer for different shield nets would be overlapping each other. To deal with this "dependence", the reroute attack positions for overlapping edits may need to be re-positioned to avoid interference. Fig. 9(a-b) shows the situation where edits do not overlap; thus there is no need to re-position the probing area(s) and the reroute effort estimate provided under the independent flow is fine. Fig. 9(c) shows a case which can result in overlaps. Thus, the reroute attack effort estimate (referred to as **independent case**) is optimistic. In practice, this would not be allowable due to the intersection of the probing areas and FIB edits as shown in Fig. 9(d). The dependent version of reroute attack effort estimate corrects this by re-positioning probing area #1 to avoid the overlap. This is more accurate and may increase the reroute attack estimate if the new position of probing area #1 is sub-optimal (i.e., contains more obstructing nets than the prior position).

In the process of the identification of the location of the vias for different covering shield nets, the circuit edit location of a shielding net needs to be relocated if it overlaps with the vias location of the other target net and it would be proper to add the constraint to the assessment flow that no overlapping between different circuit edits is allowed, as shown in Fig. 10 and Algorithm 2 **(lines 23-28)**. If the constraint that the location conflicts of the vias are not allowed is considered (referred to in this paper as the **dependent case**), first, we will need to record their coordinates. Then, after the location of the current via is identified, we will need to check whether it overlaps with any of other vias. If it does, we will have to follow the procedure shown in Fig. 10. That is, the position of the probing area for the current target is moved until there is not overlap with a previously edited target's probing area.

IV. EXPERIMENTAL RESULTS

In this section, we first articulate our experimental setup including the experimental layout designs. Next, we elaborate on the results of reroute attack efforts separately in independent and dependent scenarios using the Detour framework.

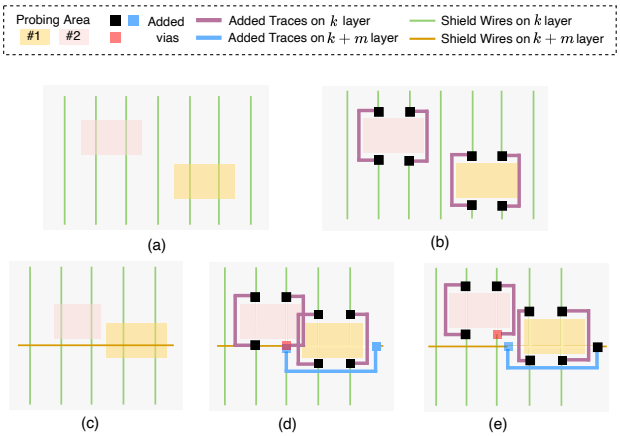


Fig. 9. Reroute attack effort estimation in independent and dependent scenario. (a): No overlapping in circuit edits resulting in (b) same reroute attack efforts for both independent and dependent case (no re-positioning needed); (c) Overlapping in circuit edit areas (re-positioning of edits needed) which leads to different estimation results between (d) independent case and (e) dependent case.

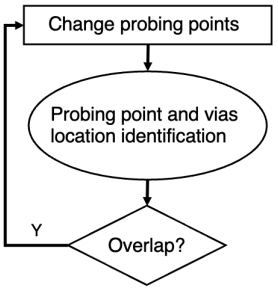


Fig. 10. Workflow of the non-overlapping circuits edit location identification.

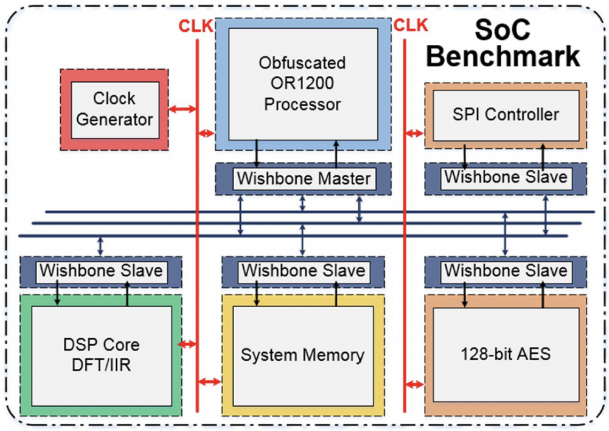
A. Experimental Setup

In this section, we use our proposed Detour approach to evaluate different design layouts against the reroute attack. We aim to evaluate how much reroute attack efforts the adversaries need to invest to perform a successful probing attack given different shield structures and different asset nets. Also, we compare our layout-aware estimation results with the prior state-of-the-art technique from [12]. Besides, the evaluation covers two situations: consider the probing action on each target net independently and dependently, which differs whether the overlapping of the circuit edits is allowed or not.

