A Combined Logical and Physical Attack on Logic Obfuscation

Michael Zuzak
Rochester Institute of Technology
mjzeec@rit.edu

Yuntao Liu, Isaac McDaniel, Ankur Srivastava
University of Maryland, College Park
{ytliu,ilm,ankurs}@umd.edu

Abstract

Logic obfuscation protects integrated circuits from an untrusted foundry attacker during manufacturing. To counter obfuscation, a number of logical (e.g. Boolean satisfiability) and physical (e.g. electro-optical probing) attacks have been proposed. By definition, these attacks use only a subset of the information leaked by a circuit to unlock it. Countermeasures often exploit the resulting blind-spots to thwart these attacks, limiting their scalability and generalizability. To overcome this, we propose a combined logical and physical attack against obfuscation called the CLAP attack. The CLAP attack leverages both the logical *and* physical properties of a locked circuit to prune the keyspace in a unified and theoretically-rigorous fashion, resulting in a more versatile and potent attack. To formulate the physical portion of the CLAP attack, we derive a logical formulation that provably identifies input sequences capable of sensitizing *logically expressive* regions in a circuit. We prove that electro-optically probing these regions infers portions of the key. For the logical portion of the attack, we integrate the physical attack results into a Boolean satisfiability attack to find the correct key. We evaluate the CLAP attack by launching it against four obfuscation schemes in benchmark circuits. The physical portion of the attack fully specified 60.6% of key bits and partially specified another 10.3%. The logical portion of the attack found the correct key in the physical-attack-limited keyspace in under 30 minutes. Thus, the CLAP attack unlocked each circuit despite obfuscation.

Keywords

Untrusted Foundry, CLAP Attack, Logic Obfuscation, EOFM Probe

1 Introduction

The drive for cost savings has incentivized hardware design companies to outsource integrated circuit (IC) manufacturing to unaffiliated and untrusted foundries. This exposes design details to untrusted entities, raising concerns of intellectual property (IP) piracy, overbuilding, counterfeiting, and reverse engineering [25]. These security concerns are known as the *untrusted foundry problem*. Logic obfuscation was developed to address this problem by rendering IC function dependent on extra primary inputs, known as key inputs, integrated into combinational modules of a design. The resulting obfuscated modules exhibit their intended function only when a correct key is applied. By withholding this key from untrusted supply-chain entities, the design is rendered non-functional. This prevents malicious actions (i.e. piracy, overbuilding, etc.) during fabrication. After fabrication, the design house applies the correct key to activate the IC, enabling correct function for the end-user. Several surveys of obfuscation research have been compiled [5, 9].

The prevalence of logic obfuscation prompted the development of attacks against it. These attacks can be divided into two families: 1) logical and 2) physical. Logical attacks infer the key based on information leaked by the Boolean behavior or structural topology of the obfuscated circuit [2, 6, 10, 30, 32–34, 37, 39]. One of the most prolific logical attacks is the Boolean satisfiability (SAT) attack which identifies the secret key using a SAT solver [2, 10, 30, 32, 34]. Conversely, physical attacks infer the key based on empirical side-channel data collected from an obfuscated IC [17, 24, 31]. These attacks often employ non-invasive radiation measurements to gather side-channel data. Both logical and physical attack families are constrained to a limited view of the IC. This restricts each attack to a subset of information leaked by a circuit, enabling countermeasures to prevent each attack by exploiting their limited scope. For example, SAT attacks [2, 10, 30, 32, 34] are thwarted by adding SAT-resilient [27, 28, 36, 41] or SAT-Hard [13, 14] instances which are Boolean functions that yield an unduly complex circuit when viewed through a SAT-based lens. Conversely, physical attacks, such as electro-optical probing (EOP) or electro-optical frequency mapping (EOFM) [24, 31], rely on a rigid approach capable of extracting leakage only from key registers. A designer can include scatterers or noisy components alongside key registers to blur the side-channel and mitigate these attacks [23, 31].

These examples show how the limited scope of prior attacks (i.e. logical or physical) results in blind-spots that can be exploited to mitigate them. The existence of these targeted countermeasures limits the ability of prior work to generalize to arbitrary circuits. This motivates us to explore a unified attack approach that leverages the information leaked through both logical and physical channels in a mathematically-rigorous fashion. As a result, we design a combined logical and physical (CLAP) attack against obfuscation that exploits information leaked through both logical and physical channels. This allows the CLAP attack to maintain the advantages of both attack families without succumbing to the countermeasures that mitigate prior work. We summarize our contributions below:

(1) We derive a novel mathematical basis for an optical physical probing attack that extracts a secret key by logically sensitizing a design and measuring the impact on the index of refraction.
(2) Based on the mathematical basis, we develop two physical attacks. The first guides key extraction with a fixed EOFM probe. The second builds on the first to maximize the key leakage extracted by each probe using Weighted Max-SAT. A scanning EOFM probe then extracts this leakage from the entire die at once, minimizing probing events to unlock an IC.
(3) We define a process to integrate physical attack data into a logical SAT attack and vice versa. This mathematically links our physical and logical attacks, creating the unified CLAP attack. This combined approach exhibits the advantages of both attacks without the corresponding limitations. We show this in benchmarks which are not unlocked by either attack alone, but are unlocked by the combined CLAP attack. We also prove that the key returned by a CLAP attack is *always* correct.
(4) We evaluate the CLAP attack against four obfuscation schemes [13, 36, 40, 41] in benchmarks. The physical portion of the CLAP attack fully inferred an average of 60.6% of key bits and partially inferred another 10.3% with only 21 probes of the die. The logical portion of the CLAP attack identified a fully correct key from the physical-attack-limited keyspace in <30 minutes.

ICCAD '22, October 30-November 3, 2022, San Diego, CA, USA

ACM ISBN 978-1-4503-9217-4/22/10…$15.00
https://doi.org/10.1145/3508352.3549349

Thus, the CLAP attack fully unlocked each circuit in under 2.5 hours, compared to a conventional logical-only attack strategy that did not unlock most circuits before a 100 hour timeout.

(5) We release an open-source CLAP attack tool that can attack hardware or be simulated to assess security at design time [1].

2 Preliminaries

2.1 Logic Obfuscation and SAT Attacks

Logic obfuscation protects an IC by adding logic into combinational blocks driven by both internal signals and added primary inputs, called key inputs. A sample obfuscated circuit is in Fig. 2A. Notice that the I/O relationship of the circuit changes based on the key values ($key1, key2$) applied to the added key gates ($KG1, KG2$). Hence, obfuscation results in a design that only exhibits intended function when a correct key is applied. This allows the function of the circuit to be hidden during fabrication, preventing unauthorized use. Common obfuscation schemes include SFLL [27, 28, 41], Full-Lock [13, 14], LoPher [26], DECOY [7], and others [20, 29, 42].

Boolean satisfiability (SAT) attacks are potent logical attacks on obfuscation [2, 10, 30, 32, 34, 43]. These attacks iteratively prune the keyspace until only functionally-correct keys remain using a locked netlist and a black-box oracle. The locked netlist is in conjunctive normal form (CNF), a product-of-sums representation of a circuit, denoted $C^*(X, K, Y)$ that evaluates to true if and only if X, K, and Y are found such that $Y = C(X, K)$. The attack proceeds as follows:

(1) Eqn. 1 is solved. This finds two keys, K_1 and K_2, and an input, X_{di}, such that the two keys produce different outputs for X_{di}. X_{di} is called a *distinguishing input* (DI).

$$C^*(X_{di}, K_1, Y_1) \wedge C^*(X_{di}, K_2, Y_2) \wedge (Y_1 \neq Y_2) \quad (1)$$

(2) The oracle is queried with X_{di}. This finds the correct output Y_{di}. X_{di} is paired with Y_{di} to produce a *distinguishing input/output pair* (DIP). This DIP is appended to Eqn. 1, resulting in Eqn. 2.

$$C^*(X_{di}, K_1, Y_1) \wedge C^*(X_{di}, K_2, Y_2) \wedge (Y_1 \neq Y_2) \bigwedge_{j=1}^{i-1} (C^*(X_j, K_1, Y_j) \wedge C^*(X_j, K_2, Y_j)) \quad (2)$$

(3) Eqn. 2 is then solved to find a new DI (X_{di}) and associated keys (K_1 and K_2). The keys must be correct for all prior DIs in addition to producing different behavior for X_{di}.

