証明可能安全ロジックロッキング：安全性定式化の新たな枠組みとその実現

渡邉 洋平[1,2,a)]　淺野 京一[1,2]　平田 遼[1]　小野 知樹[1]　楊 明宇[3]　岩本 貢[1]　李 陽[1]　原 祐子[3]

概要：ロジックロッキング (Logic Locking: LL) 技術とは，回路の IP 保護と IC チップ製造の外部委託を両立する暗号技術であり，半導体業界のファブレス化が進む背景から盛んに研究が進められている．しかし，多くの既存 LL 方式の安全性は実験的に示されており，方式が提案されては攻撃が見つかる “いたちごっこ” が続いている．いくつかの既存研究では LL 技術の証明可能安全性に取り組んでいるものの，定式化の不備や莫大な回路オーバーヘッドといった課題が残る．そこで本稿では，証明可能安全 LL 技術の安全性定式化に関する新たな枠組みを提案する．提案する枠組みは，任意の攻撃に対して特定の漏洩 $\mathcal{L}$ 以上の情報を漏らさないことを保証する安全性の定式化と，その漏洩 $\mathcal{L}$ がどの程度具体的な攻撃に影響があるのかを示す攻撃耐性の定式化の二段階からなる．特に，我々は回路のトポロジが漏洩した場合の具体的な攻撃耐性を導出した上で，トポロジの漏洩のみを許す新たな証明可能安全 LL 方式 PSYLOCKE を提案する．また，実装実験を通じた性能評価を行い，PSYLOCKE が代表的な攻撃に耐性があること，及び面積や電力，遅延のオーバーヘッドも既存の証明可能安全 LL 方式に比べて大幅に効率化されていることを示す．

Provably Secure Logic Locking: A New Framework for Security Formalization and Its Realization

Yohei Watanabe[1,2,a)]　Kyoichi Asano[1,2]　Haruka Hirata[1]　Tomoki Ono[1]　Mingyu Yang[3]
Mitsugu Iwamoto[1]　Yang Li[1]　Yuko Hara[3]

Abstract: Logic locking is a cryptographic technique that protects the intellectual property (IP) of hardware designs while enabling outsourced IC manufacturing. A key challenge is the ongoing cat-and-mouse game between attackers and defenders, driven by the heuristic nature of existing designs. While recent work has explored *provably secure* logic locking, issues like high implementation overhead remain unresolved. In this paper, we introduce a new framework for the security formalization of logic locking. In this paper, we propose a new two-layer security framework: a general formalization that guarantees no information beyond a predefined leakage $\mathcal{L}$ is exposed, and a resilience layer that quantifies how much $\mathcal{L}$ can aid specific attacks. We show a concrete resilience against the circuit topology leakage and propose a new logic locking scheme, PSYLOCKE, resilient to the topology leakage. Moreover, we demonstrate that PSYLOCKE indeed has the resilience against well-known attacks and is significantly more efficient than previous provably secure schemes.

1. はじめに

半導体関連技術の向上に伴い，IC 製造設備を管理及び維持するには莫大なコストがかかるため，多くの半導体メーカでファブレス化が進み，IC チップの製造を外部のファウンドリに委託し IC チップの製造を行っている．しかし，委託に伴い，リバースエンジニアリングや偽造，回路設計情報の盗用等，半導体メーカは様々な脅威に直面している [15]．特に回路設計情報は各メーカにとって商用的価値の高い IP（Intellectual Property）情報であり，ファブレス化と IP 情報保護の両立は重要な問題である．

ロジックロッキング（Logic Locking: LL）はこの問題への解決策として期待されている暗号技術であり，これまでに

[1] 電気通信大学 / The University of Electro-Communications
[2] 国立研究開発法人 産業技術総合研究所 / AIST
[3] 東京科学大学 / Institute of Science Tokyo
[a)] watanabe@uec.ac.jp

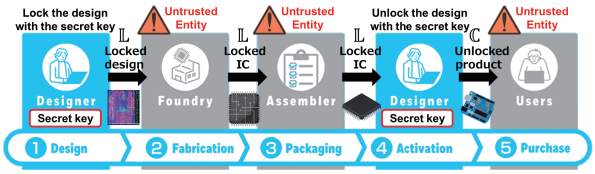


図 1: 簡略化した IC サプライチェーン.

Fig. 1 A simplified IC supply chain.

盛んに研究が進められてきた [2], [3], [8], [9], [13], [14], [29]. LL では，回路 $\mathbb{C}$ を秘密鍵を用いて “ロック” し，回路 $\mathbb{L}$ を生成する．図 1 にあるように，$\mathbb{L}$ のデザインをファウンドリに提供し，ロックされた IC チップを製造してもらうことで，IP 情報を漏洩することなくファブレス化を達成できる．この時，鍵を知らない限り $\mathbb{L}$ から元の回路 $\mathbb{C}$ を復元することはできず，鍵を知る半導体メーカは IC チップをアクティベートし，回路 $\mathbb{C}$ の機能性を復元できる．

証明可能安全 LL．多くの LL 方式（例えば [1], [8], [9], [15], [16], [17], [18], [19]）の性能及び安全性は実験的に評価されており，方式が提案されては攻撃が発見される “いたちごっこ” が繰り返されている．そのような現状に鑑みて進められているのが**証明可能安全 *LL* 研究** [3], [5], [13], [14], [23], [29], [30], [31] であり，具体的には次の 2 つの安全性定式化手法（と課題）が知られている．

- **攻撃耐性** [5], [6], [13], [23], [29], [30]：SAT 攻撃 [12], [24] 等，特定のクラスの攻撃に対する方式の頑強性を解析することに焦点を当てた安全性．本要件を満たす効率的な構成は多数提案されているものの，対象外のクラスの攻撃には何も安全性が保証されていない．実際，攻撃耐性の意味で “証明可能安全” な LL 方式の多くは構造攻撃によってその安全性が破られている．また，多くの定式化に不備があることも指摘されている [3].
- **全般的安全性** [3], [13], [14], [31]：特定の攻撃クラスではなく，広範な攻撃に対して定式化された安全性．特に，“最小限の漏洩（回路の入出力サイズ）を除き回路 $\mathbb{C}$ の情報が一切漏れない” ことを保証する非常に強い安全性を対象としているものが多い [3], [13], [14]. 結果として，どんな攻撃に対しても安全であるという非常に強い理論保証を与える一方で，それを満たす構成は効率性及び実用性に非常に乏しい．事前に定めた漏洩以外に一切情報が漏れないことを保証する定式化 [31] も知られているが，その許容した漏洩に実際どのような影響があるのかについては考えられていない.

したがって，**広範な攻撃クラスを対象にしつつも効率的に実現可能な証明可能安全 *LL* はいまだに知られていない**.