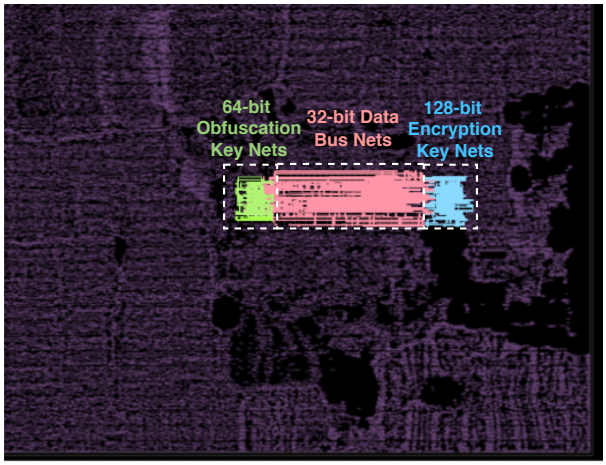
In order to perform the comparison between our method and [12], we choose the same SoC benchmark, i.e., the common evaluation platform (CEP) [19] developed by MIT as the common ground and synthesize the register-transfer level implementation using Synopsys Design Compiler in Synopsys SAED 32nm technology library. As depicted in Fig. 11, the SoC includes main components such as an AES encryption core, DSP core, SPI controller, Arbiter data bus structure, and clock generator. Also, the same set of target nets are chosen as [12], i.e., the encryption key nets in the AES module, the data bus nets from the OpenRISC processor (OR1200) to the AES module, and the obfuscation key nets in the OpenRISC

TABLE IV
REROUTE ATTACK ASSESSMENT ON DIFFERENT SHIELD STRUCTURES.

	Design No.	Enc. Key			Data Bus			Obf. Key		
		Vias	Traces	Length (mm)	Vias	Traces	Length (mm)	Vias	Traces	Length (mm)
No shield nets	**1**	374	169	122	1726	997	1798	567	266	135
Previous calculation [12]	**2**	494	247	93	2140	1070	1739	594	297	134
	3	990	495	279	4280	2140	5217	1190	595	403
	4	744	372	233	3210	1605	4347	894	447	337
Shield nets + other nets	**2**	556	316	160	2777	1221	2299	652	316	182
	3	1048	699	379	4980	2556	5797	1466	676	527
	4	866	456	352	3971	2020	4929	1010	592	420
Only shield nets	**2**	427	208	84	2167	998	1679	580	279	127
	3	921	536	264	4150	2042	5170	1220	570	399
	4	699	331	232	3147	1489	4279	869	466	310



(a)



(b)

Fig. 11. (a) Diagram of the SoC used to evaluate our algorithm [12]. (b) Target group nets in the SoC benchmark: obfuscation key nets, data bus nets and encryption key nets.

processor.

We also create shield nets in the layout of the CEP SoC for comparison with [12]. In detail, there are three optimal options for the shield layer structure as concluded by [12], i.e., M6, M6+M8, and M6+M7, for single layer, parallel two-layer, and orthogonal two-layer, respectively. Each of them leads to the largest required efforts of reroute attacks in the

