Disguising Topology and Side-Channel Information through Covert Gate- and ML-Enabled IP Camouflaging

Junling Fan, David Koblah, Domenic Forte
Department of Electrical and Computer Engineering
University of Florida, Gainesville, USA
{fan.j, dkoblah}@ufl.edu, dforte@ece.ufl.edu

***Abstract*—Semiconductor intellectual property (IP) theft incurs hundreds of billions in annual losses, driven by advanced reverse engineering (RE) techniques. Traditional "cryptic" IC camouflaging methods typically focus on hiding localized gate functionality but remain vulnerable to system-level structural analysis. This paper explores "mimetic deception," where a functional IP (F) is designed to structurally and visually masquerade as a completely different appearance IP (A). We provide a comprehensive evaluation of three deceptive methodologies: IP Camouflage, Graph Matching, and DNAS-NAND Gate Array, analyzing their resilience against GNN-based node classification, and Differential Power Analysis (DPA). Crucially, we demonstrate that mimetic deception achieves a novel anti-side-channel defense: by forcing the mis-classification of cryptographic primitives, the adversary is led to apply an incorrect power model, causing the DPA attack to fail. Our results validate that this multi-layered approach effectively thwarts the entire RE toolchain by poisoning the structural and logical data used for netlist understanding.**

***Keywords*—Reverse Engineering, IC Camouflage, Cyber Deception, Machine Learning, Hardware Security**

I. Introduction

The integrity of modern electronic hardware is increasingly threatened by physically invasive reverse engineering (RE). By systematically delayering and imaging integrated circuits (ICs), attackers can reconstruct gate-level netlists to extract sensitive intellectual property (IP), identify security vulnerabilities, or facilitate illegal cloning. This poses catastrophic risks to critical domains, including defense and finance, where the exposure of underlying algorithms can undermine national and economic security [1], [2].

IC camouflaging has emerged as a key defense, utilizing fabrication-level secrets, such as dummy contacts [3], threshold voltage [4] or technology [5], to make different logic gates appear identical under Scanning Electron Microscope (SEM) imaging. However, existing methods often focus on localized gate modifications and fail to achieve broader system-level deception. This narrow focus allows advanced AI-enhanced RE tools and SAT solvers to eventually de-obfuscate the design.

This paper evaluates a holistic methodology known as mimetic deception [6], [7], which integrates cryptic and mimetic principles to ensure a circuit not only hides its true function but also presents a realistic yet misleading outward appearance. We evaluate three primary methods:

- **IP Camouflage:** An ML-driven approach utilizing an And-Inverter Graph Variational Autoencoder (AIG-VAE) to blend the functionality of a circuit with a deceptive appearance.
- **Graph Matching:** A layer-by-layer greedy heuristic that operates on standard logic gates to map a functional netlist onto a decoy topology while maintaining formal equivalence
- **DNAS-NAND Gate Array:** A scalable synthesis model using Differentiable Neural Architecture Search (DNAS) to generate end-to-end deceptive circuits within a single-stage flow.

Beyond traditional metrics of SAT resilience and PPA overhead, this work focuses on a unique defensive vector: side-channel misclassification. We analyze how mimetic deception can disguise cryptographic primitives to cause the failure of Differential Power Analysis (DPA). By making one cipher (e.g., PRESENT) appear as another (e.g., AES or DES), the attacker is misled into using an incompatible hypothetical power model. Since the success of DPA relies on the precise correlation between the power trace and the specific logic of the algorithm, this functional misclassification prevents the adversary from ever reaching the true secret key.

II. Background

A. IC Camouflaging and Covert Gates

IC camouflaging techniques traditionally fall into two categories: gate-level and interconnect-level [5]. Gate-level methods replace standard cells with variants that implement different functions based on fabrication secrets like dummy contacts [3] or threshold voltage variations [8]. Interconnect camouflaging [9] obscures signal paths by manipulating internal wiring.

The current state-of-the-art is the covert gate methodology [5]. Covert gates exploit always-on and always-off transistors to realize multiple gate functionalities within a single physical footprint. Because these transistor states are created through doping and remain indistinguishable under SEM imaging, covert gates appear identical to standard CMOS cells. This work utilizes specialized variants: Fake Inverters (FI), Fake Buffers (FB), and Universal Transmitters (UT), im-

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited. Award # W911NF-19-1-0102.