1.1 本稿の貢献

本研究では，上述した課題に対し，広範な攻撃クラスに対して証明可能安全かつ効率的に実装可能な LL 方式を初めて提案する．そのために，そのような LL 方式を実現可能な新たな安全性定式化の枠組みを導入し，その安全性を満たす新たな証明可能安全 LL 方式を提案する.

LL 安全性定式化の新たな枠組み（3 節）．本研究では，最小限の漏洩に対してだけではなく，特定の（許容可能な）漏洩の下であっても，その安全性を定量化できるような安全性の枠組みの構築を目指す．我々の目標は，証明可能安全性と実用的な効率性を両立するような新たな LL 方式の実現である．そこで，次のような新たな枠組みを提案する.

(i) 事前に定めた任意の “許容可能な” 漏洩 $\mathcal{L}$ の下での全般的安全性を定義する．これにより，**任意の攻撃者**に対して，LL 方式のアルゴリズムやロック後の回路 $\mathbb{L}$ から $\mathcal{L}$ 以上の情報が一切漏れないことが保証される.

(ii) 攻撃耐性を再考し，特定の攻撃，特にそれぞれ SAT 攻撃と構造攻撃を含む重要な攻撃クラスである**鍵復元攻撃**と**回路復元攻撃**に対して，攻撃成功に必要なオラクルアクセス数を尺度に用いた新たな攻撃耐性を定義する．上記の全般的安全性によって既に LL 方式が漏洩し得る情報 $\mathcal{L}$ について解析できていることから，提案する攻撃耐性定義は漏洩 $\mathcal{L}$ のみによって特徴づけられる．この点が従来の定義との大きな違いであり，これによって Lock アルゴリズムの構成とは独立に解析が可能である．すなわち，各 LL 構成に対して個別に攻撃耐性の解析を行う必要はなく，特定の漏洩 $\mathcal{L}$ を許す構成であれば共通の解析を適用可能である.

漏洩による影響の定量化（4 節）．我々の新たな攻撃耐性の下で，具体的な漏洩が実際に鍵/回路復元攻撃にどの程度耐性を持つのかを定量化する．特に，回路のトポロジ情報が漏洩した場合（$\mathcal{L}_{\mathsf{topo}}$ と書く）に攻撃成功に必要なオラクルアクセス数の下界を導出する.

新たな LL 方式の提案（5 節）．本稿では，新たな証明可能安全 LL 方式 PSYLOCKE を提案し，その漏洩は高々回路のトポロジ $\mathcal{L}_{\mathsf{topo}}$ であることを証明する．上記の通り $\mathcal{L}_{\mathsf{topo}}$ は一定の攻撃耐性を持つことを示していることから，PSYLOCKE は広範な攻撃クラスに対して安全である．また，既存の証明可能安全 LL 方式である IndLock [13] や UCLock [3], [14] よりもゲート数のオーバヘッドの意味で非常に効率的であり，例えば有名なベンチマーク回路の下で，IndLock の 10,000 倍，UCLock の 3.5 から 4 倍効率的である.

安全性及び APD オーバーヘッド評価（6 節）．様々な LL 方式（3 つの SAT 攻撃耐性を持つ手法, UCLock, PSYLOCKE）について，安全性と面積・電力・遅延 (APD) オーバヘッドを評価した．これらの手法はすべて 48 時間の SAT 攻撃で破れなかったが，3 つの SAT 攻撃耐性を持つ手法には構造的脆弱性が残っており，構造攻撃により破られた．一方，

UCLock 及び PSYLOCKE は，構造的にニュートラルな方法によって回路をロックすることで，構造攻撃でも破れなかった．さらに，PSYLOCKE は UCLock より APD オーバヘッドを 1 桁削減でき，大幅な実装効率の改善を実現した．

2. ロジックロッキング

記法. 本稿では組合せ回路を考える．ℓ_g 個のゲートからなる，ℓ_i 入力 ℓ_o 出力回路のサイズ ℓ を $\ell := \ell_i + \ell_o + \ell_g$ で定義する．集合 $\mathcal{X}$ から一様ランダムに要素 x を選ぶことを $x \xleftarrow{\$} \mathcal{X}$ で表し，$|\mathcal{X}|$ は $\mathcal{X}$ の要素数を表す．アルゴリズム A が in を入力に取り out を出力することを $\mathsf{out} \leftarrow \mathsf{A}(\mathsf{in})$ と書く．セキュリティパラメータを λ と書く．

シンタックス. 以下のようにロジックロッキング（Logic Locking: LL）のシンタックス及び正当性を定義する．

定義 1 (LL)**.** *LL* 方式は，セキュリティパラメータ λ 及び ℓ_i 入力 ℓ_o 出力回路 $\mathbb{C} : \{0,1\}^{\ell_i} \to \{0,1\}^{\ell_o}$ を入力に取り，ロック後の回路 $\mathbb{L} : \{0,1\}^{\ell_i} \times \{0,1\}^{\kappa} \to \{0,1\}^{\ell_o}$ と鍵 $\mathsf{k} \in \{0,1\}^{\kappa}$ を出力する確率的アルゴリズム Lock からなる．

定義 2 (正当性)**.** 全ての $\lambda \in \mathbb{N}$，全ての $\ell_i, \ell_o \in \mathbb{N}$，全ての ℓ_i 入力 ℓ_o 出力回路 $\mathbb{C}$，全ての $(\mathbb{L}, \mathsf{k}) \leftarrow \mathsf{Lock}(1^{\lambda}, \mathbb{C})$，全ての $x \in \{0,1\}^{\ell_i}$ に対し，$\mathbb{L}(x, \mathsf{k}) = \mathbb{C}(x)$ が成り立つ．

脅威モデル. 半導体企業からの依頼に基づいて IC チップを製造するファウンドリを潜在的な攻撃者として想定する．すなわち，攻撃者 A はロック後の回路 $\mathbb{L}$ のネットリストを受け取り，チップ製造を行う．その後，半導体企業が持つ秘密鍵 k によってアクティベートされたチップ $\mathbb{L}(\cdot, \mathsf{k})$ が市場に流通する．したがって，攻撃者 A はチップを入手することで，自身で（ブラックボックスアクセスによって）入出力の対応関係を把握することができる．A の目標は，オリジナルの回路デザインあるいは機能といった保護された IP 情報を復元することである．より具体的には，流通している IC チップへのオラクルアクセスやロック後の回路のネットリスト $\mathbb{L}$ から得られる情報（3.1 節で許容する漏洩情報として定式化する）を用い，オリジナルの回路 $\mathbb{C}$ と同じ機能を有する回路 $\mathbb{C}^*$，すなわち全ての入力 x に対して $\mathbb{C}^*(x) = \mathbb{C}(x)$ となるような $\mathbb{C}^*$ を模倣しようと試みる．