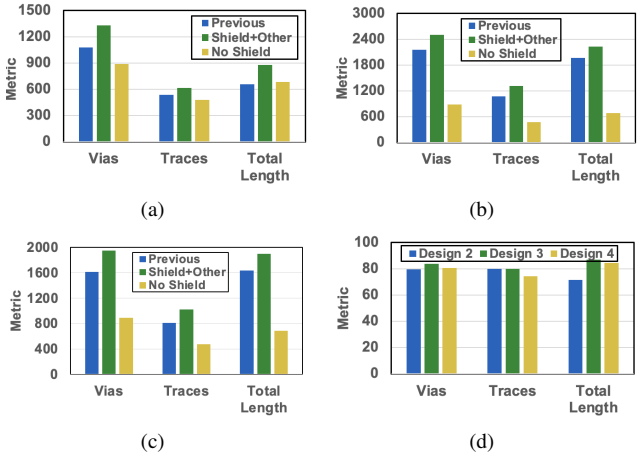


Fig. 12. Reroute attack efforts on different shield structures: (a) Single layer shield on M6 metal layer. (b) Orthogonal two-layer shield on M6 and M7 metal layer. (c) Parallel two-layer shield on M6 and M8 metal layer. (d) The percentage of the reroute attack cost from shield nets' protection.

corresponding scenarios. Therefore, we also follow the same models of shield protection in our benchmark implementation for fair evaluation and comparison. More details regarding the experimental layouts can be found in Table III.

B. Evaluation of Independent Reroute Attack

The *independent* reroute attack vulnerability evaluation focusing on different probing targets is presented when the overlapping of the circuit edits is allowed. We quantify the reroute efforts as three metrics, i.e., the number of added vias, the number of added traces, and the total length of added traces. Also, as introduced in Section IV-A, there are four layout-level implementations in our assessment, specifically, no shield nets (the shielding protection is provided by non-shield obscuring nets in the original design), one-layer single shield at M6, two-layer orthogonal shield, and two-layer parallel shield, corresponding to Design No. from 1 through 4, respectively, as illustrated in Table III and IV. In addition to the baseline Design No. 1 (no shield nets), Table IV denotes other three sets of assessment results regarding Design 2/3/4 as *Previous calculation* (the results reported by [12]), *Shield nets + other nets*, and *Only shield nets*. Note that the statistics of quantified reroute attack efforts are tabulated in Table IV

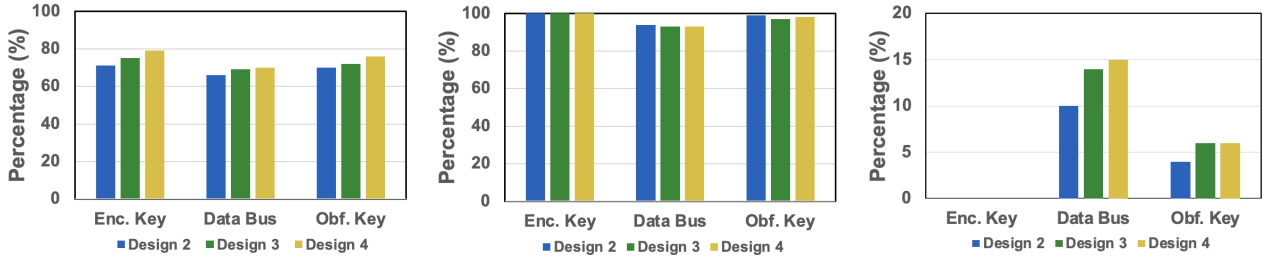


Fig. 13. The proportion of the nets routed in their designated metal layers. (a) Shield nets layer distribution. (b) Target nets layer distribution. (c) Assets that distribute above designated layers.

for the three groups of target nets carrying AES encryption key, SPI data, and obfuscation key.

We can observe from Table IV that, the baseline design layout without any shield structures (Design No. 1) costs the *least* reroute attack efforts and thus provides the *worst* protection among all of the design layouts. For instance, it only needs 374 added vias, 169 added traces, and 122 mm added trace length to enable reroute attacks for all the AES encryption key nets as analyzed by our Detour solution utilizing linear programming algorithms. In contrast, when we are using both shield nets and other functional nets for protection (*Shield nets + other nets*), a best case can be found if a two-layer orthogonal shield M6+M7 is deployed where triple times of added resources are required than the baseline scenario, i.e., 1048 added vias, 699 added traces, and 379 mm added trace length. It is worth noting that there might be some variations regarding the resiliency given the inherent randomness during design placement and routing, e.g., the total added trace length with a single layer shield at M6 for the *Only shield nets* case is slightly lower (84 mm) than the baseline one for the AES encryption key assets.

