# Towards Provably Secure LLM Fingerprinting: A Formal Security Framework

Research Assistant AI

October 2, 2025

**Abstract**

The practice of fingerprinting large language models (LLMs) to protect intellectual property is becoming increasingly critical. However, current evaluation of fingerprinting schemes relies on an empirical, attack-driven methodology, creating an arms race between attackers and defenders. As demonstrated by Nasery et al. (2025), many existing schemes are vulnerable to adaptive adversarial attacks. To move beyond this cycle, we propose the first formal security framework for LLM fingerprinting. Our framework introduces rigorous, game-based definitions for security, capturing the adversary's dual goals of evading detection while preserving model utility. We define the core components of a fingerprinting system the fingerprinting algorithm, the verification algorithm, and the adversary and formalize security in terms of a Utility-Preserving Evasion (UPE) game. Using this framework, we analyze the systemic vulnerabilities of current methods, such as those exploiting overconfidence or unnatural queries, and show why they fail our formal security definitions. Finally, we leverage our framework to outline a principled path toward designing next-generation, provably secure fingerprinting schemes, emphasizing properties like non-localizability and semantic verification.

## 1 Introduction

The development of state-of-the-art Large Language Models (LLMs) requires immense computational resources and curated data, representing a significant investment. Consequently, protecting this intellectual property from unauthorized duplication and use is a paramount concern for model developers. Model fingerprinting has emerged as a leading paradigm for ownership verification, allowing a developer to embed a secret, detectable signal within a model's weights.

However, the security of these fingerprinting schemes is currently evaluated in an ad-hoc, empirical manner. A new scheme is typically proposed and demonstrated to be robust against a known set of transformations, such as fine-tuning or quantization. Subsequently, new research often reveals adaptive attacks that completely bypass these defenses. A prominent example is the work of Nasery et al. [1], which identified four fundamental vulnerabilities and presented attacks that successfully broke ten different fingerprinting schemes. This reactive cycle of attack and defense lacks predictive power and fails to establish clear design principles for building secure systems.

To mature the field beyond this empirical arms race, we argue for a formal, cryptographic-style approach. In this paper, we introduce the first formal security framework for LLM fingerprinting. Our contributions are as follows:

1. We define the formal syntax of a fingerprinting scheme, consisting of `Fingerprint` and `Verify` algorithms.
2. We formalize the threat model, considering a white-box adversary with complete access to the model's architecture and parameters, whose goal is to remove the fingerprint.
3. We introduce a quantitative and formal measure of model utility, which is essential for defining a realistic adversary who must preserve the model's performance on benign tasks.
4. We propose a formal security definition centered on a **Utility-Preserving Evasion (UPE)** game. An adversary wins this game if they can evade fingerprint verification while incurring only a negligible loss in model utility.

5. We analyze the vulnerabilities identified by Nasery et al. through the lens of our framework, formally demonstrating why existing schemes are insecure.

This framework provides a rigorous foundation for analyzing, comparing, and ultimately designing the next generation of provably secure fingerprinting schemes.

# 2 Preliminaries: Defining a Fingerprinting Scheme

We model an LLM, $M$, as a function parameterized by a vector of weights $\theta$. For a given input prompt (a sequence of tokens) $p$, the model outputs a probability distribution over the next token from a vocabulary $\mathcal{V}$:

$$M(p;\theta) = P(\cdot|p;\theta)$$

**Definition 1** (Fingerprinting Scheme)**.** *A fingerprinting scheme* $\Pi$ *is a pair of probabilistic polynomial-time algorithms* ($\texttt{Fingerprint}$, $\texttt{Verify}$)*:*

- $\texttt{Fingerprint}(M, S) \to M'$*: The fingerprinting algorithm takes as input the original model* $M$ *with parameters* $\theta$ *and a set of secret fingerprint data* $S = \{(q_i, r_i)\}_{i=1}^{n}$*, where* $q_i$ *are fingerprint queries and* $r_i$ *are the target responses. It outputs a new, fingerprinted model* $M'$ *with parameters* $\theta'$*.*
- $\texttt{Verify}(M'', S) \to \{Accept, Reject\}$*: The verification algorithm takes as input a suspect model* $M''$ *and the secret data* $S$*. It interacts with* $M''$ *(e.g., by providing the queries* $q_i$*) and outputs a decision.*

A functional fingerprinting scheme must satisfy a basic correctness property, ensuring that an honestly fingerprinted model is recognized by the verifier.