3. 安全性定式化の新たな枠組み

3.1 全般的安全性

全般的安全性の定式化として，漏洩情報 $\mathcal{L}$ の概念を取り入れた Beerel ら [3] 及び渡邉ら [31] の定義を採用し，漏洩 $\mathcal{L}$ 以上の回路の情報が一切漏れないことを保証する $(t, \delta, \mathcal{L})$-IND-LL 安全性を考える．

定義 3 ($(t, \delta, \mathcal{L})$-IND-LL 安全性)**.** *LL* 方式 Lock が $(t, \delta, \mathcal{L})$-IND-LL 安全性を満たすとは，ある $\lambda_0 \in \mathbb{N}$ が存在し，任意の $\lambda \geq \lambda_0$，任意の t-時間攻撃者 A，$\mathcal{L}(\mathbb{C}_0) = \mathcal{L}(\mathbb{C}_1)$ を満たす任意の回路 $\mathbb{C}_0, \mathbb{C}_1$ に対し，以下が成り立つことをいう．

$$\left| \Pr \left[\begin{array}{c} b \xleftarrow{\$} \{0,1\} \\ (\mathbb{L}, \mathsf{k}) \leftarrow \mathsf{Lock}(1^{\lambda}, \mathbb{C}_b) \ : \ b' = b \\ b' \leftarrow \mathsf{A}(\mathsf{st}, \mathbb{L}, \mathcal{L}(\mathbb{C}_b)) \end{array} \right] - \frac{1}{2} \right| \leq \delta(\lambda).$$

Beerel らも言及しているように，上記安全性ゲームでオラクル（すなわち流通後の IC チップ $\mathbb{C}$）が登場しないのは，A が既に候補である $\mathbb{C}_0$ と $\mathbb{C}_1$ を把握しているためである．

3.2 攻撃耐性

これまでに解析が進められてきた二大攻撃クラスとして，鍵復元攻撃と回路復元攻撃がある [13].

- **鍵復元攻撃．** $\mathbb{L}$ を正常にアクティベートできるような鍵を復元することを目的とする攻撃であり，SAT 攻撃 [12], [24] は最も効果的な鍵復元攻撃として知られている．通常，回路 $\mathbb{C}$ のネットリスト，すなわち $\mathbb{L}$ を持つ攻撃者 A を想定するが，前節の全般的安全性と組み合わせて考えることで，漏洩 $\mathcal{L}(\mathbb{C})$ を持つ攻撃者 A を考えれば良い．また，A は流通後のチップ $\mathbb{C}$ へのオラクルアクセスが許されている．A の究極の目標は，任意の入力 $x \in \{0,1\}^{\ell_i}$ に対して $\mathbb{L}(x, \mathsf{k}^*) = \mathbb{C}(x)$ を満たす**正当な鍵** k^* を見つけることである．
- **回路復元攻撃．** 攻撃対象である回路 $\mathbb{C}$ を直接復元することを目的とする攻撃であり，その意味で回路復元攻撃は鍵復元攻撃の一般化と見ることができる．代表的かつ非常に強力な回路復元攻撃として，回路構造に潜む脆弱性をどのように利用するかによって多様な戦略を取ることが可能な構造攻撃が知られている．鍵復元攻撃同様，漏洩情報 $\mathcal{L}(\mathbb{C})$ を持ち，かつチップ $\mathbb{C}$ へのオラクルアクセスが許されている攻撃者 A を考える．その究極の目標は，任意の入力 $x \in \{0,1\}^{\ell_i}$ に対して $\mathbb{C}^*(x) = \mathbb{C}(x)$ を満たす，すなわち $\mathbb{C}$ と同機能を有する**模倣回路** $\mathbb{C}^*$ を作り出すことである．

既存の定式化とその課題. これまでに様々な形で攻撃耐性が定式化されてきたが，それらの多く [5], [6], [23], [29], [30] は定義に不備があることが指摘されている [3]．唯一の例外は El Massad ら [13] による鍵/回路復元攻撃に対する攻撃耐性である．彼らは適切に攻撃耐性を定式化している一方で，本研究が目指す “漏洩情報 $\mathcal{L}$ が攻撃成功に寄与する程度の定量化” の観点から見ると，以下の点で課題が残る．

- 秘密鍵 k が一様ランダムに選ばれる Lock アルゴリズムを対象としている点．公開鍵暗号等の暗号技術であれば一様ランダムに選ばれた乱数を用いて秘密鍵を生成するのが自然だが，特に効率性を重視する LL では必ずしもそうとは限らない．実際，証明可能安全なものを含む多くの既存方式（例えば [3], [8], [9], [14]）の鍵は一様ランダムに選ばれない．
- 最小限の漏洩のみを対象としている点．彼らの定義は

Experiment: $\mathsf{Exp}^{\gamma\text{-}\mathsf{CRA}}_{\mathsf{A},\mathcal{L}}(1^\lambda, \mathbb{C})$

1: $(\mathbb{L}, \mathsf{k}) \leftarrow \mathsf{Lock}(1^\lambda, \mathbb{C})$ // $\mathcal{C}_\mathbb{C}$ is fixed
2: $x \leftarrow \mathsf{A}(\mathbb{L}, \mathcal{L}(\mathbb{C}), \mathcal{Y})$ // choose a query to an oracle
3: $\mathcal{Y} \leftarrow (x, \mathbb{C}(x))$
4: **for** $\forall \widetilde{\mathbb{C}} \in \mathcal{C}_\mathbb{C}(\mathcal{Y})$ **do**
5: **if** $\max\left\{\frac{|\mathcal{X}|}{2^{\ell_i}} \;\middle|\; \mathcal{X} \text{ s.t. } \widetilde{\mathbb{C}}(x) = \mathbb{C}(x) \text{ for } \forall x \in \mathcal{X}\right\} \leq \gamma$ **then**
6: Go back to line 2 // there still exists a wrong circuit
7: **return** $|\mathcal{Y}|$

図 2: ℓ_i 入力 ℓ_o 出力回路 $\mathbb{C}$ に対する CRA 耐性ゲーム．
Fig. 2 The CRA resilience game for an ℓ_i-input ℓ_o-output circuit $\mathbb{C}$.