(4) Eqn. 2 is iteratively solved/appended (steps 2-3) until it is not satisfiable. This occurs when no more DIs exist. A key is found satisfying all prior DIs. This key is functionally correct [20].

2.2 Electro-Optical Probing/Frequency Mapping

Electro-optical probing (EOP) is an IC test and validation technique that performs contactless voltage/timing measurements of internal die signals [15, 16]. To do so, an optical laser, focused on the active region, is applied to the backside of a silicon die. The voltages applied to the observed die region influence the free carriers present, altering the optical index of refraction. By measuring the power of the laser reflected by the die, coarse voltage data can be gathered, allowing non-destructive measurements of IC state.

EOP is limited by the wavelength of the probing laser, which must not exceed the band-gap of silicon (i.e. $E_g = 1.12eV$ or $\lambda = 1107$nm). Using a smaller wavelength results in *invasiveness*, where the probing laser alters measured reflectivity [15]. This limits the effective spatial resolution of EOP to between $220 - 775$nm [15, 19]. This is much larger than modern technology nodes, resulting in many transistors interfering in a complex fashion to produce each EOP measurement [15]. To address EOP resolution limitations, electro-optical frequency mapping (EOFM) was developed [15, 19, 22]. EOFM relies on the fact that switching transistors cause the reflected power of an incident laser to oscillate at the switching frequency. By band-pass filtering the reflected power at the switching transistor's frequency, it can be isolated from surrounding bulk and other logic not switching at this rate. This isolates a region of interest, improving the resolution of EOFM compared to EOP. Contactless probing has been adopted for failure analysis [15, 19, 22] and security [3, 35]. It has also been used to attack obfuscation [17, 23, 24, 31], including demonstrations in real hardware [23, 24].

2.3 Attacker Model

We consider the untrusted foundry attacker in [23, 24, 31] with:

(1) A contactless probing setup for EOFM. These are common in IC test facilities and can be rented for a modest fee [24].

(2) A netlist of the locked IC from reverse engineering GDSII files.

(3) An unlocked, probe-able oracle with scan-chain access. The oracle serves as a golden image for contactless probing or to query input/output relationships for combinational blocks.

Goal: Find a key, K, to make the locked circuit, $C(X, K)$, equivalent to the oracle, $C_o(X)$, for all inputs, X (i.e. $\{\forall X | C(X, K) = C_o(X)\}$).

3 Mathematical Basis for EOFM-Based Attacks

We begin our work by formalizing the physical component of the CLAP attack. To do this, we derive a mathematical basis for an optical physical probing attack that outlines a provable approach to EOFM-based key extraction. Prior work [24, 31] probes in an ad-hoc fashion with a limited theoretical basis guiding key extraction. This leads to two limitations: 1) the attacks fail to scale to increasingly small technology nodes [23] and 2) the attacks can be mitigated by targeted optical scatterers [31] or noisy components [23]. The theoretical rigor we introduce into our physical probing attack overcomes these limitations, distinguishing it from prior art.

3.1 Prior EOFM-Based Probing Attacks

To launch an EOFM attack on obfuscation (e.g. [24]) an electro-optical probing laser is applied to the backside of a correctly-keyed oracle IC. This laser illuminates a die feature (i.e. a collection of transistors), referred to as a *node*. The regions labeled "Probed Node" in the c17 circuit in Fig. 2A are several examples of probe-able die features represented logically. Some of the radiated power incident on the node is reflected back and can be measured. We refer to the process of illuminating a feature and measuring the reflected power as *probing*. We depict a probing setup in Fig. 1A. The voltages applied to die features in the probed node alters their index of refraction, in turn impacting the reflected power. Because the inputs/logic signals applied to the probed node determine these voltages, some transistor state information can be inferred.

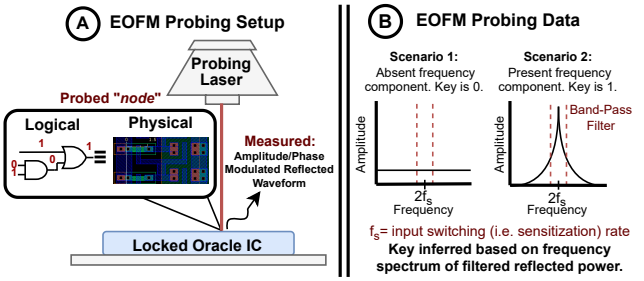

Figure 1: Sample EOFM probing process. A) EOFM probing setup. B) Sample data from EOFM probing.

Existing EOFM attacks extract the key by probing the output buffer of a key register during its hold state. For the c17 circuit in Fig. 2A, the spot labeled "Probed Node 1" depicts the illuminated feature. To do so, the key register is probed while being repeatedly reset using the global logic reset signal. This forces the register's

output temporarily to zero. The correct key value is then reloaded into the register during subsequent cycles before it is reset again. If the correct key value for a key register is one, the output of this register will toggle between zero and one as it is reset. This results in differing voltages applied to the probed output buffer, resulting in differing reflected power measurements by the EOFM probe. If the correct key value is zero, the output of this register will remain zero without toggling. This results in constant voltages applied to the probed output buffer, resulting in uniform reflected power. By band-pass filtering the reflected power at the frequency of logic reset toggling (i.e. sensitization frequency), the attacker can distinguish between these two key values as shown in Fig. 1B.

While these attacks have been employed against real hardware devices [23, 24], they use a rigid and ad-hoc strategy. This forgoes much of the leaked key information that can be inferred via probing, weakening the attack. We highlight two limitations of prior work [24, 31]. First, key register output buffers are the only probed nodes. Even though, as shown by [15, 19, 22], any die feature can be imaged. This limited view not only forgoes key leakage elsewhere in the die, but it also assumes that probes retain sufficient resolution to image these features. As technology shrinks, this becomes increasingly unrealistic, causing the attack to break down. Moreover, a designer can close the targeted side-channel by inserting scatterers [31] or noisy components [23] alongside key registers. Second, global logic reset toggling is the only node sensitization procedure. This rigidity forgoes key leakage by more targeted approaches, such as those in this work. These limitations motivate us to distill probing attacks to their underlying theory to produce a generalized form.

3.2 Efficient Key Pruning with EOFM Probes

An EOFM probing attack can be distilled as follows. If a node (i.e. a set of illuminated transistors) can be periodically sensitized by an input sequence such that some subset of keys alter node voltage and another subset does not, then a probing approach can be applied to the oracle circuit to disqualify a subset of keys. To do so, the attacker applies this input sequence while repeatedly probing the node. The attacker then band-pass filters the reflected power measurement at the rate that the input sequence is repeated, which we refer to as the *sensitization frequency*. If a frequency component is present in the resulting oracle measurement, then the correct key cannot be from the subset of keys that *do not* periodically sensitize the node. If a frequency component is not present, then the correct key cannot be from the subset of keys that *do* periodically sensitize the node. This allows the keyspace to be pruned to identify the key.

Let us now define notation. Consider obfuscated circuit $C(X, K)$ with input X, specifying the set of primary input pins $\mathcal{X}$, and key K, specifying the set of key input pins $\mathcal{K}$. To probe the circuit, we apply a probing laser incident on a die feature (i.e. a *node*). This node contains a set of transistors illuminated by the beam. We define vector $\vec{T}$ which contains a Boolean corresponding to whether a high or low voltage is applied to each transistor region in the probed node (i.e. voltage applied to source, gate, and drain). For example, if the beam illuminates a node containing a single PMOS transistor with an activated gate and a high voltage applied to the drain, the state vector is $\vec{T} = [Source, Gate, Drain] = [1, 0, 1]$. A subset of primary inputs, $\mathcal{X}_n \subseteq \mathcal{X}$, and key inputs, $\mathcal{K}_n \subseteq \mathcal{K}$, drive logic in the probed node's fan-in cone. $\vec{T}$ can be calculated based on input X_n and key K_n specifying only these inputs. We define function $N(X_n, K_n)$ for each node to calculate $\vec{T}$ (i.e. $\vec{T} = N(X_n, K_n)$). The values in $\vec{T}$ influence the reflectivity of a node, altering its reflected power when probed. While $\vec{T}$ is insufficient to calculate expected reflected power, probing a node with two different $\vec{T}$ values will produce different measurements. Finally, we define two subkeys, K_1 and K_2, that specify a subset of key input pins $\mathcal{K}_s \subseteq \mathcal{K}$. We now derive a mathematical basis for EOFM probing.