**Definition 2** (Correctness)**.** *A scheme* $\Pi$ *is correct if for any model* $M$ *and any secret* $S$ *generated according to the scheme's specification, the following holds for a security parameter* $\lambda$*:*

$$Pr[\texttt{Verify}(\texttt{Fingerprint}(M, S), S) = Accept] \geq 1 - negl(\lambda)$$

*where* $negl(\cdot)$ *is a negligible function.*

# 3 The Formal Security Model

The central contribution of this work is a formal definition of security for LLM fingerprinting. This requires formalizing both the adversary's goal (evasion) and constraints (utility preservation).

## 3.1 Model Utility

An adversary who steals a model wishes to use it. Therefore, any modifications made to erase a fingerprint must not significantly degrade the model's performance on general tasks. We formalize this constraint with a utility function.

**Definition 3** (Model Utility)**.** *Let* $\mathcal{D}$ *be a distribution over benign, real-world task instances* $(x, y)$*, where* $x$ *is a prompt and* $y$ *is a desired (ground-truth) response. Let* $Score(M(x), y)$ *be a function measuring the quality of model* $M$*'s response to prompt* $x$ *against* $y$*. The utility of a model* $M$*, denoted* $U(M)$*, is its expected performance over this distribution:*

$$U(M) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[Score(M(x), y)]$$

*The scoring function can be accuracy for classification tasks, ROUGE for summarization, or other relevant metrics.*

## 3.2 The Utility-Preserving Evasion (UPE) Game

We define the security of a scheme $\Pi$ through a game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

**Definition 4** (The UPE Game)**.** *The Utility-Preserving Evasion game, denoted* $Game_{UPE}(\mathcal{A}, \Pi, \lambda)$*, proceeds as follows:*

1. ***Setup:*** *The challenger $\mathcal{C}$ selects a base model $M$ and generates a secret fingerprint set $S$ according to a security parameter $\lambda$.*

2. ***Fingerprinting:*** *The challenger computes the fingerprinted model $M' \leftarrow \texttt{Fingerprint}(M, S)$ and sends $M'$ to the adversary $\mathcal{A}$.*

3. ***Attack:*** *The adversary $\mathcal{A}$, given white-box access to $M'$, produces a modified model $M''$.*

4. ***Winning Condition:*** *The adversary $\mathcal{A}$ wins the game (output is 1) if both of the following conditions are met:*

   - ***Evasion:*** $\texttt{Verify}(M'', S) = \textit{Reject}$.
   - ***Utility Preservation:*** $U(M'') \geq U(M') - \varepsilon$, *for a pre-defined utility-loss tolerance* $\varepsilon \geq 0$.

   *Otherwise, the adversary loses (output is 0).*

### 3.3 Security Definition

Based on the UPE game, we can now define what it means for a fingerprinting scheme to be secure.

**Definition 5** ($(\varepsilon, \delta)$-Security)**.** *A fingerprinting scheme $\Pi$ is $(\varepsilon, \delta)$-secure if for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the UPE game is less than $\delta$:*

$$Pr[Game_{UPE}(\mathcal{A}, \Pi, \lambda) = 1] \leq \delta$$

This definition formalizes the security guarantee. It asserts that no efficient adversary can remove the fingerprint (evade verification) without either suffering a significant utility loss (greater than $\varepsilon$) or having only a small probability of success (less than $\delta$).

## 4 Analysis of Existing Vulnerabilities

We now use our formal framework to analyze the vulnerabilities identified by Nasery et al. [1], demonstrating that the corresponding schemes are not $(\varepsilon, \delta)$-secure for any reasonably small $\varepsilon$ and $\delta$.

### 4.1 Verbatim Verification Schemes

Many early schemes rely on the fingerprinted model producing an exact, memorized string.

- **Formal Flaw:** The `Verify` algorithm checks for an exact match: $\text{argmax}_y P(y|q_i; \theta'') = r_i$. An adversary can deploy a simple `SuppressTop-k` attack, which slightly perturbs the output distribution at the first token generation step by disallowing the most probable token. This minimal change causes verification to fail, so the **Evasion** condition is met. Because the modification is tiny and can be targeted, its effect on the model's behavior over the broad distribution $\mathcal{D}$ of benign prompts is negligible. Thus, the utility loss is minimal, satisfying the **Utility Preservation** condition for a very small $\varepsilon$. The adversary's winning probability $\delta$ approaches 1.