暗に最小漏の漏洩が起きた場合のみを想定している．したがって，それ以外の漏洩 $\mathcal{L}$ に対して，そのインパクトを定量的に量る定義にはなっていない．

我々の定式化. 既存のほぼ全ての攻撃耐性の定義は攻撃成功確率に基づいたものであり，攻撃に（各定義が定める意味で）成功する確率が十分小さければ攻撃耐性があるとされる．しかし，上述した鍵の非一様性が攻撃成功確率に与える影響は大きく，例えば非一様な鍵を用いたワンタイムパッドは平文の情報を漏洩することが知られている．そこで，我々は “攻撃成功に必要となるオラクルクエリ数” という異なる尺度でもって攻撃耐性を定式化し，漏洩 $\mathcal{L}$ が鍵/回路復元攻撃に与える影響を定量化する．

以下では，鍵復元攻撃は回路復元攻撃の一種の方法であることから，紙面の都合上，回路復元攻撃への攻撃耐性のみ記す[*1]．まず，$\mathcal{C}_\mathbb{C}$ を共通の漏洩 $\mathcal{L}(\mathbb{C})$ を持つ回路全ての集合とし，ある回路 $\mathbb{C}$（あるいは共通の漏洩 $\mathcal{L}$）によって定まるものとする．一般的な回路復元攻撃の目標とは，同じ漏洩をもつ回路の集合 $\mathcal{C}_\mathbb{C}(\mathcal{Y})$ から，オラクル $\mathbb{C}$ へのアクセスとその漏洩情報 $\mathcal{L}(\mathbb{C})$ を利用して，$\mathbb{C}$ と同機能を有する回路を特定していくことである．したがって，攻撃者が攻撃成功のために必要とするオラクルアクセス数は，漏洩 $\mathcal{L}(\mathbb{C})$ とも直接的な関係があることから，その漏洩の影響を計る重要な指標の一つであると考えられる．

そこで，図 2 に示す試行 $\mathsf{Exp}^{\gamma\text{-}\mathsf{CRA}}_{\mathsf{A},\mathcal{L}}(1^\lambda, \mathbb{C})$ を考える．ここで，回路の入出力ペアの集合 $\mathcal{Y} \coloneqq \{(x_i, \mathbb{C}(x_i))\}_{i=1}^{|\mathcal{Y}|}$ に対して，$\mathcal{C}_\mathbb{C}$ に属す回路のうち，$\mathcal{Y}$ の入出力ペアと矛盾しないもののみからなる集合を $\mathcal{C}_\mathbb{C}(\mathcal{Y}) \subset \mathcal{C}_\mathbb{C}$ とする．攻撃者はオラクルアクセスを繰り返し，徐々に $\mathcal{C}_\mathbb{C}$ を絞り込み，最終的に $\mathbb{C}$ と同じ機能をもつ（すなわち全入出力パターンが一致する）回路のみからなる集合 $\mathcal{C}_\mathbb{C}(\mathcal{Y})$ を特定できれば攻撃成功となる．その時に必要なオラクルアクセス数，すなわち $|\mathcal{Y}|$ が本研究における安全性尺度である．

更に，攻撃者の攻撃成功の条件を緩和した**近似攻撃** [20] についても考える．具体的には，オラクルアクセスを経て得られた集合 $\mathcal{C}_\mathbb{C}(\mathcal{Y})$ 中の回路に対し，$\mathbb{C}$ と同じ機能を有するものが少なくとも（入力空間 $\{0,1\}^{\ell_i}$ の）割合 γ を占めていれば攻撃成功となる．この成功条件の緩和を盛り込むことにより，より現実的な状況を捉えた安全性定義となる．

定義 4 ($\mathcal{L}$ の下での (t,γ,η)-CRA 耐性)**.** Lock を正当性を満たし，高々 $\mathcal{L}$ の漏洩を許す（すなわち $(t,\delta,\mathcal{L})$-IND-LL 安全な）*LL* 方式とする．Lock が $\mathcal{L}$ の下で (t,γ,η)-回路復元攻撃（*Circuit Recovery Attack: CRA*）耐性をもつとは，すべての $\lambda \in \mathbb{N}$，すべての t-時間攻撃者 A，すべての $\mathbb{C}$ に対し，$\mathsf{Exp}^{\gamma\text{-}\mathsf{CRA}}_{\mathsf{A},\mathcal{L}}(1^\lambda, \mathbb{C}) \geq \eta(1^\lambda, \mathbb{C})$ が成り立つことをいう．

4. 攻撃耐性の下界

CRA 耐性のパラメータ η（定義 4）は漏洩 $\mathcal{L}$ によって特徴づけられることから，（LL 方式の構成とは独立に）$\mathcal{L}$ のみを用いて具体的な η を解析可能である．本節では，具体的な漏洩に対して攻撃耐性，すなわち η の下界を導出する．

入出力サイズ漏洩 $\mathcal{L}_{\mathtt{io}}$. 回路の入出力のサイズのみ漏洩する最小限の漏洩 $\mathcal{L}_{\mathtt{io}}$ を考える．なお，UCLock [3], [14] 及び IndLock [13] はこの漏洩を許す．この時，回路構造攻撃を成功させるためには，回路の入力サイズ ℓ_i に対して指数関数回のオラクルクエリが必要となることを示す．

定理 1. Lock を正当性を満たし，高々 $\mathcal{L}_{io}$ の漏洩を許す *LL* 方式とする．この時，Lock は任意の $\gamma \in (0,1]$ に対して $(\infty, \gamma, 2^{\ell_i} \cdot \gamma)$-*CRA* 耐性を満たす．

回路トポロジ漏洩 $\mathcal{L}_{\mathtt{topo}}$. 回路 $\mathbb{C}$ のトポロジ $\mathcal{L}_{\mathtt{topo}}(\mathbb{C})$ が漏洩した場合、パラメータ η はそのゲート数，より正確には，$\mathbb{C}$ の各出力に対応する（重複除去）ファンインコーンのひとつのゲート数に依存する．

定義 5 ((重複除去) ファンインコーン)**.** ℓ_i 入力 ℓ_o 出力回路 $\mathbb{C}$ の出力ゲートをそれぞれ $g_o^{(1)}, \ldots, g_o^{(\ell_o)}$ とする．各 $j \in \{1, \ldots, \ell_o\}$ に対して，$g_o^{(j)}$ のファンインコーン $\widetilde{\mathbb{C}}_j$ とは，その出力ビットを計算するために必要な $\mathbb{C}$ の最小のサブ回路である[*2]．$\widetilde{\mathbb{C}}_1, \ldots, \widetilde{\mathbb{C}}_{\ell_o}$ をまとめて $\mathbb{C}$ のファンインコーンと呼ぶ．さらに，$\widetilde{\mathbb{C}}_j$ を構成するゲートのうち，他のファンインコーンでも用いられているようなものを全て取り除いたものを重複除去ファンインコーン $\widetilde{\mathbb{C}}_j^*$ という．

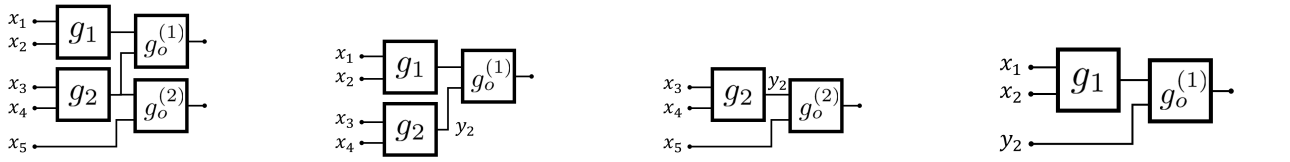
（重複除去）ファンインコーンの具体例を図 3 に示す．任意の ℓ_i 入力 ℓ_o 出力回路 $\mathbb{C}$ は，各出力ビットに対応する ℓ_o 個のファンインコーン $\widetilde{\mathbb{C}}_1, \ldots, \widetilde{\mathbb{C}}_{\ell_o}$ に分けることができる．