Theorem 3.1 (Mathematical Basis for EOFM-Probing Attack). *An EOFM probe on node $N(X_n, K_n)$ of obfuscated oracle circuit $C(X, K)$ periodically sensitized by inputs X_1, X_2 can disqualify any key containing either subkey K_1 or K_2 from the keyspace if:*

$$(N(X_1, K_1) = N(X_2, K_1)) \wedge (N(X_1, K_2) \neq N(X_2, K_2)) \quad (3)$$

Proof. The left side of Eqn. 3 ($N(X_1, K_1) = N(X_2, K_1)$) indicates that K_1 results in the same state ($\vec{T_1} = \vec{T_2}$) at node $N(X_n, K_n)$ for X_1 and X_2. The right of Eqn. 3 indicates that K_2 results in different node states ($\vec{T_1} \neq \vec{T_2}$) for X_1 and X_2. If an EOFM probe sensitizes node $N(X_n, K_n)$ with X_1 and X_2 at the sensitization rate. A sizable component at this frequency will exist only when X_1 and X_2 produce different states in $N(X_n, K_n)$. Thus, if such a frequency component is present, K_1 is disqualified. Otherwise, K_2 is disqualified. □

Thm. 3.1 constitutes a generalized mathematical basis for EOFM-based probing attacks on obfuscation. Based on Thm. 3.1, we quantify the number of keys eliminated by each EOFM probe with Cor. 3.2. Given that our eventual goal is to eliminate all incorrect keys, Cor. 3.2 quantifies the efficacy of a probing configuration.

Corollary 3.2. *The keys eliminated by Thm. 3.1 is $2^{|\mathcal{K}|-|\mathcal{K}_s|}$.*

We omit a proof for brevity, however, this result follows from the fact that all complete keys in the keyspace that contain the subset of values specified in the disqualified subkey are also eliminated. We now bound the number of key inputs specified for a subkey satisfying Thm. 3.1 for a node (i.e. $|\mathcal{K}_s|$) in Thm. 3.3.

Theorem 3.3 (Max Subkey). *If Eqn. 3 is satisfiable for a node with $|\mathcal{K}_n|$ key inputs, there is a satisfying subkey such that $|\mathcal{K}_s| \leq |\mathcal{K}_n|$.*

Proof. The behavior of a node, $N(X_n, K_n)$, is determined by its fan-in cone. If a solution to Thm. 3.1 for a node defines subkeys such that $|\mathcal{K}_s| > |\mathcal{K}_n|$, then there must exist key inputs in $\mathcal{K}_s$ not in $\mathcal{K}_n$ (i.e. not in the node's fan-in). These key inputs cannot impact node behavior, allowing them to be removed from $\mathcal{K}_s$ without violating Eqn. 3. As a result, if a solution to Thm. 3.1 exists such that $|\mathcal{K}_s| > |\mathcal{K}_n|$, then a solution must also exist such that $|\mathcal{K}_s| \leq |\mathcal{K}_n|$. □

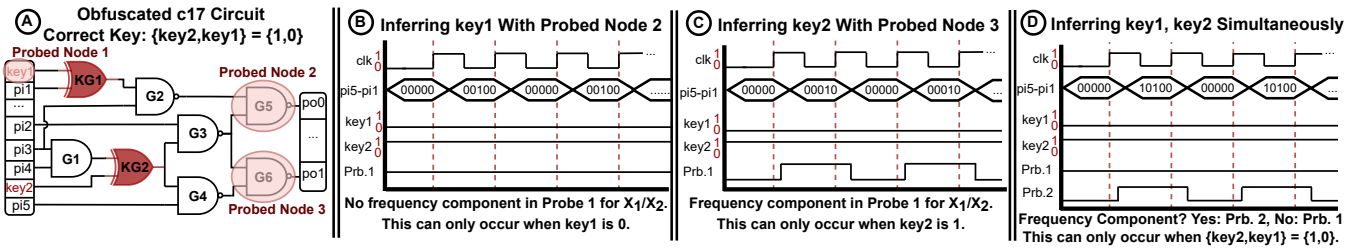

Figure 2: Mathematically-rigorous EOFM probing example. A) Obfuscated c17 circuit. B) Inferring key input $key1$ at Probed Node 2. C) Inferring key input $key2$ at Probed Node 3. D) Simultaneous inference of $key1$ and $key2$ using scanning EOFM probe.

This guarantees that the fewest keys eliminated by probing a node with $|\mathcal{K}_n|$ fan-in key inputs is $2^{|\mathcal{K}|-|\mathcal{K}_n|}$.

3.2.1 Mathematically-Rigorous Probing Example To demonstrate Thm. 3.1, we use it to infer the key of the c17 circuit in Fig. 2A. This circuit has 2 key gates, $KG1$ and $KG2$, fed by 2 key inputs, $key1$ and $key2$. We probe a correctly-keyed oracle of the circuit at the two red-shaded regions labeled Probed Node 2 and 3 in Fig. 2A[1].

Thm. 3.1 proves that the key can be partially inferred by identifying keys/inputs satisfying Eqn. 3. For example, the inputs X_1 ='00000' and X_2 ='00100' satisfy Eqn. 3 for subkeys K_1 ='×0' and K_2 ='×1' at Probed Node 2. To show this, consider each clause of Eqn. 3. To satisfy the first clause, we must ensure that inputs X_1 and X_2 produce the same transistor state at Probed Node 2 for subkey K_1. This is indeed the case in Fig. 2A where X_1 and X_2 produce identical inputs to the probed node ($G2 = 1, G3 = 1$). To satisfy the second clause, inputs X_1 and X_2 must produce differing transistor states at Probed Node 2 for subkey K_2. In Fig. 2A, if K_2 is applied, X_1 results in inputs $G2 = 1$ and $G3 = 1$ at the probed node, whereas X_2 results in inputs $G2 = 0$ and $G3 = 1$, producing differing transistor states.

To infer the key, the attacker periodically (i.e. cyclically) sensitizes the oracle with X_1 and X_2 while probing Node 2. This causes the oracle to produce distinguishable frequency signatures in the reflected power based on the subkey. If the correct key contains K_1, a constant reflected power is produced in the oracle with no dominant frequency component. If the correct key contains K_2, the measured reflected power will oscillate at the frequency that inputs X_1 and X_2 are applied (i.e. the sensitization frequency) due to the differing reflectively of the node for each input. This results in a dominant frequency component at the sensitization frequency that can be isolated by band-pass filtering at this rate. We depict both the input signals and the behavior of the probed oracle for this scenario in Fig. 2B. Notice that no frequency component is present in the reflected power at Probed Node 2 (Prb. 2). Thus, the correct key cannot contain K_2. Because K_2 specifies only a single key bit, $key1$ must be 0. Probed Node 3 can be used to infer $key2$, as shown in Fig. 2C. Thus, the entire key can be recovered using Thm. 3.1.

3.2.2 Merits of a Thm.-3.1-Guided Probing Approach

Probe-able Node Agnostic: Prior attacks only probe key register output buffers [24, 31]. Thus, they are mitigated by scatterers [31] or noisy components [23] alongside key registers. Conversely, Thm. 3.1 can be used to extract key leakage from *any* die feature. A countermeasure must protect every probe-able feature to mitigate our attack, leading to high overheads, as shown in Sec. 6.1.1.

Probe Resolution Agnostic: Probe resolution is physically limited by *invasiveness* to be between 220-775nm [15, 19]. With shrinking transistor size, probing lasers will include more transistors in a beam, increasing the complexity of probed die features. Prior work requires that key register output buffers be isolated from surrounding logic [24, 31]. When the technology size or probe resolution makes this impossible, these attacks fail. Thm.-3.1-driven probing, however, remains applicable because it makes no assumptions on the number/composition of transistors in the probed node.

Sensitization Agnostic: Nodes can be sensitized by any inputs satisfying Eqn. 3. This expands the inputs that create key leakage beyond the reset signal from prior art [24, 31], enabling input patterns to be chosen to optimize key leakage as done in Sec. 4.2.

Subkey Extraction: By probing nodes sensitized by few key inputs, large portions of the keyspace are eliminated. For example, if only one key input drives a probed node satisfying Eqn. 3, then any full key containing the eliminated key bit value is eliminated (50% of the keyspace). This makes Thm.-3.1-driven probing potent.

[1]This assumes that the resolution of the probe is sufficient to image any single gate in the die for the ease of understanding. However, since Thm. 3.1 makes no assumptions on probe resolution, this example can be extrapolated to other resolutions.