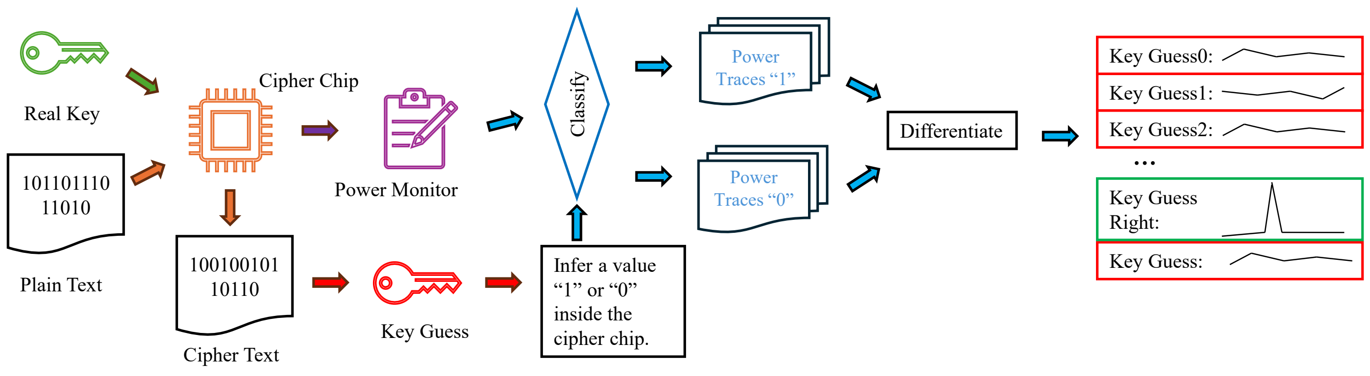

Fig. 1. Overview of a Differential Power Analysis (DPA) attack workflow targeting a cryptographic S-Box with a peak for correct key guess.

plementing function-appearance mismatches that thwart both visual inspection and SAT-based logical attacks [6].

B. IC Reverse Engineering and Netlist Understanding

Physically invasive reverse engineering (RE) is a destructive process where an attacker systematically dissects an IC to uncover its complete netlist [10], [11]. The typical workflow begins with decapsulation to expose the silicon die, followed by iterative deprocessing. During this phase, each metal and dielectric layer is removed through mechanical polishing or automated ion beam milling [12]. High-resolution SEM imaging is then used to capture the layout, which is subsequently analyzed to extract a gate-level netlist [11].

Once extracted, netlist understanding tools are employed to abstract high-level functions from the flattened "sea of gates" [13]. Algorithmic approaches identify recurring structures or infer control logic through Boolean reasoning [14], [15]. More recently, machine learning-based tools such as GNN-RE [16] and FGNN2 [17] interpret circuits as graphs to classify gate functions and functional boundaries. Mimetic deception aims to poison the structural data these tools rely on, forcing them to produce misleading architectural labels.

C. DPA Attack on Ciphers

Differential Power Analysis (DPA) is a statistical side-channel attack that exploits the correlation between a device's instantaneous power consumption and the intermediate data values it processes [18]. Unlike Simple Power Analysis (SPA), which relies on visual inspection, DPA utilizes large datasets to extract secret keys even amidst significant noise [19].

The attack is predicated on a power leakage model, typically the Hamming Weight (HW) or Hamming Distance (HD) model. As detailed by Prouff [19], the power consumption $C(X)$ of a device processing data X is modeled as proportional to the number of bit transitions (HD) or the number of set bits (HW). In block ciphers like AES or DES, the Substitution Box (S-Box) is the primary target due to its non-linearity, which relates plaintext, key, and ciphertext in a complex but deterministic manner.

To perform a DPA attack, an adversary defines a *selection function* $D(X, K, j)$, which predicts the value of the j-th bit of a specific intermediate state (e.g., the S-Box output) given a plaintext X and a hypothetical key K. The adversary then computes the *differential trace* Δ_K by partitioning the measured power traces into two sets based on whether D predicts a 0 or a 1, and calculating the difference of their means:

$$\begin{aligned}\Delta_K(j) = &\frac{\sum_{i=1}^{N} D(X_i, K, j)C(X_i)}{\sum_{i=1}^{N} D(X_i, K, j)} \\ &- \frac{\sum_{i=1}^{N}(1 - D(X_i, K, j))C(X_i)}{\sum_{i=1}^{N}(1 - D(X_i, K, j))}\end{aligned} \tag{1}$$

where N is the number of traces and $C(X_i)$ is the measured power consumption for the i-th input.