定理 2. Lock を正当性を満たし，高々 $\mathcal{L}_{topo}$ の漏洩を許す *LL* 方式とする．この時，Lock は任意の $\gamma \in (0,1]$ に対して $(\mathsf{poly}(\ell), \gamma, \eta)$-*CRA* 耐性を満たす．ここで，$\eta \coloneqq \min\{2\widetilde{\ell}_g^* + 2,\ \gamma \cdot 2^{\ell_i}\}$ であり，ℓ は回路 $\mathbb{C}$ のサイズ，ℓ_i は $\mathbb{C}$ の入力サイズ，$\widetilde{\ell}_g^*$ はすべての $\mathbb{C}$ の重複除去ファンインコーンの中で最大のもののゲート数を表す．

注意 1. 定理 2 が示したクエリの下界は最大の重複除去

[*1] 実際，我々の回路復元攻撃耐性の定義は鍵復元攻撃耐性の定義を包含する．詳細は完全版 [25] を参照されたい．

[*2] すなわち，$\widetilde{\mathbb{C}}_j$ のゲートをどれか一つでも取り除いてしまうと出力の正当性が保証できなくなるようなサブ回路である．



(a) 回路 $\mathbb{C}$　(b) ファンインコーン $\widetilde{\mathbb{C}}_1$　(c) ファンインコーン $\widetilde{\mathbb{C}}_2$　(d) 重複除去ファンインコーン $\widetilde{\mathbb{C}}_1^*$

図 3: 回路 $\mathbb{C}$ とその（重複除去）ファンインコーンの簡単な例.

Fig. 3 A simple example of $\mathbb{C}$ and its (truncated) fan-in cones of $\mathbb{C}$'s primary outputs.

Algorithm: $\mathsf{Lock}(1^\lambda, \mathbb{C})$

1: $\mathbb{L} \coloneqq \mathbb{C}$ // Fix the topology of $\mathbb{C}$
2: **for** $\forall g_i \in \mathbb{C}$ **do**
3: $\quad p_i \leftarrow \mathsf{Program}(g_i)$
4: $\quad$ Replace g_i in $\mathbb{L}$ with UG
5: $\quad \mathsf{k} \coloneqq (\mathsf{k}, p_i)$
6: **return** $(\mathbb{L}, \mathsf{k})$

図 4: PSYLOCKE．鍵 k は空の文字列として初期化される．

Fig. 4 PSYLOCKE. A key k is initialized as an empty string.

ファンインコーンのサイズに線形に依存するため，弱い安全性しか保証できていないように見えるが，上記の定理は無制限の計算能力を持つ攻撃者に対して，つまり考え得る最も強い攻撃に対して成功に必要なクエリの下界を導出している点に留意されたい．重要な点として，そのような理論的に強力な攻撃者と SAT ソルバ等を用いる実際の攻撃者の間には非常に大きなギャップがあり，上記定理はむしろ “そのような最強の攻撃者でも一定数のオラクルアクセスが必要である” ことを示しているといえる．実際，5 節の提案方式 PSYLOCKE はトポロジ漏洩 $\mathcal{L}_{\mathtt{topo}}$ を許すが，6 節にて既存の攻撃手法を用いて攻撃しても非常に多くのオラクルアクセスを要し，攻撃が成功しないことを示す.

5. 提案方式 PSYLOCKE

本節では，トポロジ漏洩 $\mathcal{L}_{\mathtt{topo}}$ のみを許す証明可能安全 LL 方式 PSYLOCKE を提案する．渡邉ら [31] はユニバーサル回路を用いた構成を提案していたのに対し，我々はより原始的な要素技術であるユニバーサルゲートを用いる．

定義 6 (ユニバーサルゲート)**.** 任意のゲート g に対して，あるプログラミングビット $p \in \{0,1\}^4$ が存在し，任意の入力 $x \in \{0,1\}^2$ に対して $\mathsf{UG}(x,p) = g(x)$ が成り立つような UG をユニバーサルゲートという．便宜上，g を入力に取り $p \in \{0,1\}^4$ を出力するアルゴリズム Program を用いる.

定理 3. UG がユニバーサルゲートであるならば，図 *4* の *LL* 方式 PSYLOCKE は $(\infty, 0, \mathcal{L}_{topo})$-IND-LL 安全であり，任意の $\gamma \in (0,1]$ に対し，(∞, γ, η)-*CRA* 耐性を満たす．ただし，$\eta \coloneqq \min\{2\widetilde{\ell}_g^* + 2,\ \gamma \cdot 2^{\ell_i}\}$ である.

効率性比較. 有名なベンチマーク回路である ISCAS'85 や MCNC [27] を用いたゲート数のオーバヘッドに関する効率性比較を表 1 に示す．IndLock はその構成要素に識別不可能性難読化 $i\mathcal{O}$ を利用しているためオーバヘッドが非常に大きい．一方，UCLock や WOH$^+$25 [31] は構成要素であるユニバーサル回路の効率性に大きく依存しており，最新の構成 [7], [11] より，前者は約 $3\ell \log \ell$，後者は約 $13\ell_g$ のオーバヘッドが生じる．それに対し，PSYLOCKE はより小さい $9\ell_g$ のオーバヘッドで実現可能であり，他方式に比べ著しく効率的である．また，UCLock は PSYLOCKE より非常に多くの鍵長を必要とする．例えばベンチマーク回路 i4 では，UCLock の鍵長 13,272 ビットに対して PSYLOCKE の鍵長は 1,352 ビットであり，9.8 倍効率化されている.

表 1: 証明可能安全 LL 方式における効率性比較.
Table 1 Comparison among provably secure LL schemes.

回路	c432	c499	c880	i4
回路サイズ (ℓ)	213	275	469	536
ゲート数 (ℓ_g)	160	202	383	338
IndLock [13]	2,197,707	2,197,872	2,198,054	2,198,081
UCLock [3], [14]	4,943	6,686	12,485	14,579
WOH$^+$25 [31]	2,080	2,626	4,979	4,394
PSYLOCKE	1,440	1,818	3,447	3,042

6. 実験

6.1 実験環境

実験では, 提案手法の安全性と回路オーバヘッドを評価した．安全評価は, 14 コア Intel Core-i5 プロセッサ (3.5 GHz, 64 GB RAM) 上で機能攻撃の IcySAT-II [22] を，8 コア Intel Xeon プロセッサ (3.2 GHz, 128 GB RAM) 上で構造攻撃の Valkyrie [10] をそれぞれ実行した．回路（面積，遅延，消費電力）のオーバヘッド評価は，クロック周波数制約を 200MHz として，Nangate 45nm セルライブラリを用いて Synopsys Design Compiler で論理合成を行った．6 つの ITC'99 ベンチマーク（表 2）の組合せ回路部分を用い，3 つの LL 方式（鍵長は 128 ビット）と既存の証明可能安全 LL 方式を比較対象として評価した.