4 Proposed Physical Attack Algorithms

We now extend Thm. 3.1 into two physical attacks, which serve as the physical portion of the CLAP attack. Both attacks are theoretically rigorous attacks on obfuscation. Moreover, because Thm. 3.1 guides the EOFM probe for both algorithms, they exhibit the advantages in Sec. 3.2.2, differentiating them from prior art.

4.1 Attack 1: Logic-Guided Fixed EOFM Probe

Our first attack unlocks a circuit with a fixed EOFM probe. This setup applies a set of inputs to the circuit and then probes a single node. While any node can be measured, probing is performed in a one-at-a-time fashion. This reflects the simplest attack setup. To guide this approach, we represent Eqn. 3 as a logical SAT formula unique to the circuit being attacked. We then solve this logical formulation to identify inputs that maximize key leakage for a node. This results in a logically-guided physical attack on obfuscation.

4.1.1 Formatting Probing as Boolean SAT In Sec. 3.2.1, we manually identified inputs/subkeys satisfying Eqn. 3. To formalize input/subkey identification, we map Eqn. 3 to a Boolean SAT problem. By doing so, a SAT solver can identify input/key pairings satisfying Eqn. 3 (i.e. capable of causing key leakage). This allows a probing style attack to be launched efficiently. The SAT formulation to identify inputs, X_1 and X_2, and subkeys, K_1 and K_2, which satisfy Eqn. 3 for an arbitrary node is in Fig. 3A. The generic block, denoted *Probe Fanin*, corresponds to the fan-in cone to all transistors present in the probed node. For example, in Fig. 2A, the *Probe Fanin* for Probed Node 2 is gates G1, G2, G3, KG1, and KG2. The *Probe Fanin* block is unique to each node in the circuit. It allows the impact of inputs on the state of transistors in a node to be calculated. The remainder of Fig. 3A can be understood as two miters mapping to the two clauses in Eqn. 3 and an "Eliminated Keys" block to reflect the iterative nature of our attack. Miter 1 maps to clause 1 and ensures that $\vec{T}_1 = \vec{T}_2$ for subkey K_1. Miter 2 maps to clause 2 and

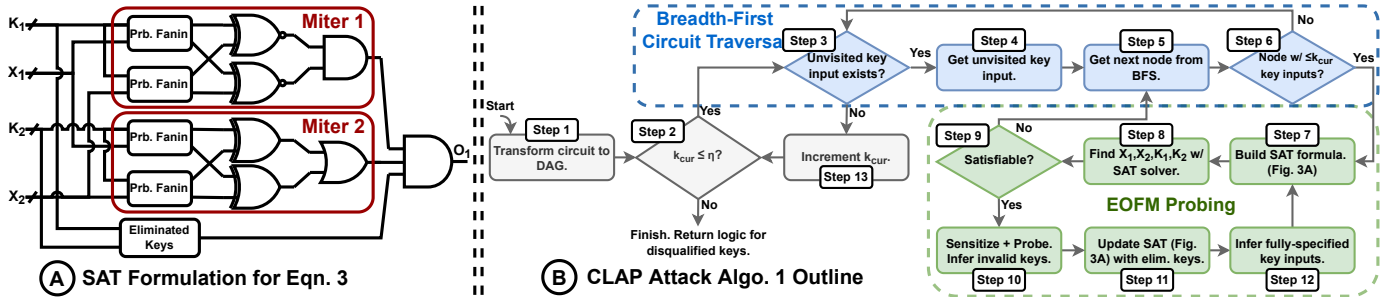

Figure 3: CLAP Attack Algorithm 1. A) SAT formula for Eqn. 3. B) Outline of physical CLAP attack algorithm 1.

ensures that $\vec{T}_1 \neq \vec{T}_2$ for subkey K_2. The "Eliminated Keys" block disqualifies any keys eliminated by prior probes from consideration as K_1 or K_2. This avoids duplicate work between iterations.

4.1.2 Fixed EOFM Probe Physical Attack Algorithm The SAT formulation in Fig. 3A guides our first algorithm for the physical part of the CLAP attack. This algorithm uses Thm. 3.1 to guide a fixed EOFM probe to infer the key. The goal is to disqualify the maximum keys each time a node is probed. The attack is outlined in Fig. 3B.

At a high level, this algorithm selects a key input and traverses its fan-out cone in a breadth-first fashion. As each probe-able node is visited, the number of fan-in key inputs is calculated. If there are $\leq k_{cur}$ key inputs, where k_{cur} is initially set to 1, the logical formulation in Fig. 3A is built and solved for the node. If it is satisfiable, the node is probed and disqualified key values are inferred. If the value of a key input becomes known, this key input is removed and replaced with a constant. This reduces the number of key inputs in the fan-in to future nodes, making more eligible for probing. Traversal halts when all remaining nodes in the fan-out for a key input have $> k_{cur}$ fan-in key inputs. Traversal restarts from the next key input until all are traversed. This ensures that every node in the circuit with $\leq k_{cur}$ fan-in key inputs is evaluated. The fewer the fan-in key inputs to a node, the more keys can be eliminated by probing it (Thm. 3.3). Hence, only nodes with the highest possible key elimination potential are considered, maximizing the disqualified keys per probe. k_{cur} is then incremented and traversal restarts. Termination occurs when k_{cur} reaches a user-defined limit, denoted η, which quantifies the fewest keys that must be eliminated. η guarantees that $\geq 2^{|\mathcal{K}|-\eta}$ keys will be disqualified per probe (Thm. 3.3). In this way, probes inducing insignificant leakage are ignored.

4.2 Attack 2: Multi-Node Probing with Max-SAT

Physical attack 1 sensitizes the oracle and probes a single node. While this is simple, it is inefficient because it ignores key leakage elsewhere in the die. To design a more efficient attack, we consider a scanning EOFM probe (rather than a fixed probe) that allows the entire die to be imaged with a single probe by raster-scanning over it. This setup is common, especially for failure analysis [15, 19, 22]. By using it, we can extract key leakage from multiple nodes with a single probe, improving efficiency and reducing the probes to unlock the circuit. For example, consider the circuit in Fig. 2A. By using the input sequence '00000' and '10100' we satisfy Eqn. 3 at Probed Nodes 2 and 3 simultaneously. This allows us to infer the full key ($key1$ from Node 2, $key2$ from Node 3) using a single probe by scanning both nodes. Conversely, our prior attack requires at least two probing events. This key extraction scenario for each probed node is in Fig. 2D. We refer to this as *multi-node probing*.

4.2.1 Formulating Multi-Node Probe as Weighted Max-SAT The previous SAT formulation in Fig. 3A is inadequate to guide a multi-node probe because it considers only a single node. To guide a multi-node probe we must identify inputs capable of satisfying Eqn. 3 for many nodes. We can modify Fig. 3A to reflect a multi-node scenario by appending the SAT formulation for multiple nodes. The resulting formulation, depicted in the red-shaded region of Fig. 4A, uses common inputs, X_1 and X_2, and independent subkeys, $K_{1,i}$ and $K_{2,i}$, for each node, where $i \in \{1 \dots N\}$ and N is the number of simultaneously probed nodes. Any inputs satisfying this larger formulation must satisfy all included nodes, allowing subkey information to be extracted from each with a multi-node probe. The limitation of this approach is that a solution must satisfy *every* probe-able node simultaneously. Based on circuit topology and probe resolution, this may be impossible. To remedy this, we propose mapping this formulation to a Weighted Max-SAT problem.

Weighted Max-SAT (WMS) is a maximization form of the Boolean SAT problem where each clause in the SAT formula is given a non-negative weight. The solution is an input that maximizes the combined weight of satisfied clauses. Taking a WMS approach provides a number of benefits. First, WMS is a maximization problem which means that solutions need not satisfy every node simultaneously. This guarantees that feasible probing configurations will be found (if they exist) and makes it possible to launch a physical attack regardless of topology. Second, there are many efficient heuristics to solve WMS [12, 18, 21]. This allows inputs/keys satisfying Eqn. 3 to be rapidly identified. Finally, because WMS is a maximization problem, cost functions can be tuned towards varied attacker goals.

To map multi-node probing onto a WMS problem, we implement three increasingly complex cost functions to arrive at our final formulation in Fig. 4A. First, let us consider the simplest cost function: maximizing the number of satisfied nodes. To reflect this, we start from the multi-node SAT formulation in the red-shaded region of Fig. 4A (i.e. appending Fig. 3A for multiple nodes) and assign a cost of 1 to satisfying each probe's final AND gate and a cost of 0 to all other clauses. This cost is labeled $cost_1$ in Fig. 4A. Whenever an input satisfies a variable $O_{1\dots N}$, Eqn. 3 will also be satisfied for that probe-able node. The solution to this WMS problem maps to the inputs, X_1 and X_2, that cause key leakage in the most nodes.