When the hypothetical key K matches the actual secret key, the selection function D accurately classifies the power traces, resulting in a statistically significant peak in Δ_K at the time instant the intermediate value is processed – see Fig. 1. Conversely, incorrect key hypotheses result in uncorrelated groupings, where the difference of means tends toward zero. However, as noted in [19], incorrect keys can sometimes produce "ghost peaks" due to residual correlations, though these are typically distinguishable from the correct key signal with sufficient data.

Crucially, the success of this methodology relies entirely on the correctness of the selection function D. The attacker must precisely know which cryptographic primitive (e.g., AES S-Box vs. PRESENT S-Box) is executing to accurately predict the intermediate bit j. If the target logic is obfuscated such that the attacker employs a selection function derived from a decoy S-Box, the predictions will fail to correlate with the physical power leakage, rendering the attack futile.

III. Deceptive Design Methodologies

To realize the paradigm of mimetic deception, we propose and evaluate three distinct methodologies. Each approach aims to synthesize a circuit that performs a target function (F) while mimicking the structural topology of a decoy appearance circuit (A). However, they differ fundamentally in their algorithmic foundations – ranging from heuristic graph theory

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited. Award # W911NF-19-1-0102.

to generative machine learning – and their respective trade-offs in scalability, overhead, and representation.

A. IP Camouflage

IP Camouflage is the foundational generative approach for mimetic deception [6]. Unlike heuristic matching, this method utilizes deep learning to capture and manipulate the latent functional representations of circuits.

The workflow employs a novel And-Inverter Graph Variational Autoencoder (AIG-VAE). The encoder utilizes an asynchronous message-passing mechanism and Gated Recurrent Units (GRUs) to encode the structural and functional dependencies of an AIG into a probabilistic latent space defined by a mean μ and variance σ [6]. To generate a deceptive circuit, the model interpolates between the latent vector of the functional circuit ($\mathbb{Z}_F$) and the appearance circuit ($\mathbb{Z}_A$) using a proportion factor $p \in [0, 1]$:

$$\mathbb{Z}_{G'} = (1-p)\mathbb{Z}_F + p\mathbb{Z}_A \tag{2}$$

The decoder then reconstructs a "blended" circuit from $\mathbb{Z}_{G'}$ by predicting node types, connections, and inverter placements [6].

Because the generative output is a probabilistic approximation, this method requires a critical "Post-Generation Rectification" phase to restore logical integrity. In this step, the blended circuit is compared against the target function F. Discrepancies are systematically resolved by inserting specialized covert gates:

- **Fake Inverters (FI) and Fake Buffers (FB):** Components that appear as standard inverters/buffers but function as constant logic or wires [6].
- **Universal Transmitters (UT):** A flexible component that appears as a NAND gate but can be configured to act as a buffer, inverter, or constant source [6].

While this approach effectively creates function-appearance mismatches, the reliance on post-generation fixes introduces area overhead and limits the method's scalability to smaller logic cones (approx. 200 nodes) [6].

B. Graph Matching

The Graph Matching method addresses the representation mismatch inherent in AIG-based approaches [6] by operating directly on standard logic gate netlists [7]. Unlike AIGs, which decompose logic into abstract AND/NOT nodes, this method processes the "real logic" representation found in standard Process Design Kits (PDKs). This ensures that the generated deceptive circuit is composed of standard cells (e.g., NAND, NOR, XOR) that are directly compatible with industrial design flows and standard reverse engineering tools.