Generally, we found that the efforts from *Previous calculation [12]* are better than the ones of *Only shield nets* scenarios but worse than *Shield nets + other nets*estimated by our new Detour framework. The underlying reason lies that [12] assumes the maximum number of shield nets that can *always* be routed in the layers above the target nets area without considering practical constraints and possible routing congestion. As such, the attack cost is merely calculated based on the theoretical analysis in an ideal case. In practice, in order to enable accurate and fair assessment, Detour is additionally aware that not all of the shield nets can be routed on their designated metal layers, e.g., part of them have to be placed on other metal layers due to the restricted area. In other words, our experimental results reveal that the assumptions in [12] are not fair enough and *optimistically* estimate the available shield nets at the specified layer, yielding inaccurate results.. Such inaccuracies are corrected by Detour by considering the placement and routing (congestion) information at the layout-level of the entire design.

Additionally, three sets of target nets are treated as one group and their reroute attack efforts cost are demonstrated in Figure 12, and Figure 12(a), 12(b), 12(c) shows the results of Design No. 2, 3 and 4 as in Table III respectively, indicating the parameters and their amount to evaluate the attack cost. We can see more clearly that shield nets and other ordinary metal wires provide the most protection compared to the other categories.

Besides, Figure 12(d) reveals the proportion of the protection from the shield nets only in terms of the percentage for different design structures and all of them are over 70% and they can be as high as nearly 90%. It is consistent with the results shown in Figure 13(a), which demonstrates the proportion of the shield nets of all the covering nets, and nearly 70% of the protection comes from the shield nets. Figure 13(b) shows the target nets layer distribution, indicating that they are well constrained below the shield nets and that nearly 100% target nets are routed and distributed in their designated metal layers. Figure 13(c) reveals the proportion of the assets that are routed above shield, from which we can see that at least 85% of the targets are well protected under the shield nets layer. Its worth noting that all encryption keys nets are routed below the shield regardless of the shield structure used in the design.

C. Evaluation of Dependent Reroute Attack

In this section, we will evaluate the reroute attack efforts in the dependent manner, i.e., the circuit edits location of different target nets cannot overlap each other.

Fig. 14(a) demonstrates Detour's visualization results for the reroute attack with two probing areas (grey region) in the design layout. We can clearly see that there would be an overlap point as to make a reroute path on the green obscuring net for the probing area #1 and on the blue obscuring net for the probing area #2, and the length of the added traces is 1.674 μm and 1.872 μm for #1 and #2 probing area respectively. Therefore, when the dependent scenario is considered, our framework would require a different probing area location to avoid the conflicts in circuits edits location, as shown in Fig. 14(d), and the length of the added traces is 1.674 μm and 3.160 μm for #1 and #2 probing area respectively.

In addition, we have counted the number of the iterations implemented in order to make sure that there are no overlapping of circuit edits location. As the number of iterations increases, the reroute attack efforts for all designs will also grow since on one hand, re-identifying the circuits edits location would be time-consuming and most importantly, relocated