Our attack aims to maximize the keys eliminated by each probe. While the number of satisfied nodes is related to this, it is not the same. To update our WMS to reflect this, remember that at least $2^{|\mathcal{K}|-|\mathcal{K}_n|}$ keys can be eliminated by probing a node, where $|\mathcal{K}|$ is the total key inputs and $|\mathcal{K}_n|$ is the key inputs that fan-in to the node (Thm. 3.3). If we set the number of eliminated keys as the cost of satisfying each node in the WMS, rather than a fixed cost of 1, the resulting solution maximizes the keys eliminated per multi-node

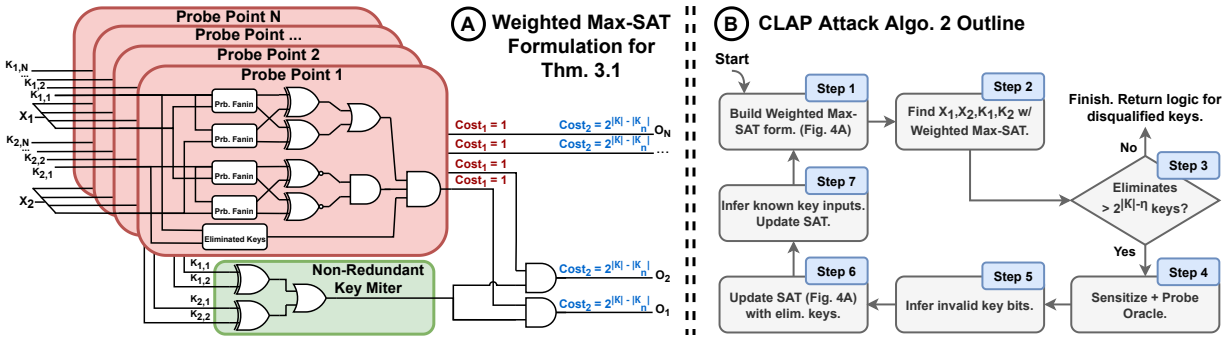

Figure 4: CLAP Attack Algorithm 2. A) Weighted Max-SAT formulation for multi-node probing. Solution identifies inputs (X_1, X_2) which optimally sensitize circuit for maximum key extraction via Thm. 3.1. B) Outline of physical CLAP attack algorithm 2.

probe. We depict the updated cost for satisfying each probe's final AND gate in blue, labeled as $cost_2$ in Fig. 4A. This replaces $cost_1$.

Finally, we modify our cost function to address key value redundancy between nodes. If two nodes eliminate the same key values, probing them both provides no additional attack progress. Hence, redundant probing should be excluded from the cost. We have included the green miter in Fig. 4A to address redundancy between probe point 1 and 2. The added logic prohibits probe point 1 and 2 from both contributing to the cost of a set of sensitizing inputs if they are satisfied by the same subkey values. This miter is only applied between nodes that share identical key inputs. Nodes with a non-identical key inputs always eliminate non-identical subkeys. Because this miter is not included between all nodes, we depicted it only between probe point 1 and 2 in Fig. 4A. Fig. 4A constitutes the final WMS formulation. This formulation finds inputs, X_1 and X_2, and subkeys, $K_{1,1..N}$ and $K_{2,1..N}$, such that the non-redundant keys eliminated by the multi-node probe is maximized.

4.2.2 Multi-Node Probing Attack Algorithm We now formalize our second physical attack algorithm. The goal is to eliminate the maximum keys from the keyspace with each multi-node probe. The configuration of each multi-node probe is the optimal (i.e. highest cost) solution to the WMS formulation in Fig. 4A. This maps to the probing configuration maximizing the non-redundant keys eliminated. Because multiple nodes are probed simultaneously, our multi-node probing algorithm eliminates more subkeys per probe than our prior attack, improving efficiency by requiring fewer probes to unlock the circuit. The attack is outlined in Fig. 4B.

A distinction between our physical attack algorithms is the increased diversity of nodes considered by the multi-node probing approach in Fig. 4B. While both algorithms only launch probing events disqualifying $\geq 2^{|\mathcal{K}|-\eta}$ keys, our multi-node probing attack can eliminate subkeys from multiple nodes with a single probe. As a result, we now consider nodes eliminating $< 2^{|\mathcal{K}|-\eta}$ keys (i.e. with more than η key inputs in the fan-in). This is because aggregating probing data from many of these nodes could eliminate $\geq 2^{|\mathcal{K}|-\eta}$ keys. Thus, our multi-node probing approach considers a more diverse set of nodes resulting in more key leakage. However, this increases attack complexity. To remedy this, we introduce the user-defined μ parameter to set the maximum number of fan-in key inputs to a node for it to be considered in the WMS formulation. By tuning μ, the attacker can exclude nodes that provide insignificant information, even in aggregate, reducing attack complexity.

At a high level, the attack iteratively constructs and solves the WMS formulation in Fig. 4A for all probe-able nodes with $\leq \mu$ key inputs in their fan-in. Each solution represents a set of inputs and a cost quantifying the number of keys eliminated by probing with these inputs. The oracle is then sensitized and probed with this configuration. Probing-disqualified subkeys are incorporated into the "Eliminated Keys" logic of successive WMS formulations. In this way, eliminated keys are not re-considered. This proceeds until probing can no longer eliminate $\geq 2^{|\mathcal{K}|-\eta}$ keys from the keyspace, causing termination. Similar to algorithm 1, η is user-defined.

5 Proposed CLAP Attack Algorithm

Physical attack strategies, including ours from Sec. 4, rely solely on physical leakage to infer the key. This limits them to a subset of the information leaked by the circuit. At some point, it will become cost inefficient or even infeasible to recover further key leakage due to factors such as the physical limitations of the probing setup or the topology of the circuit. A similar issue can be seen in a purely logical attack strategy as well (e.g. SAT attacks). For example, SAT-resilient structures make the effective progress of SAT-style attacks insignificant. Therefore, rather than relying on a purely logical or physical view to unlock a circuit as is done in prior work [2, 6, 10, 17, 24, 30–34], we instead combine both a logical and physical attack into a unified, mathematically-rigorous attack strategy. This allows us to exploit a broader spectrum of leakage than prior art. In this section, we outline the complete CLAP attack. This attack integrates our logically-guided physical attacks (Sec. 4.1.2/4.2.2) with a conventional SAT attack [34].

5.1 Integrating Logical and Physical Attacks

Let us consider how the results of our physical attack algorithms can be integrated into a SAT attack such that progress is maintained. This empowers a SAT attack to build upon prior physical attack results. As output, both physical attack algorithms produce a set of disqualified subkeys. The conventional SAT attack is driven by repeatedly solving the SAT-CNF formulation in Eqn. 2. Therefore, to link the physical and logical portions of the CLAP attack, we must integrate the set of subkeys eliminated by our physical attack strategy into Eqn. 2. This is done by converting the "Eliminated Keys" logic into CNF and appending it to Eqn. 2. This results in Eqn. 4, where the subkey eliminated by probe $m = 1 \ldots l$ is denoted K_e^m. Such an approach prohibits all physical-attack-eliminated keys from being incorporated into a valid distinguishing input (DI) for a SAT attack iteration. This excludes these physical-attack-eliminated keys from the keyspace without requiring a DI to do so, forcing the SAT attack to identify DIs capable of eliminating novel keys.

$$C^*(X_{di}, K_1, Y_1) \wedge C^*(X_{di}, K_2, Y_2) \wedge (Y_1 \neq Y_2)$$
$$\bigwedge_{j=1}^{i-1}(C^*(X_j, K_1, Y_j) \wedge C^*(X_j, K_2, Y_j)) \bigwedge_{m=1}^{l}((K_1 \neq K_e^m) \wedge (K_2 \neq K_e^m)) \quad (4)$$

To integrate logical SAT attack results into our physical attack, this process is performed in reverse. We must append the distinguishing input/output pairs (DIPs) identified in each SAT iteration to the "Eliminated Keys" logic used in the formulation described in Sec. 4.1.1 (and extended in Sec. 4.2.1). This forces any subkey identified as a solution to the logical formulation guiding the EOFM probe to also satisfy the DIPs identified by the logical SAT attack. This excludes previously eliminated keys, avoiding redundant work.