The core of this method is a layer-by-layer greedy heuristic that maps nodes from the appearance graph G_A to the functional graph G_F [7]. The algorithm proceeds in three distinct phases:

1) **Graph Levelization:** Both the functional (G_F) and appearance (G_A) graphs are independently levelized using a topological sort (Kahn's algorithm). This assigns every node to a specific depth layer, establishing a structured basis for matching [7].
2) **Bipartite Matching:** The algorithm iterates through the circuit depth from inputs to outputs. At each layer k, it solves a minimum-cost bipartite matching problem to map nodes from $Layer_k(G_A)$ to $Layer_k(G_F)$ using the Hungarian Algorithm. This ensures a globally optimal assignment for that specific layer before propagating constraints to the next [7].
3) **Asymmetric Cost Optimization:** The matching decisions are driven by a specialized cost function designed to enforce deception. The cost $C_{i,j}$ of matching an appearance node a_i to a functional node f_j is the sum of two components:
   - *Node Cost:* A penalty is derived from the difficulty of camouflaging the logic function. For example, hiding a simple buffer or inverter within a complex NAND/NOR gate incurs zero cost, as supported by Covert Gate technology. However, matching incompatible logic types incurs a high penalty [7].
   - *Connection Cost:* This enforces a structural containment constraint based on the matching from the previous layer (M_{prev}). A high penalty is assigned if a connection exists in the functional graph G_F but the corresponding mapped nodes in G_A are not connected. This effectively encourages the connectivity of G_F to be a structural subset of G_A, allowing the functional wires to be ”hidden” within the appearance topology [7].

C. DNAS-Based Gate Array Optimization

To overcome the scalability limitations of the AIG-VAE and the representation constraints of standard AIGs, we introduced the DNAS-NAND Gate Array method [7]. This approach leverages Differentiable Neural Architecture Search (DNAS) to synthesize an end-to-end deceptive circuit without the need for post-generation rectification.

We proposed a unified model named "SelectorTNet", which extends the T-Net [20] architecture to learn two distinct logic functions simultaneously. The network incorporates a global selector variable p that governs the effective parameters θ_{eff} of the network during the forward pass:

$$\theta_{eff} = p \cdot \theta_{Functional} + (1-p) \cdot \theta_{Appearance} \tag{3}$$

When $p = 1$, the network parameters adapt to synthesize the functional circuit; when $p = 0$, they adapt to the appearance circuit. The use of Gumbel-Softmax sampling ensures the selection of connections remains differentiable [7].

The training is guided by a composite loss function that jointly optimizes for correctness and mimicry:

- **Hardness-Aware Loss:** Ensures 100% logical correctness for both F (at $p = 1$) and A (at $p = 0$) by penalizing "hard" examples where the network's prediction diverges from the truth table [7].

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited. Award # W911NF-19-1-0102.

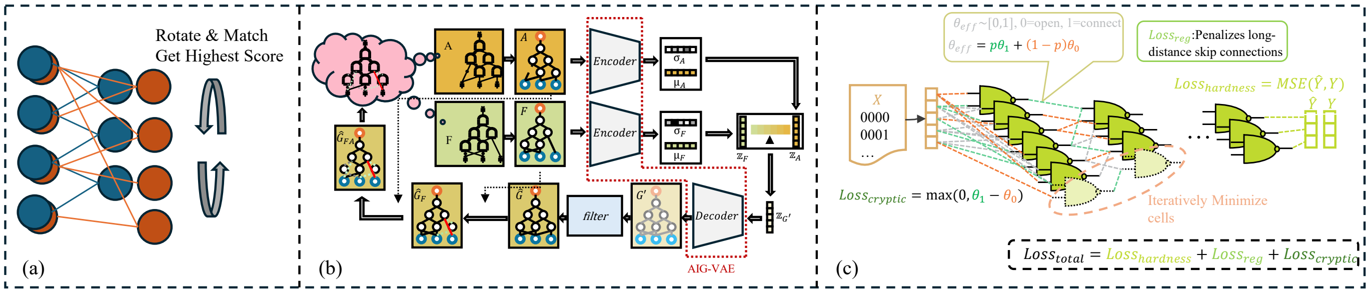
Rotate & Match
Get Highest Score
(a)
A
F
Encoder
Encoder
filter
Decoder
AIG-VAE
(b)
$\theta_{eff}\sim[0,1]$, 0=open, 1=connect
$\theta_{eff} = p\theta_1 + (1-p)\theta_0$
$Loss_{reg}$:Penalizes long-distance skip connections
$Loss_{hardness} = MSE(\hat{Y}, Y)$
X
0000
0001
…
$Loss_{cryptic} = \max(0, \theta_1 - \theta_0)$
Iteratively Minimize cells
$Loss_{total} = Loss_{hardness} + Loss_{reg} + Loss_{cryptic}$
(c)

Fig. 2. Overview of the three proposed deceptive design methodologies: (a) Graph Matching using standard gates, (b) IP Camouflage using AIG-VAE interpolation, and (c) DNAS-Based NAND Array optimization.