### 4.2 Overconfidence as a Side-Channel

Invasive fingerprinting often leads to the model being overconfident in its responses to fingerprint queries.

- **Formal Flaw:** Overconfidence means that for a fingerprint query $q_i$, the output probability is sharply peaked: $\max_t P(t|q_i; \theta') \gg \max_t P(t|x; \theta')$ for a benign prompt $x \sim \mathcal{D}$. This is a statistical leak of information. An adversary can exploit this by applying an attack (like `SuppressTop-k`) *only when* the model's confidence exceeds a high threshold. This conditional attack almost never triggers on benign inputs from $\mathcal{D}$, leading to a near-zero utility loss ($\varepsilon \approx 0$). However, it reliably triggers on fingerprint queries, ensuring evasion. The scheme is therefore not secure.

### 4.3 Unnatural Queries

Intrinsic fingerprinting methods sometimes generate queries that are statistically distinguishable from natural language.

- **Formal Flaw:** The distribution of fingerprint queries, $S_q = \{q_i\}$, is statistically distinguishable from the benign prompt distribution $\mathcal{D}$. An adversary can compute the perplexity of an incoming prompt $p$ using a generic language model. The adversary's model $M''$ can then implement a simple rule: if Perplexity($p$) is above a threshold, refuse to answer; otherwise, compute the response using $M'$. Since all $q_i \in S_q$ have high perplexity, verification will always fail. Since prompts $x \sim \mathcal{D}$ have low perplexity, the filter does not affect utility, meaning $\varepsilon = 0$. The adversary wins with probability $\delta = 1$.

## 5 Principles for Provably Secure Fingerprinting

Our formal framework not only allows us to diagnose failures but also guides the design of more robust schemes. We propose the following principles for achieving $(\varepsilon, \delta)$-security.

**Principle 1** (Semantic Verification)**.** *The* `Verify` *algorithm must be robust to semantically meaningless perturbations. Instead of relying on exact string matches, verification should operate in a semantic space. For example, using a sentence embedding model $E(\cdot)$:*

$$\texttt{Verify} \text{ accepts if } \text{distance}(E(M''(q_i)), E(r_i)) \leq \tau$$

*To evade this, an adversary must produce an output that is semantically different, which is more likely to incur a significant utility penalty $\varepsilon$.*

**Principle 2** (Fingerprint Non-Localization)**.** *The changes induced by the* `Fingerprint` *algorithm should be distributed diffusely across the model's parameters $\theta$. If the fingerprint is localized to a small, identifiable set of neurons or layers, an adversary can use model editing techniques to erase it with minimal collateral damage. A diffuse fingerprint, represented by small changes to millions of parameters, is much harder to isolate and remove without causing a catastrophic drop in utility.*

**Principle 3** (Input Indistinguishability)**.** *The distribution of fingerprint queries $\{q_i\}$ must be computationally indistinguishable from the distribution of benign prompts $\mathcal{D}$. This directly counters filtering attacks based on perplexity or other statistical anomalies, forcing an adversary to process fingerprint and benign queries identically. This increases the difficulty of mounting a targeted attack that preserves utility.*

## 6 Conclusion

The current approach to validating LLM fingerprinting schemes is stuck in a reactive cycle of empirical attacks and patches. We have introduced the first formal security framework to break this cycle, providing a path towards provable security. Our central contribution, the Utility-Preserving Evasion (UPE) game and the resulting $(\varepsilon, \delta)$-security definition, provides a rigorous method for analyzing and comparing the security of fingerprinting schemes. By using this framework to formally explain the vulnerabilities in existing systems, we have demonstrated its diagnostic power. Furthermore, the principles of semantic verification, non-localization, and input indistinguishability, derived from our formal model, provide a constructive roadmap for future research. The ultimate goal is to design novel fingerprinting schemes and to formally prove that they are $(\varepsilon, \delta)$-secure within this framework.

## References

[1] Nasery, A., Contente, E., Kaz, A., Viswanath, P., & Oh, S. (2025). *Are Robust LLM Fingerprints Adversarially Robust?*. arXiv preprint arXiv:2509.26598. Available at: http://arxiv.org/pdf/2509.26598v1