5.2 CLAP Attack Algorithm

We now formalize the complete CLAP attack. The CLAP attack combines the physical attack algorithms defined in Sec. 4.1.2 and 4.2.2 with the conventional SAT attack defined in [34]. Any key returned by the CLAP attack is necessarily functionally correct key (Thm. 5.1). The CLAP attack algorithm is outlined in Fig. 5.

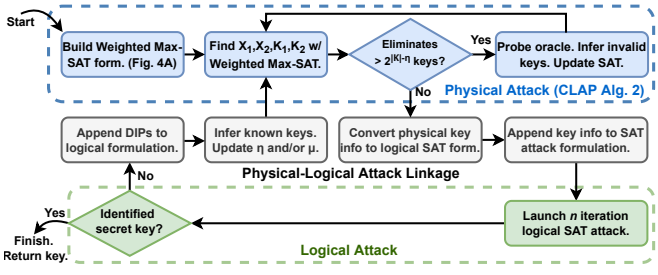

Figure 5: Outline of the unified CLAP attack algorithm.

From the start point in Fig. 5, the CLAP attack proceeds by launching one of our logically-guided physical probing attacks until termination. This occurs when a probing configuration can no longer be identified disqualifying at least $2^{|\mathcal{K}|-\eta}$ keys. This constitutes the physical component of the CLAP attack. It is outlined in the blue region of Fig. 5. At this point, the eliminated keys identified during the physical portion of the attack are converted into CNF

and appended to the SAT attack formulation for the circuit. The two-way linkage between the physical and logical portions of the CLAP attack is outlined in the gray region of Fig. 5. An n-iteration conventional SAT attack is then launched against the circuit. n is a user-defined parameter that allows the attacker to tune the balance between logical and physical attack strategies. This constitutes the logical portion of the CLAP attack and is outlined in the green region of Fig. 5. The SAT attack proceeds until either n DIs are located or no more DIs exist. In the former case, the identified DIs are appended to the "Eliminated Keys" logic guiding our physical attack strategy and the physical probing attack is re-launched. η and/or μ may need to be scaled to enable progress. If no more DIs exist, a key is located that satisfies all SAT-attack-identified DIs and is not in the set of physically-eliminated keys. This key is *functionally correct*, meaning it causes the locked circuit to produce output equivalent to the oracle for all inputs, as proved in Thm. 5.1.

Theorem 5.1. *The key returned by a CLAP attack is always correct.*

Proof. Proof by contradiction. Assume the CLAP attack has returned a wrong key, K_i. Thus, there exists some input, X_i, such that the output produced by the wrong key, K_i, and the correct key, K_c, differ ($Y_i \neq Y_c$) and Eqn. 5 is satisfied for the locked circuit, C.

$$C^*(X_i, K_i, Y_i) \wedge C^*(X_i, K_c, Y_c) \wedge (Y_i \neq Y_c) \quad (5)$$

For the CLAP attack to terminate with K_i, there must exist no more DIs for the circuit and K_i must satisfy all identified DIs. However, because X_i satisfies Eqn. 5 and K_i has not been eliminated from the keyspace, X_i must also satisfy Eqn. 4, constituting a valid DI. Thus, the CLAP attack could not terminate before selecting X_i as a DI, eliminating K_i from consideration. This is a contradiction. □

6 Evaluation of the CLAP Attack

To evaluate the CLAP attack, we implemented it. This implementation has been made open source [1]. The physical portion of the attack was developed as a custom function in the Berkeley ABC Synthesis Tool [4]. This implementation includes both physical attack algorithms (Sec. 4.1.2 and 4.2.2) for both probing setups (fixed and scanning probes). Our Weighted Max-SAT solver was implemented using a clustering-based approximation [12]. Our physical-logical attack linkage was implemented in ABC. The logical portion of the CLAP attack was built on the SAT attack in [34].

For evaluation, we used four benchmark circuits (c1908, c5315, des, b14) from ISCAS'85 [11], ITC'99 [8], and MCNC [38]. Four common obfuscation schemes were implemented in each circuit: Strong Logic Locking (SLL) [40], Anti-SAT [36] + SLL, Full-Lock [13], and SFLL-HD [27, 28, 41]. Each circuit was synthesized and mapped using the Synopsys Design Compiler with the Synopsys 90nm SAED library. Benchmark characteristics are in Tab. 1.

Table 1: Obfuscated benchmark circuit characteristics.

Benchmark				Locking Configuration Key Length (Bits)			
Name	PIs	Gates	POs	SLL	Anti-SAT+SLL	Full-Lock	SFLL-HD0
c1908	33	880	25	88	78	384	33
c5315	178	2307	123	231	156	540	178
des	256	5104	245	256	368	540	256
b14	277	9767	299	256	404	540	277

6.1 Evaluating Physical Portion of CLAP Attack

To evaluate the physical portion of the CLAP attack, we assumed that the minimal probe-able node size was a single gate. Note that Thm. 3.1, the derived basis for probing, is resolution-agnostic. This means the attacker can scale node size if technology scaling renders this unrealistic. Thus, while our results are specific to this assumption, the CLAP attack functions without the loss of generality. We also degrade our probe resolution in Sec. 6.1.2 to show that such degradation does not greatly impact attack performance.

The attack was launched with the user-defined η parameter set to 6. This ensures that no probe will be launched unless $\geq 2^{|\mathcal{K}|-\eta}$ keys can be eliminated. This is roughly 1.6% of the complete keyspace ($2^{|\mathcal{K}|-\eta}/2^{|\mathcal{K}|}$). For our multi-node probing algorithm (algorithm 2), we only considered probe-able nodes with ≤ 10 fan-in key inputs ($\mu = 10$). We launched both physical CLAP attack algorithms against all benchmarks. The key inputs fully/partially inferred via probing are in Fig. 6. The number of probing events and runtime is in Fig. 7.

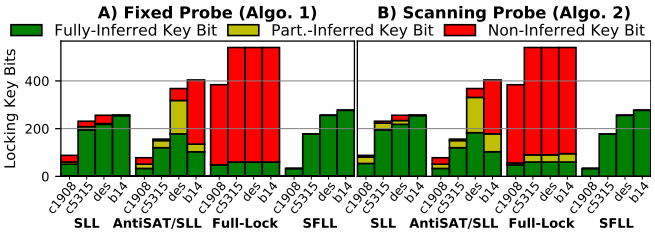

Figure 6: Number of fully and partially inferred key inputs by physical attack algorithm 1 (Sec. 4.1.2) and 2 (Sec. 4.2.2).

Both physical attack algorithms recovered the majority of key inputs. Algorithm 1 (fixed EOFM probe) fully recovered an average of 60.4% and partially recovered another 6.5% of key inputs. Algorithm 2 (scanning EOFM probe) fully recovered an average of 60.6% and partially recovered another 10.3% of key inputs. Each inferred key bit exponentially reduces the keyspace. Thus, both algorithms greatly reduced obfuscation complexity. However, for all but SFLL, the circuit was not fully unlocked. This motivates the need for a compound approach, such as CLAP, to fully unlock the circuit.

The recovered key bits varied between locking schemes. Our algorithms were weakest against Full-Lock and strongest against SFLL. This is due to locking scheme topology. Full-Lock uses cascaded layers of key-dense switch-boxes [13]. Empirically, both algorithms inferred a subset of key inputs for the first layer and ran out of eligible probe-able nodes by the second layer. This is because each switch-box from the second layer on has many switch-boxes in their fan-in, exceeding the $\eta = 6$ fan-in key limit and disqualifying these nodes. However, as we later show, the recovered first-layer key inputs sufficiently constrain the keyspace such that the whole key is recovered by the logical portion of the CLAP attack in <30 minutes. This is only possible because our logical/physical attacks build on each other, distinguishing the CLAP attack from prior art.

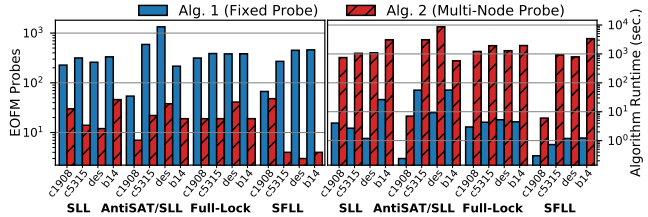

Figure 7: Number of EOFM probes and runtime for physical attack algorithm 1 (Sec. 4.1.2) and 2 (Sec. 4.2.2).