TABLE I
COMPARISON OF DECEPTIVE DESIGN METHODOLOGIES

Feature	Graph Matching	IP Camouflage	DNAS-NAND
Core Algo.	Greedy Heuristic	AIG-VAE	DNAS
Logic Basis	Standard Gates	AIGs	NAND Array
Scalability	High	Low ($<$ 200 nodes)	High
Rectification	Not Required	Required	Not Required
Formal Eq.	Guaranteed	Guaranteed	Verified ex-post
Target	Arithmetic/Cipher	Arithmetic/Cipher	Arithmetic

- **Inner-Architecture Regularized Loss:** Penalizes long-distance skip connections to ensure the synthesized netlist is physically realizable and efficient [7].
- **Cryptic Loss (Connection Containment):** This is the core mechanism for deception. It penalizes any connection in the functional mode ($p = 1$) that exhibits a stronger activation magnitude than its counterpart in the appearance mode ($p = 0$). The loss is defined as:

$$Loss_{cryptic} = \max(0, \theta_1 - \theta_0) \tag{4}$$

This constraint effectively forces the functional circuit to be "hidden" within the wiring of the appearance circuit. The result is a single NAND-gate array that satisfies the functional requirements while structurally appearing as the decoy, achieving high scalability and ultra-low overhead [7].

D. Summary of Methodologies

Table I summarizes the key characteristics of the three proposed methods. Graph Matching excels in formal correctness and standard representation, but offers moderate overhead due to strict matching constraints. IP Camouflage pioneered the generative approach but is limited by the rectification bottleneck. The DNAS-NAND Array offers the best scalability and PPA efficiency, albeit with a homogeneous NAND-only representation and difficulty in accurate representation of all functions such as ciphers.

In this work, we target the S-Boxes of varying Ciphers, distinct for their high non-linearity. Although test-vector based synthesis [7], [20], [21] is efficient for arithmetic circuits, it struggles to encompass the complex behavior of S-Boxes without incurring unacceptable hardware costs. As demonstrated in [22], the number of neurons required to approximate a function grows exponentially with its number of oscillations. For highly non-linear S-Boxes, this theoretical bound manifests as a massive area overhead, rendering standard learning-based synthesis ineffective.

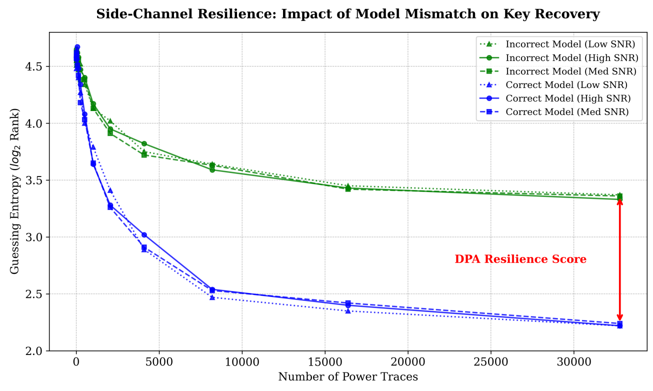

Fig. 3. Evaluation of DPA Resilience under Mimetic Deception. The plot compares the Guessing Entropy (GE) of a target S-Box against an attacker using the correct power model (Blue/Baseline) versus an incorrect model induced by deception (Green/Deceptive). While the baseline attack rapidly converges to the correct key (low entropy) within 2,000 traces, the deceptive design maintains high entropy even after 32,000 traces, preventing successful key recovery. The red arrow indicates the DPA Resilience Score, quantifying the sustained protective gap achieved by forcing a statistical model mismatch.

IV. EXPERIMENTS

We evaluate the efficacy of the proposed mimetic deception methodologies, IP Camouflage (AIG) and Graph Matching (GM), against two primary threats: automated reverse engineering via Graph Neural Networks (GNNs) and recovering secret key by Differential Power Analysis (DPA).

A. Experiment Setup & Scope

Our experiments target the highly non-linear Substitution Boxes (S-Boxes) of cryptographic primitives, specifically PRESENT (4-bit), DES (6-bit to 4-bit), and AES (8-bit).