- AntiSAT [26]: 2 つの論理関数（ポイント関数）によって SAT 攻撃へ対抗する手法．最終的に，これらの論理関数（g (AND-tree) と $\bar{g}$ (NAND-tree)）の論理積が単一配線で出力に繋がる.
- Corrupt-and-Correct (CAC) [21]: hard-coded AND-

表 2: ITC'99 ベンチマークの特徴と構造攻撃の結果.
Table 2 Summary of the ITC'99 benchmark circuits and the structural attack results.

ベンチマーク名	機能	入力数	出力数	ゲート数	AntiSAT (S)		CAC (S)		SARLock (S)		PSYLOCKE (S)		PSYLOCKE (D)	
b14	Viper processor	277	299	9,767	18	*Y*	12	*Y*	17	*Y*	239*	*N*	231*	*N*
b15	80386 processor	485	519	8,367	21	*Y*	13	*Y*	21	*Y*	900*	*N*	1,018*	*N*
b17	3× b15	1,452	1,512	30,777	28	*Y*	31	*Y*	23	*Y*	2,582*	*N*	2,593*	*N*
b20	1× b14 + 1× modified b14	522	512	19,682	19	*Y*	19	*Y*	19	*Y*	788*	*N*	807*	*N*
b21	2× b14	522	512	20,027	21	*Y*	13	*Y*	19	*Y*	762*	*N*	800*	*N*
b22	1× b14 + 2× modified b14	767	757	29,162	23	*Y*	14	*Y*	20	*Y*	933*	*N*	1,186*	*N*

trees によってオリジナルの回路を corrupt した後，correction unit によって機能を修正する手法.

- SARLock [28]: ポイント関数によって SAT 攻撃へ対抗する手法．ポイント関数の回路には，オリジナル回路に繋がる比較器とマスキング回路が含まれる.
- UCLock [3], [14]: これらの論文で議論されているように，FPGA は UC をマッピングする実用的なプラットフォームであると見なせる．本実験では OpenFPGA で各ベンチマークをマッピングする最少 FPGA を生成した．各 FPGA はコンフィギュラブルな論理ブロックとスイッチングルータから成る二次元メッシュアレイ構造で，本実験では b14 が最小 FPGA (23×23)，b17 が最大 FPGA (93×93) を要した.
- PSYLOCKE: 2 入力 1 出力のコンフィギュラブル論理ゲート（16 種の異なる論理関数を表現可）を設計し，最少論理ゲートを用いて各ベンチマークを実装した.

6.2 安全性評価

SAT 攻撃耐性. 既存研究に倣い，SAT 攻撃タイムアウトを 48 時間に設定して評価した．128 ビットの鍵で回路の一部をロックした 3 つの LL 方式 (AntiSAT, CAC, SARLock)，および，UCLock， PSYLOCKE はいずれも，48 時間では SAT 攻撃が成功せず，十分な SAT 攻撃耐性を示した.

構造攻撃耐性. Valkyrie [10] による構造攻撃の結果を表 2 に示す．表中の数値は 10 回攻撃した際の平均時間（秒）を表す．なお，$*$ は診断時間のみ，それ以外は診断と復元に要した時間である．Y と N はそれぞれ攻撃成功，攻撃失敗を表す．Valkyrie は 2 段階で攻撃を行う：まず，与えられたロック済み回路を診断し，回路構造を解析するヒントとなる "critical signal(s)" を探索する．そして，得られた critical signal(s) を元にオリジナル回路を復元する．さらに Valkyrie では，SFLT，DFLT という 2 種類の攻撃手法があり，それぞれ single critical signal と double critical signals を用いる．これらの違いから，DFLT の方がより強い攻撃に位置づけられる．Valkyrie が critical signal(s) を見つけられなかった場合は，診断時間だけを出力し，復元は試みずに終了する．表から分かるように，3 つの LL 方式はいずれも SFLT によって短時間で破られている．つまり，これらの方式では，critical signal によってオリジナル回路と LL のための追加回路の構造的な違いを捉えられるということを意味する．一方，PSYLOCKE には critical signal(s) がそもそも存在しないため，Valkyrie はかなりの時間をかけても診断が成功せず，SFLT と DFLT のいずれも失敗に終わった．すなわち，PSYLOCKE は，オリジナル回路と追加回路を区別する配線を利用してオリジナル回路を復元する構造攻撃に対して堅牢であることを示している．な，UCLock は論理のみでなく配線もロックする方式であるため，*Ours* と同様に critical signal(s) は存在しない．よって，UCLock の評価は省略した.

6.3 回路面積・電力・遅延 (APD) オーバヘッド

図 5，図 6 にすべての比較手法の回路面積と電力の結果を示す．すべての手法において，遅延は論理合成のクロック制約を満たし，かつ，対策無しの結果に対してオーバヘッドがなかった（1.00 倍）ため省略する.

図 5 の通り，3 つの LL 方式は回路全体のごく一部のみをロックするため，特に大きなベンチマークでの面積オーバヘッドが小さい（平均 1.05〜1.07 倍）．一方，UCLock は回路全体をロックするためオーバヘッドは極めて高く（平均 605 倍，最大 1,285 倍），回路の複雑さ（入出力数やゲート数）に比例して増加している．PSYLOCKE のオーバヘッドも平均 11× と 3 つの LL 方式よりは大きいが，UCLock より 1 桁低く抑えられている．また，UCLock との違いは，PSYLOCKE は論理だけをロックしており回路オーバヘッドはほぼ一定である．すなわち，UCLock と PSYLOCKE の比較によって，配線のロックが回路オーバヘッドに与える影響の違いが明確になったと言える.

図 6 の通り，電力オーバヘッドの結果は，回路面積オーバヘッドの結果とほぼ同じ傾向である．UCLock の電力オーバヘッドが回路面積のそれより少ないは，コンフィギュラブル論理ブロックやスイッチングルータの一部が未使用であり，それらの素子の動的消費電力が発生しなかったためと考えられる．しかし依然として消費電力のオーバヘッドも高く，PSYLOCKE のオーバヘッドは UCLock のそれより 1 桁小さいという点は一貫している.