Based on Fig. 7, attack algorithm 1 and 2 provide a trade-off between required probes and runtime. Algorithm 1 (fixed EOFM probe) runs in 11 seconds, but requires 385 probes on average. Algorithm 2 (scanning EOFM probe) runs in 30 minutes, but requires only 21 probes on average. When only a fixed probe is available or the cost of probing is negligible, algorithm 1 outperforms 2. However, when probing has an associated cost, algorithm 2 increases compute time to more efficiently configure each probe.

6.1.1 Impact of Probing Countermeasures Probing countermeasures insert scattering/noisy structures alongside probe-able nodes to prevent information from being leaked [23, 31]. For example, Rahman et al. propose the insertion of 2 inverters alongside a probe-able die feature driven by inverted inputs to close the targeted side-channel [23]. A small number of these structures inserted on targeted die features (i.e. key register output buffers) are sufficient to mitigate prior art [24, 31]. However, a key contribution of the CLAP attack is the wide variety of die features which can be probed. To mitigate the CLAP attack, these countermeasures must protect any node through which key leakage can be extracted (i.e. satisfying Eqn. 3). To assess the overheads of this, we quantified the probe-able nodes in each benchmark. There exists an average of 775, 2000, 3137, and 5828 probe-able nodes for c1908, c5315, des, and b14. This means that the countermeasure in [23] would more than double the gates in each benchmark to prevent the physical portion of the CLAP attack. These overheads would be on-top of obfuscation. The diversity in probe-able die features considered by the CLAP attack forces mitigation strategies to protect large portions of the circuit, resulting in massive overheads and rendering these countermeasures infeasible against the CLAP attack.

6.1.2 Impact of EOFM Probe Resolution To quantify the impact of probe resolution on attack performance, we relax our assumption that any single gate can be resolved. Rather, we halve the resolution, requiring that we extract key leakage by probing two adjacent gates at once. This coarser resolution results in a more complex SAT formulation that must be satisfied. To evaluate the physical portion of the CLAP attack with this coarser resolution, we launched both attack algorithms on each benchmark. Compared to our prior results in Fig. 6, halving the effective resolution of the EOFM probe did not substantially degrade attack performance. On average, physical attack algorithm 1 (fixed EOFM probe) and 2 (scanning EOFM probe) fully recovered 58.6% and 58.9% and partially recovered 5.6% and 7.9% of key inputs, respectively. This compares to fully recovering 60.4% and 60.6% and partially recovering another 6.5% and 10.3% of key inputs with our original resolution. We draw two primary conclusions from this. First, these results empirically affirm our prior assertion that Thm.3.1-driven probing attacks are probe-resolution-agnostic. Second, the degradation of our attack algorithms is small despite halving our effective resolution, indicating that the CLAP attack will remain potent as technology scales.

6.2 Evaluating Logical Portion of CLAP Attack

To evaluate the logical portion of the CLAP attack, we launched it on each benchmark with the physical-attack-limited keyspace from the prior section. We set the limit on SAT attack iterations to $n = 200$. The secret key was recovered for all benchmarks before reaching this limit, causing attack termination. The SAT iteration count prior to CLAP attack termination is in Tab. 2. This data considers our logical attack strategy following both physical attack algorithms. We also quantified the SAT iterations to unlock each benchmark without the physical-attack-limited keyspace for comparison.

The logical portion of the CLAP attack recovered the correct key from the physical-attack-limited keyspace in less than 150 SAT iterations in all cases. Moreover, the logical portion of the CLAP attack ran quite quickly, on average taking only 109 and 101 seconds following physical attack algorithm 1 and 2. In the presence of partially-unlocked SAT-resilient (Anti-SAT and SFLL) locking, our logical attack strategy did not exhibit exponential SAT iterations in the length of the unspecified key. This indicates that the partially-unlocked keyspace provided by the physical portion of the CLAP attack was sufficiently specified to overcome the SAT-resilient nature of these schemes. We note a similar result for SAT-Hard instances (Full-Lock), where SAT iteration runtime did not scale exponentially between iterations for the physical-attack-limited keyspace.

Table 2: SAT iterations to unlock each circuit after physical portion of CLAP attack. Baseline column has data from SAT attack [34] against complete obfuscated circuit. Values $> 2^{16}$ are estimated with [36, 41]. Full-Lock iterations could not be estimated, so attacks > 100 hours timed-out (T/O).

	SLL			Anti-SAT + SLL		
Circuit	Baseline	Alg. 1	Alg. 2	Baseline	Alg. 1	Alg. 2
c1908	26	8	7	2^{16}	1	1
c5315	39	3	3	2^{16}	1	1
des	20	1	1	2^{16}	3	3
b14	11	1	1	2^{16}	31	28
	Full-Lock			**SFLL**		
Circuit	Baseline	Alg. 1	Alg. 2	Baseline	Alg. 1	Alg. 2
c1908	24	16	16	2^{32}	0	0
c5315	T/O	41	39	2^{177}	0	0
des	T/O	29	29	2^{255}	0	0
b14	T/O	145	145	2^{276}	0	0

6.3 Evaluating the Unified CLAP Attack

Finally, we consider the complete CLAP attack. The runtime of the entire attack strategy against all 16 benchmark circuits is in Fig. 8. We have also included the runtime of the conventional SAT attack [34] for comparison. Attack time-out (T/O) was 100 hours.

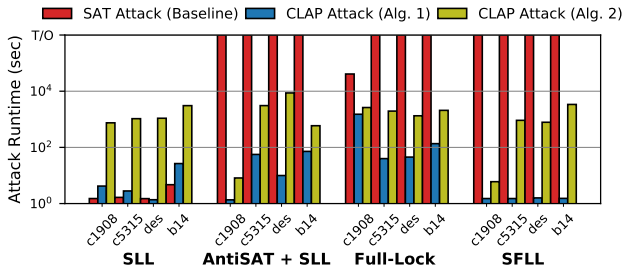

Figure 8: Runtime to unlock benchmark with CLAP attack.

The CLAP attack unlocked all benchmarks in <2.5 hours despite full-scale obfuscation configurations with large keys. Moreover, all three state-of-the-art obfuscation schemes (Anti-SAT, Full-Lock, and SFLL) were unlocked by the CLAP attack in exponentially less time than the SAT attack [34]. Many existing attacks are unable to feasibly unlock *all* of these benchmarks due to limitations in flexibility [37, 39] and scalability [2, 6, 10, 30, 32, 34]. Based on these results, the CLAP attack is a potent and scalable attack on obfuscation capable of unlocking circuits with varied topology/size.

7 Conclusions

We developed an open-source combined logical and physical (CLAP) attack on logic obfuscation. For the physical portion of the attack, we derived a theoretical basis for optical probing attacks to guide optimal key extraction. We proposed two physical attack algorithms based on this derivation. For the logical portion of the attack, we derived a process to incorporate key leakage from our physical attack into a logical SAT attack. To evaluate the CLAP attack, we launched it on four prominent obfuscation schemes. The physical portion of the attack fully recovered an average of 60.6% key inputs and partially recovered another 10.3%. The logical portion of the CLAP attack recovered a fully correct key from this physical-attack-limited keyspace in under 30 minutes in all cases.

8 Acknowledgement

This work was supported by the NSF under Grant 1953285.

References

[1] https://github.com/mzuzak/CLAP_Attack.git.

[2] Kimia Zamiri Azar, Hadi Mardani Kamali, Houman Homayoun, and Avesta Sasan. 2019. SMT attack: Next generation attack on obfuscated circuits with capabilities and performance beyond the SAT attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2019), 97–122.

[3] Christian Boit, Shahin Tajik, Philipp Scholz, Elham Amini, Anne Beyreuther, Heiko Lohrke, and Jean-Pierre Seifert. 2016. From ic debug to hardware security risk: The power of backside access and optical interaction. In *2016 IEEE 23rd International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*. IEEE, 365–369.

[4] Robert Brayton and Alan Mishchenko. 2010. ABC: An academic industrial-strength verification tool. In *International Conference on Computer Aided Verification*. Springer, 24–40.

[5] Abhishek Chakraborty, Nithyashankari Gummidipoondi Jayasankaran, Yuntao Liu, Jeyavijayan Rajendran, Ozgur Sinanoglu, Ankur Srivastava, Yang Xie, Muhammad Yasin, and Michael Zuzak. 2019. Keynote: A disquisition on logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 10 (2019), 1952–1972.