Synthesis and Attack: All designs were camouflaged, modeled and synthesized using SAED 90nm Cell Library. GNN-based structural analysis was performed using a state-of-the-art

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited. Award # W911NF-19-1-0102.

TABLE II
COMPARISON OF DECEPTION METRICS AND OVERHEAD: IP CAMOUFLAGE (AIG) VS. GRAPH MATCHING (GM)

Scenario ($F \rightarrow A$)	**Method**	**Deception Scores**		**GNN F1 Scores**		**DPA Key Hiding**	**Overhead (Normalized)**	
		$Score_{GNNRE}$	$Score_{DPA}$	$F1_{expose}$	$F1_{mimicry}$	$Rank_{Hidden}$	*Area*	*Power*
$PRESENT \rightarrow DES$ (Avg)	AIG	3.32	0.47	0.10	0.37	34.1/64	1.12	1.11
	GM	9.40	0.44	0.05	0.32	33.6/64	1.14	1.25
$PRESENT \rightarrow AES$	AIG	18.0	0.43	0.05	0.95	115.2/256	1.02	1.02
	GM	7.40	0.44	0.10	0.84	120.2/256	1.05	1.17
$DES \rightarrow AES$ (Avg)	AIG	∞ (High)	0.45	0.005	0.90	136.1/256	1.07	1.07
	GM	84.6	0.38	0.015	0.89	124.6/256	1.11	1.12

Note: $F \rightarrow A$ denotes Functional Circuit disguised as Appearance Circuit. 'Avg' represents the mean across all 8 S-Box variants ($S1 - S8$). Score_GNNRE '∞' indicates the classifier failed to converge or produced random guess equivalent results. In best scenario, $Rank_{Hidden} = 0.5$.

GNN-RE classifier [16]. Side-channel resilience was evaluated using a Differential Power Analysis (DPA) attack framework.

Exclusion of DNAS-NAND Method: While section III introduced the DNAS-NAND Gate Array as a scalable candidate for arithmetic circuits, we exclude it from this experimental evaluation of S-Boxes. As theoretically argued in subsection III-D, emulating highly non-linear functions (like cryptographic S-Boxes) with a shallow neural-based synthesis model requires a prohibitively large number of neurons to capture the high-frequency oscillations. Preliminary trials confirmed that trying to synthesize these S-Boxes via DNAS resulted in area overhead exceeding $10\times$, rendering the method impractical for cryptographic primitives compared to AIG and GM.

B. Side-Channel Resilience Analysis

To quantify the mimetic deception effect on side-channel security, we disguised a PRESENT S-Box to structurally appear as a DES S-Box. We then performed a DPA attack assuming the adversary relies on the visual appearance to select the power model. Figure 3 plots the Guessing Entropy (GE) defined as the expected rank of the secret key against the number of power traces (up to 32k).

- **Baseline (Correct Model):** When the attacker uses the correct power model (Blue lines), the attack succeeds rapidly. The guessing entropy drops logarithmically, approaching $\approx$ 2.2 bits (rank $<$ 5) with 32k traces, indicating the key is effectively recovered.
- **Deceptive (Incorrect Model):** When the attacker is misled into using the model of the appearance circuit (Green lines), the attack fails to converge. Even with 32k traces, the guessing entropy remains high ($\approx$ 3.4 bits), meaning the correct key remains hidden among a large pool of candidates.

C. Comparative Analysis: AIG vs. GM

In this section, we evaluate the performance of the proposed methods using two novel security metrics designed to quantify the effectiveness of mimetic deception in both the side-channel and structural domains.

1) Side-Channel Resilience (DPA Score): To measure the "protective gap" created by the function-appearance mismatch as shown in Figure 3, we define the **DPA Resilience Score** as the normalized divergence in key ranking between the correct and incorrect power models. It is calculated as:

$$Score_{DPA} = \frac{Rank_{disguise} - Rank_{leak}}{Rank_{max}} \tag{5}$$

where $Rank_{leak}$ is the final rank of the secret key under the correct model (baseline), $Rank_{disguise}$ is the rank under the deceptive model, and $Rank_{max}$ is the size of the key hypothesis space (e.g., 64 for DES, 256 for AES).