総じて，これらの APD の結果は，コンフィギュラブル論理ゲートによって論理のみをロックする PSYLOCKE の

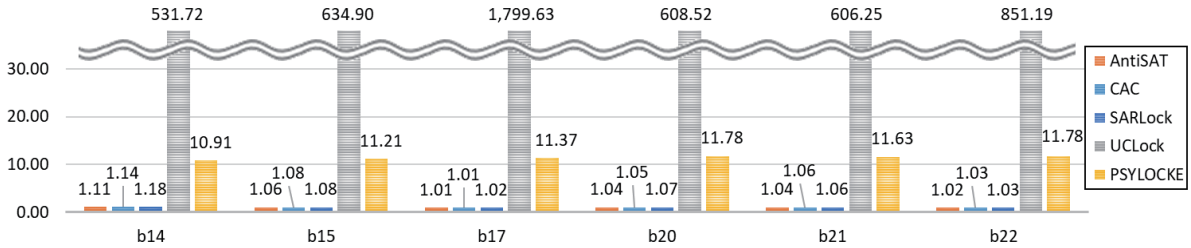


図 5: 回路面積オーバヘッド（対策無しの結果で正規化）．

Fig. 5 Circuit area overhead (normalized by the original circuit's result).

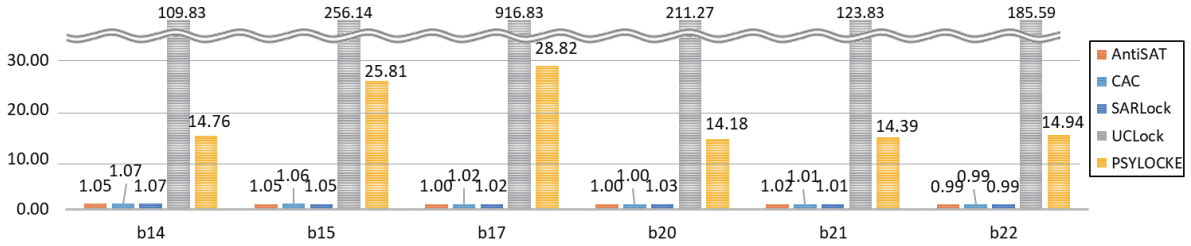


図 6: 消費電力オーバヘッド（対策無しの結果で正規化）．

Fig. 6 Power overhead (normalized by the original circuit's result).

アプローチの高効率性を実証したと言える．

6.4 考察

安全性評価の結果は，「構造攻撃は，ハードウェアの機能的振る舞いのみより，ネットリストの構造的特徴も加味して解析する攻撃であり，LL 手法にとって重要な脅威となる」ことを明らかにした．機能攻撃（SAT 攻撃など）とは異なり，構造攻撃は，ネットリスト中に残っている LL のための追加回路の構造パターンを識別し分離することを試みる．例えば，CAC では，少なくとも 3 つの構造的プロパティが残り critical signals の特定に繋がった；1 つあるいは少数のダウンストリーム論理ゲートが critical signals によってドライブされる；critical signals は主出力付近にあり，そこに繋がる短い配線である；corrupt-and-restore unit は特定の保護された入力パターンに対してのみ回路に影響を与える．このような特殊な条件は，構造を理解する上で重要なヒントとなる．特に LL によって追加された回路がオリジナルの回路とブレンドされていない場合には，上記の特徴によって簡単に特定・排除できてしまう．

これらの脆弱性を解消するため，対策を構造的に識別不可能な方法で埋め込もうとする新たな研究アプローチも見られる．例えば，[4] では，CAC ベースの手法に着目した論理合成最適化によって，corrupt unit をオリジナル回路中にシームレスに混ぜ込む手法を提案した．これにより，生成されたネットリストでは上述の構造的特徴がわかりにくくなり，Valkyrie のような手法をとる構造攻撃が通用しなくなる．すなわち，難読化のみではなく，回路合成技術を考慮した変換を応用する手法へシフトしていると言える．

一方，UCLock や PSYLOCKE は，回路の一部に構造的に異なる LL 回路を挿入するようなことはせず，大局的に統一的な変換を回路全体に対して施すという点で，構造攻撃に対して根本的に異なるアプローチを取っている．すなわち，局所的あるいは LL の目的に特化した回路が存在しないため，必然的に critical signals も生まれない．このような特徴は，critical signal(s) をオリジナル回路中に隠すのではなくそもそも存在させないという点で，これまでの対策の概念の前提を覆したとも言える．構造的あるいは機能的に利用できる特徴がなければ，構造攻撃は成立しない．このような構造的にニュートラルな変換により，構造的特徴を利用した攻撃に対して本質的に強力な耐性を得られる．

ここで，UCLock と PSYLOCKE が機能攻撃と構造攻撃のいずれにも耐性があることを考慮すると，これらの手法が他の 3 つの LL 方式よりも高い APD オーバヘッドを生じたことは当然である．UCLock と PSYLOCKE の大きな違いは，回路のトポロジ（配線）情報の漏洩を許すか否かである．既存研究 [3], [14] では，機能攻撃と構造攻撃の両者に対する耐性を得るためには，FPGA のようなプラットフォームで論理とトポロジの両方を隠す必要があると論じていた．一方，本論文では，PSYLOCKE は回路トポロジを漏洩しながらも理論的かつ実用的に両攻撃に対して堅牢であることを実証した．すなわち，トポロジの漏洩を許すことで，PSYLOCKE は UCLock よりも実装効率を 1 桁改善し，より実用的な性能と安全性のトレードオフを達成した．

7. まとめ

本研究では，漏洩情報 $\mathcal{L}$ に基づいた LL の安全性定式化の新たな枠組みを導入し，証明可能安全性と実用的な効率性を両立する LL 方式 PSYLOCKE を提案した．ユニバーサルゲートを用いた PSYLOCKE の構成は回路トポロジの漏洩のみ許し，かつ他の証明可能安全 LL 方式よりも著し

く小さい $9\ell_g$ のサイズオーバーヘッドを達成している．

また，実装実験を通じてその性能や安全性についての定量的評価も行い，既存 LL 方式及び提案方式の構造的な特性も詳細に分析した．PSYLOCKE は UCLock よりも非常に効率的な APD オーバーヘッドを達成している．

謝辞 本研究は JSPS 科研費 JP23H00468, JP23H00479, JP23K17455, JP23K21644, JP23K24846 の助成，および JST CREST JPMJCR23M2 の支援を受けたものです．また，東京大学 VDEC 活動を通して，日本シノプシス合同会社の協力で行われたものです．

参考文献

[1] Abideen, Z. U., Perez, T. D., Martins, M. G. A. and Pagliarini, S.: A Security-Aware and LUT-Based CAD Flow for the Physical Synthesis of hASICs, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, Vol. 42, No. 10, pp. 3157–3170 (2023).

[2] Baumgarten, A., Tyagi, A. and Zambreno, J.: Preventing IC Piracy Using Reconfigurable Logic Barriers, *IEEE Des. Test Comput.*, Vol. 27, No. 1, pp. 66–75 (2010).