[6] Prabuddha Chakraborty, Jonathan Cruz, and Swarup Bhunia. 2018. SAIL: Machine learning guided structural analysis attack on hardware obfuscation. In *2018 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*. IEEE, 56–61.

[7] Jianqi Chen, Monir Zaman, Yiorgos Makris, RD Shawn Blanton, Subhasish Mitra, and Benjamin Carrion Schafer. 2020. DECOY: DEflection-Driven HLS-Based Computation Partitioning for Obfuscating Intellectual PropertY. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[8] Fulvio Corno, Matteo Sonza Reorda, and Giovanni Squillero. 2000. RT-level ITC'99 benchmarks and first ATPG results. *IEEE Design & Test of computers* 17, 3 (2000), 44–53.

[9] Sophie Dupuis and Marie-Lise Flottes. 2019. Logic locking: A survey of proposed methods and evaluation metrics. *Journal of Electronic Testing* 35, 3 (2019), 273–291.

[10] Mohamed El Massad, Siddharth Garg, and Mahesh V Tripunitara. 2015. Integrated Circuit (IC) Decamouflaging: Reverse Engineering Camouflaged ICs within Minutes.. In *NDSS*. 1–14.

[11] Mark C Hansen, Hakan Yalcin, and John P Hayes. 1999. Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering. *IEEE Design & Test of Computers* 16, 3 (1999), 72–80.

[12] Saurabh Joshi, Prateek Kumar, Ruben Martins, and Sukrut Rao. 2018. Approximation strategies for incomplete MaxSAT. In *International Conference on Principles and Practice of Constraint Programming*. Springer, 219–228.

[13] Hadi Mardani Kamali, Kimia Zamiri Azar, Houman Homayoun, and Avesta Sasan. 2019. Full-lock: Hard distributions of sat instances for obfuscating circuits using fully configurable logic and routing blocks. In *Proceedings of the 56th Annual Design Automation Conference 2019*. 1–6.

[14] Hadi Mardani Kamali, Kimia Zamiri Azar, Houman Homayoun, and Avesta Sasan. 2020. InterLock: An intercorrelated logic and routing locking. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 1–9.

[15] Ulrike Kindereit. 2014. Fundamentals and future applications of laser voltage probing. In *2014 IEEE International Reliability Physics Symposium*. IEEE, 3F–1.

[16] Ulrike Kindereit, Gary Woods, Jing Tian, Uwe Kerst, Rainer Leihkauf, and Christian Boit. 2007. Quantitative investigation of laser beam modulation in electrically active devices as used in laser voltage probing. *IEEE Transactions on Device and Materials Reliability* 7, 1 (2007), 19–30.

[17] Leonidas Lavdas, M Tanjidur Rahman, Mark Tehranipoor, and Navid Asadizanjani. 2020. On Optical Attacks Making Logic Obfuscation Fragile. In *2020 IEEE International Test Conference in Asia (ITC-Asia)*. IEEE, 71–76.

[18] Chu Min Li and Felip Manya. 2009. MaxSAT, hard and soft constraints. In *Handbook of satisfiability*. IOS Press, 613–631.

[19] Shih Yuan Liu, Hsin Hung Chou, Man Ting Pang, Kuang Yuan Chao, James CC Chang, Jian Chang Lin, and Chun Ming Chen. 2017. Laser voltage imaging and probing, efficient techniques for scan chain verification in advanced node. In *2017 IEEE 24th International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*. IEEE, 1–4.

[20] Yuntao Liu, Michael Zuzak, Yang Xie, Abhishek Chakraborty, and Ankur Srivastava. 2021. Robust and Attack Resilient Logic Locking with a High Application-Level Impact. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 17, 3 (2021), 1–22.

[21] António Morgado, Federico Heras, Mark Liffiton, Jordi Planes, and Joao Marques-Silva. 2013. Iterative and core-guided MaxSAT solving: A survey and assessment. *Constraints* 18, 4 (2013), 478–534.

[22] Yin S Ng, Ted Lundquist, Dmitry Skvortsov, Joy Liao, Steven Kasapi, and Howard Marks. 2010. Laser voltage imaging: A new perspective of laser voltage probing. In *ISTFA*. 5–13.

[23] M Tanjidur Rahman, Nusrat Farzana Dipu, Dhwani Mehta, Shahin Tajik, Mark Tehranipoor, and Navid Asadizanjani. 2021. CONCEALING-Gate: Optical Contactless Probing Resilient Design. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 17, 3 (2021), 1–25.

[24] M Tanjidur Rahman, Shahin Tajik, M Sazadur Rahman, Mark Tehranipoor, and Navid Asadizanjani. 2020. The key is left under the mat: On the inappropriate security assumption of logic locking schemes. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 262–272.

[25] Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri. 2014. A primer on hardware security: Models, methods, and metrics. *Proc. IEEE* 102, 8 (2014), 1283–1295.

[26] Akashdeep Saha, Sayandeep Saha, Siddhartha Chowdhury, Debdeep Mukhopadhyay, and Bhargab B Bhattacharya. 2020. LoPher: SAT-Hardened Logic Embedding on Block Ciphers. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[27] Abhrajit Sengupta, Mohammed Nabeel, Nimisha Limaye, Mohammed Ashraf, and Ozgur Sinanoglu. 2020. Truly stripping functionality for logic locking: A fault-based perspective. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 12 (2020), 4439–4452.

[28] Abhrajit Sengupta, Mohammed Nabeel, Muhammad Yasin, and Ozgur Sinanoglu. 2018. ATPG-based cost-effective, secure logic locking. In *2018 IEEE 36th VLSI Test Symposium (VTS)*. IEEE, 1–6.

[29] Bicky Shakya, Xiaolin Xu, Mark Tehranipoor, and Domenic Forte. 2020. CAS-lock: A security-corruptibility trade-off resilient logic locking scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2020), 175–202.

[30] Kaveh Shamsi, Meng Li, Travis Meade, Zheng Zhao, David Z Pan, and Yier Jin. 2017. AppSAT: Approximately deobfuscating integrated circuits. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 95–100.

[31] Haoting Shen, Navid Asadizanjani, Mark Tehranipoor, and Domenic Forte. 2018. Nanopyramid: An optical scrambler against backside probing attacks. In *ISTFA 2018: Proceedings from the 44th International Symposium for Testing and Failure Analysis*. ASM International, 280.

[32] Yuanqi Shen and Hai Zhou. 2017. Double DIP: Re-evaluating security of logic encryption algorithms. In *Proceedings of the on Great Lakes Symposium on VLSI 2017*. 179–184.

[33] Deepak Sirone and Pramod Subramanyan. 2020. Functional analysis attacks on logic locking. *IEEE Transactions on Information Forensics and Security* 15 (2020), 2514–2527.

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

[35] Shahin Tajik, Heiko Lohrke, Jean-Pierre Seifert, and Christian Boit. 2017. On the power of optical contactless probing: Attacking bitstream encryption of FPGAs. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1661–1674.

[36] Yang Xie and Ankur Srivastava. 2018. Anti-sat: Mitigating sat attack on logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38, 2 (2018), 199–207.

[37] Fangfei Yang, Ming Tang, and Ozgur Sinanoglu. 2019. Stripped functionality logic locking with Hamming distance-based restore unit (SFLL-hd)–unlocked. *IEEE Transactions on Information Forensics and Security* 14, 10 (2019), 2778–2786.

[38] Saeyang Yang. 1991. Logic synthesis and optimization benchmark user guide version 3.0. *MCNC, Jan. 1991* (1991).

[39] Muhammad Yasin, Bodhisatwa Mazumdar, Ozgur Sinanoglu, and Jeyavijayan Rajendran. 2017. Security analysis of anti-sat. In *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 342–347.

[40] Muhammad Yasin, Jeyavijayan JV Rajendran, Ozgur Sinanoglu, and Ramesh Karri. 2015. On improving the security of logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 9 (2015), 1411–1424.

[41] Muhammad Yasin, Abhrajit Sengupta, Mohammed Thari Nabeel, Mohammed Ashraf, Jeyavijayan Rajendran, and Ozgur Sinanoglu. 2017. Provably-secure logic locking: From theory to practice. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1601–1618.

[42] Michael Zuzak, Yuntao Liu, and Ankur Srivastava. 2020. Trace logic locking: Improving the parametric space of logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40, 8 (2020), 1531–1544.

[43] Michael Zuzak, Ankit Mondal, and Ankur Srivastava. 2021. Evaluating the Security of Logic-Locked Probabilistic Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2021).