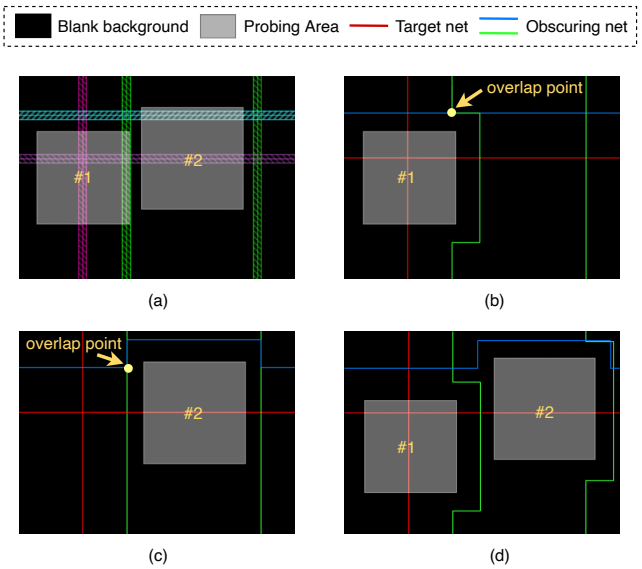


Fig. 14. Visualization results for the reroute attack efforts estimation. (a) Probing area for two target nets in the design layout. (b) Reroute path for probing area #1 in the independent scenario and the length of added traces is 1.674 μm. (c) Reroute paths for probing area #2 in the independent scenario and the length of added traces is 1.872 μm. (d) Reroute paths for two probing areas in the dependent scenario to avoid the overlap vias and the length of added traces is 1.674 μm and 3.160 μm for #1 and #2 probing area respectively.

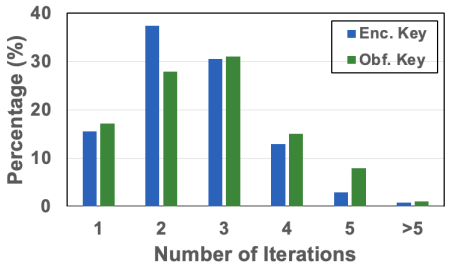


Fig. 15. The number of iterations required in order to identify non-overlapping circuit edits location in the reroute attack.

circuit edits would result in longer traces to be added and thus cause extra attack cost.

The iterations are calculated across the number from 1 to 5. Figure 15 reveals the proportion of each number of iterations required and we can see that most of them require one or two loops to identify the vias location and very few (less than 10%) need more than four iterations. Besides, we also collect the results for the total length of the traces to be added after all the iterations are done. Figure 16 demonstrates the results for the encryption key and obfuscation key target nets and compare the *Shield nets + other nets* and the *Only shield nets*. We can see that the increased reroute attack efforts varies with different shielded designs. Orthogonal and parallel two-layer shield structure (Design 2 and 3) is likely to bring more cost than single layer shielded design (Design 1), and nearly 50% increase is seen in some cases.

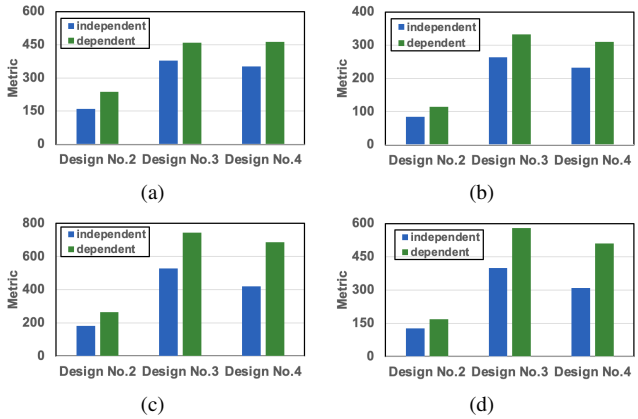


Fig. 16. Reroute attack efforts for independent and dependent scenarios. (a) Reroute attack efforts for the encryption key for shield nets + other nets. (b) Reroute attack efforts for the encryption key for shield nets only. (c) Reroute attack efforts for the obfuscation key for shield nets + other nets. (a) Reroute attack efforts for the obfuscation key for shield nets only.

V. Conclusion and Future Work

In this paper, we presented a layout-aware reroute attack assessment framework to evaluate the exploitable vulnerabilities. It takes the physical design into consideration and utilizes the linear programming method to automatically identify the FIB probing location, which determines the rerouted traces path to be added to perform the attack. With the identified circuits edits location, we can quantify the reroute attack cost using our layout-aware added traces metric. Further, the reroute attack efforts are considered in both the independent and dependent scenario, i.e., the overlapping of the circuit edits for different target nets are allowed and when it is not. Results show that all of the shielded designs act better than the non-shielded structures and two-layer shielded structure brings more attack cost than the single one, and especially for the two-layer shield layouts, orthogonal has better performance than the parallel. In addition, dependent situation is able to induce to nearly 50% attack cost compared to the independent case. In future work, we plan to extend the Detour framework to handle more generic FIB circuit edit attacks other than probing, e.g., using a FIB to create opens and shorts with security-critical nets in on-chip tamper detection and response mechanisms.