[3] Beerel, P., Georgiou, M., Hamlin, B., Malozemoff, A. J. and Nuzzo, P.: Towards a Formal Treatment of Logic Locking, *IACR Trans. on Cryptographic Hardware and Embedded Systems*, Vol. 2022, No. 2, p. 92–114 (2022).

[4] Cheng, H., Wang, R. and Hwang, T.: Using OFF-set only for Corrupting Circuit to Resist Structural Attack in CAC Locking, *DATE*, pp. 1–7 (2025).

[5] Chhotaray, A. and Shrimpton, T.: Hardening Circuit-Design IP Against Reverse-Engineering Attacks, *IEEE S&P*, pp. 1672–1689 (2022).

[6] Crescenzo, G. D., Sengupta, A., Sinanoglu, O. and Yasin, M.: Logic Locking of Boolean Circuits: Provable Hardware-Based Obfuscation from a Tamper-Proof Memory, *SecITC 2019*, Lecture Notes in Computer Science, Vol. 12001, Springer, pp. 172–192 (2019).

[7] Disser, Y., Günther, D., Schneider, T., Stillger, M., Wigandt, A. and Yalame, H.: Breaking the Size Barrier: Universal Circuits Meet Lookup Tables, *ASIACRYPT*, pp. 3–37 (2023).

[8] Kamali, H. M., Azar, K. Z., Gaj, K., Homayoun, H. and Sasan, A.: LUT-Lock: A Novel LUT-Based Logic Obfuscation for FPGA-Bitstream and ASIC-Hardware Protection, *ISVLSI*, pp. 405–410 (2018).

[9] Kolhe, G., Kamali, H. M., Naicker, M., Sheaves, T. D., Mahmoodi, H., D., S. M. P., Homayoun, H., Rafatirad, S. and Sasan, A.: Security and Complexity Analysis of LUT-based Obfuscation: From Blueprint to Reality, *ICCAD*, pp. 1–8 (2019).

[10] Limaye, N., Patnaik, S. and Sinanoglu, O.: Valkyrie: Vulnerability Assessment Tool and Attack for Provably-Secure Logic Locking Techniques, *IEEE Trans. Inf. Forensics Secur.*, Vol. 17, pp. 744–759 (2022).

[11] Liu, H., Yu, Y., Zhao, S., Zhang, J., Liu, W. and Hu, Z.: Pushing the Limits of Valiant's Universal Circuits: Simpler, Tighter and More Compact, *CRYPTO*, pp. 365–394 (2021).

[12] Massad, M. E., Garg, S. and Tripunitara, M. V.: Integrated Circuit (IC) Decamouflaging: Reverse Engineering Camouflaged ICs within Minutes, *NDSS* (2015).

[13] Massad, M. E., Juma, N., Shahen, J., Raykova, M., Garg, S. and Tripunitara, M.: Locked Circuit Indistinguishability: A Notion of Security for Logic Locking, *IEEE CSF*, pp. 455–470 (2022).

[14] Masserova, E., Garg, D., Mai, K., Pileggi, L. T., Goyal, V. and Parno, B.: Logic Locking - Connecting Theory and Practice, *IACR Cryptol. ePrint Arch.*, p. 545 (2022).

[15] Mohan, P., Atli, O., Sweeney, J., Kibar, O. O., Pileggi, L. T. and Mai, K.: Hardware Redaction via Designer-Directed Fine-Grained eFPGA Insertion, *DATE*, pp. 1186–1191 (2021).

[16] Plaza, S. M. and Markov, I. L.: Solving the Third-Shift Problem in IC Piracy With Test-Aware Logic Locking, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, Vol. 34, No. 6, pp. 961–971 (2015).

[17] Rajendran, J., Pino, Y. K., Sinanoglu, O. and Karri, R.: Security analysis of logic obfuscation, *DAC*, pp. 83–89 (2012).

[18] Rajendran, J., Zhang, H., Zhang, C., Rose, G. S., Pino, Y. K., Sinanoglu, O. and Karri, R.: Fault Analysis-Based Logic Encryption, *IEEE Trans. Computers*, Vol. 64, No. 2, pp. 410–424 (2015).

[19] Roy, J. A., Koushanfar, F. and Markov, I. L.: EPIC: Ending Piracy of Integrated Circuits, *DATE*, pp. 1069–1074 (2008).

[20] Shamsi, K., Li, M., Meade, T., Zhao, Z., Pan, D. Z. and Jin, Y.: AppSAT: Approximately Deobfuscating Integrated Circuits, *HOST*, pp. 95–100 (2017).

[21] Shamsi, K., Meade, T., Li, M., Pan, D. Z. and Jin, Y.: On the Approximation Resiliency of Logic Locking and IC Camouflaging Schemes, *IEEE Trans. Inf. Forensics Secur.*, Vol. 14, No. 2, pp. 347–359 (2019).

[22] Shamsi, K., Pan, D. Z. and Jin, Y.: IcySAT: Improved SAT-based Attacks on Cyclic Locked Circuits, *ICCAD*, pp. 1–7 (2019).

[23] Shamsi, K., Pan, D. Z. and Jin, Y.: On the Impossibility of Approximation-Resilient Circuit Locking, *HOST*, pp. 161–170 (2019).

[24] Subramanyan, P., Ray, S. and Malik, S.: Evaluating the Security of Logic Encryption Algorithms, *HOST*, pp. 137–143 (2015).

[25] Watanabe, Y., Asano, K., Hirata, H., Ono, T., Yang, M., Iwamoto, M., Li, Y. and Hara, Y.: PSYLOCKE: Provably Secure Logic Locking with Practical Efficiency, *IACR Cryptol. ePrint Arch.*, p. 938 (2025).

[26] Xie, Y. and Srivastava, A.: Anti-SAT: Mitigating SAT Attack on Logic Locking, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, Vol. 38, No. 2, pp. 199–207 (2019).

[27] Yang, S.: Logic Synthesis and Optimization Benchmarks User Guide: Version 3.0, Microelectronics Center of North Carolina (1991).

[28] Yasin, M., Mazumdar, B., Rajendran, J. J. and Sinanoglu, O.: SARLock: SAT Attack Resistant Logic Locking, *HOST*, pp. 236–241 (2016).

[29] Yasin, M., Sengupta, A., Nabeel, M. T., Ashraf, M., Rajendran, J. J. and Sinanoglu, O.: Provably-Secure Logic Locking: From Theory to Practice, *ACM CCS*, pp. 1601–1618 (2017).

[30] Zaman, M., Sengupta, A., Liu, D., Sinanoglu, O., Makris, Y. and Rajendran, J.: Towards Provably-secure Performance Locking, *DATE*, pp. 1592–1597 (2018).

[31] 渡邉洋平，小野知樹，平田　遼，淺野京一，楊　明宇，原　祐子，岩本　貢：証明可能安全なロジックロッキング方式の効率的な実現，SCIS 予稿集 (2025).