Ideally, this score approaches **0.5**. This theoretical maximum is achieved when the baseline attack succeeds perfectly ($Rank_{leak} \approx 0$) while the deceptive attack forces the adversary into a state of random guessing, where the key rank settles at the statistical median ($Rank_{disguise} \approx 0.5 \cdot Rank_{max}$). A score of 0.5 indicates the key is hidden in a "middle ground," indistinguishable from noise.

As shown in Table II, both methods achieve scores consistently in the range of **0.38 – 0.47**. This confirms that regardless of the synthesis method (Generative AIG or Heuristic GM), the topological deception successfully masks the leakage, keeping the key rank high and preventing convergence.

2) Structural Deception (GNN Score): To quantify resilience against automated structural analysis, we utilize the **GNN Deception Score** [7] as defined in Eq. (3):

$$Score_{GNN} = \frac{F1_{mimicry} - F1_{expose}}{F1_{expose}} \tag{6}$$

where $F1_{expose}$ measures the accuracy of the GNN in identifying the hidden functional logic, and $F1_{mimicry}$ measures its success in misclassifying the logic as the decoy appearance. A higher score indicates that the tool is significantly more likely to be misled by the outer topology (A) than to uncover the inner functionality (F).

- **IP Camouflage (AIG)** demonstrated peak performance, achieving "Infinite" scores in several DES$\rightarrow$AES test cases. This occurs as $F1_{expose} \rightarrow 0$, representing total obfuscation where the GNN fails to identify the functional logic entirely.
- **Graph Matching (GM)** yielded consistent high scores (e.g., ≈ 84.6 for DES$\rightarrow$AES), proving that standard-gate mapping effectively poisons the structural features relied upon by RE classifiers.

3) Overhead and Fidelity Trade-offs: We evaluate PPA metrics normalized against the *Appearance Circuit* (A) to quantify the deviation from the target topology. A value of $1.0\times$ represents ideal mimicry where the deceptive circuit is physically indistinguishable from the decoy.

- **IP Camouflage (AIG)** demonstrates superior density, maintaining an area profile near-identical to the target ($1.02\times - 1.12\times$). This efficiency stems from the AIG-VAE's latent-space interpolation, which naturally generates compact structures.
- **Graph Matching (GM)** incurs slightly higher deviations ($1.14\times$ Area, $1.25\times$ Power) due to the strict bipartite matching constraints. To guarantee formal equivalence using standard gates, GM often requires the insertion of additional dummy chains (covert gates [5]) to satisfy the connectivity requirements of the appearance graph, resulting in a slight bloating of the power footprint.

V. Conclusion

This work presents a comprehensive evaluation of "mimetic deception," a defensive paradigm that decouples a circuit's logical function from its physical topology to thwart both reverse engineering and side-channel analysis. By disguising the S-Boxes of vulnerable ciphers (e.g., PRESENT, DES) to structurally mimic robust or unrelated primitives (e.g., AES), we successfully mislead the automated toolchains used by adversaries.

Our experimental results validate two critical defensive capabilities. First, we demonstrated that structural deception breaks the fundamental assumption of DPA. By coercing the attacker into selecting an incorrect power model, the correlation peaks required for key recovery are suppressed, yielding a DPA Resilience Score of ≈ 0.45 (near the theoretical ideal of random guessing). Second, we showed that GNN classifiers are effectively poisoned by the deceptive topology, misclassifying functional logic with high confidence.

Comparison of the proposed methodologies reveals a distinct trade-off: while IP Camouflage (AIG) offers superior area efficiency and maximum structural obfuscation, Graph Matching (GM) provides a formally equivalent, standard-cell compliant alternative suitable for rigorous design flows. Ultimately, these techniques establish a new frontier in hardware security, where protection is achieved not just by hiding the logic, but by actively lying about it.

Acknowledgment

This work was supported by the US Army Research Office (ARO) under award #W911NF-19-1-0102.

References

[1] S. E. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *ACM journal on emerging technologies in computing systems (JETC)*, vol. 13, no. 1, pp. 1–34, 2016.

[2] "Global Research and Analytics Firm — Investment, Business, Intellectual Property Research and Business Valuation Services — Aranca — aranca.com," https://www.aranca.com/knowledge-library/articles/investment-research/ip-theft-china-and-beyond, [Accessed 06-03-2025].

[3] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 709–720.