REFERENCES

[1] P. Kocher, J. Jaffe, B. Jun *et al.*, "Introduction to differential power analysis and related attacks," 1998.

[2] T. Zhang, J. Park, M. Tehranipoor, and F. Farahmandi, "Psc-tg: Rtl power side-channel leakage assessment with test pattern generation," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 709–714.

[3] M.-C. Hsueh, T. K. Tsai, and R. K. Iyer, "Fault injection techniques and tools," *Computer*, vol. 30, no. 4, pp. 75–82, 1997.

[4] M. R. Muttaki, T. Zhang, M. Tehranipoor, and F. Farahmandi, "Ftc: A universal sensor for fault injection attack detection," in *2022 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2022, pp. 117–120.

[5] T. Zhang, J. Wang, S. Guo, and Z. Chen, "A comprehensive fpga reverse engineering tool-chain: From bitstream to rtl code," *IEEE Access*, vol. 7, pp. 38 379–38 389, 2019.

[6] T. Zhang, F. Rahman, M. Tehranipoor, and F. Farahmandi, "Fpga-chain: Enabling holistic protection of fpga supply chain with blockchain technology," *IEEE Design & Test*, 2022.

[7] C. Helfmeier, C. Boit, D. Nedospasov, and J.-P. Seifert, "Cloning physically unclonable functions," in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2013, pp. 1–6.

[8] V. Ray, "Freud applications of fib: Invasive fib attacks and countermeasures in hardware security devices," in *East-Coast Focused Ion Beam User Group Meeting*, 2009.

[9] J.-M. Cioranesco, J.-L. Danger, T. Graba, S. Guilley, Y. Mathieu, D. Naccache, and X. T. Ngo, "Cryptographically secure shields," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. IEEE, 2014, pp. 25–31.

[10] M. Ling, L. Wu, X. Li, X. Zhang, J. Hou, and Y. Wang, "Design of monitor and protect circuits against fib attack on chip security," in *2012 Eighth International Conference on Computational Intelligence and Security*. IEEE, 2012, pp. 530–533.

[11] S. Manich, M. S. Wamser, and G. Sigl, "Detection of probing attempts in secure ics," in *2012 IEEE International Symposium on Hardware-Oriented Security and Trust*. IEEE, 2012, pp. 134–139.

[12] H. Wang, Q. Shi, D. Forte, and M. M. Tehranipoor, "Probing assessment framework and evaluation of antiprobing solutions," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 6, pp. 1239–1252, 2019.

[13] Q. Shi, N. Asadizanjani, D. Forte, and M. M. Tehranipoor, "A layout-driven framework to assess vulnerability of ics to microprobing attacks," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 155–160.

[14] V. Sidorkin, E. van Veldhoven, E. van der Drift, P. Alkemade, H. Salemink, and D. Maas, "Sub-10-nm nanolithography with a scanning helium beam," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures Processing, Measurement, and Phenomena*, vol. 27, no. 4, pp. L18–L20, 2009.

[15] H. Wu, L. Stern, D. Xia, D. Ferranti, B. Thompson, K. Klein, C. Gonzalez, and P. Rack, "Focused helium ion beam deposited low resistivity cobalt metal lines with 10 nm resolution: implications for advanced circuit editing," *Journal of Materials Science: Materials in Electronics*, vol. 25, pp. 587–595, 2014.

[16] C. Boit, C. Helfmeier, and U. Kerst, "Security risks posed by modern ic debug and diagnosis tools," in *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*. IEEE, 2013, pp. 3–11.

[17] H. Wang, Q. Shi, A. Nahiyan, D. Forte, and M. M. Tehranipoor, "A physical design flow against front-side probing attacks by internal shielding," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2152–2165, 2020.

[18] M. Gao and D. Forte, "iPROBE-O: FIB-aware Place and Route for Probing Protection Using Orthogonal Shields," in *2022 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, 2022, pp. 1–6.

[19] "Common Evaluation Platform v4.2," [Online], https://github.com/mit-ll/CEP, Accessed Jan. 14, 2023.