[4] N. E. C. Akkaya, B. Erbagci, and K. Mai, "A secure camouflaged logic family using post-manufacturing programming with a 3.6 ghz adder prototype in 65nm cmos at 1v nominal v dd," in *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2018, pp. 128–130.

[5] B. Shakya, H. Shen, M. Tehranipoor, and D. Forte, "Covert gates: Protecting integrated circuits with undetectable camouflaging," *IACR transactions on cryptographic hardware and embedded systems*, pp. 86–118, 2019.

[6] J. Fan, D. Koblah, and D. Forte, "Designing with deception: Ml-and covert gate-enhanced camouflaging to thwart ic reverse engineering," *arXiv preprint arXiv:2508.08462*, 2025.

[7] J. Fan, G. Rushevich, G. Rusconi, M. Zhu, R. Dizon-Paradis, and D. Forte, "Scalable ip mimicry: End-to-end deceptive ip blending to overcome rectification and scale limitations of ip camouflage," *arXiv preprint arXiv:2512.12061*, 2025.

[8] B. Erbagci, C. Erbagci, N. E. C. Akkaya, and K. Mai, "A secure camouflaged threshold voltage defined logic family," in *2016 IEEE International symposium on hardware oriented security and trust (HOST)*. IEEE, 2016, pp. 229–235.

[9] C. Yu, X. Zhang, D. Liu, M. Ciesielski, and D. Holcomb, "Incremental sat-based reverse engineering of camouflaged logic circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 10, pp. 1647–1659, 2017.

[10] R. Torrance and D. James, "The state-of-the-art in semiconductor reverse engineering," in *Proceedings of the 48th Design Automation Conference*, 2011, pp. 333–338.

[11] S. E. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *ACM journal on emerging technologies in computing systems (JETC)*, vol. 13, no. 1, pp. 1–34, 2016.

[12] E. L. Principe, N. Asadizanjani, D. Forte, M. Tehranipoor, R. Chivas, M. DiBattista, S. Silverman, M. Marsh, N. Piche, and J. Mastovich, "Steps toward automated deprocessing of integrated circuits," in *International Symposium for Testing and Failure Analysis*, vol. 81504. ASM International, 2017, pp. 285–298.

[13] N. Albartus, M. Hoffmann, S. Temme, L. Azriel, and C. Paar, "Dana universal dataflow analysis for gate-level netlist reverse engineering," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 309–336, 2020.

[14] P. Subramanyan, N. Tsiskaridze, W. Li, A. Gascón, W. Y. Tan, A. Tiwari, N. Shankar, S. A. Seshia, and S. Malik, "Reverse engineering digital circuits using structural and functional analyses," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 1, pp. 63–80, 2013.

[15] T. Meade, S. Zhang, and Y. Jin, "Netlist reverse engineering for high-level functionality reconstruction," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2016, pp. 655–660.

[16] L. Alrahis, A. Sengupta, J. Knechtel, S. Patnaik, H. Saleh, B. Mohammad, M. Al-Qutayri, and O. Sinanoglu, "Gnn-re: Graph neural networks for reverse engineering of gate-level netlists," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 8, pp. 2435–2448, 2021.

[17] Z. Wang, C. Bai, Z. He, G. Zhang, Q. Xu, T.-Y. Ho, Y. Huang, and B. Yu, "Fgnn2: A powerful pre-training framework for learning the logic functionality of circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024.

[18] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Annual international cryptology conference*. Springer, 1999, pp. 388–397.

[19] E. Prouff, "Dpa attacks and s-boxes," in *International Workshop on Fast Software Encryption*. Springer, 2005, pp. 424–441.

[20] Z. Wang, J. Wang, Q. Yang, Y. Bai, X. Li, L. Chen, J. Hao, M. Yuan, B. Li, Y. Zhang *et al.*, "Towards next-generation logic synthesis: A scalable neural circuit generation framework," *Advances in Neural Information Processing Systems*, vol. 37, pp. 99 202–99 231, 2024.

[21] P. Belcak and R. Wattenhofer, "Neural combinatorial logic circuit synthesis from input-output examples," *arXiv preprint arXiv:2210.16606*, 2022.

[22] M. Telgarsky, "Benefits of depth in neural networks," in *Conference on learning theory*. PMLR, 2016, pp. 1517–1539.