

Evidence on Demand: Selectively Disclosable Watermarks for Generative AI

Yuuki Fujita^{1,2}[0009–0005–5342–4530] and Keisuke Hara^{3,2}[0000–0003–3598–0988]

¹ Kyoto University, Kyoto-shi Kyoto 606-8501, Japan
fujita.yuki.67f@st.kyoto-u.ac.jp

² National Institute of Advanced Industrial Science and Technology, Koto-ku Tokyo
135-0064, Japan

³ The University of Osaka, Suita-shi Osaka 565-0871, Japan
hara-keisuke@ist.osaka-u.ac.jp

Abstract. Cryptographic watermarking for generative AI equips a generative model with a hidden mark that identifies its outputs as machine-generated, without affecting output quality. This study proposes a new cryptographic watermarking method with *selective disclosability*. Under this scheme, detection is restricted by default to the model owner, who can selectively issue a short token for any *single* output. This token enables any third party to verify that the specific output is watermarked, while guaranteeing that all other outputs remain undetectable, even to the token holder. This scheme is particularly useful in scenarios where a provider must certify a single controversial output without compromising the privacy of other outputs. In addition to two existing security properties, namely robustness and unforgeability, we formalize two security properties, undetectability of undisclosed data and disclosure soundness, to achieve our goal. We then propose our construction based on prior work (Lin, Shahabi, and Song, CRYPTO 2026) and prove it satisfies the four security notions. Our construction is broadly applicable to autoregressive generative AI models, including LLMs.

Keywords: Cryptography · Watermark · Digital Signature · Steganography · AI Security

1 Introduction

1.1 Background

With the advancement of generative AI, it has become necessary to distinguish between AI-generated and human-created content. Deepfake media are used for pranks, fraud, and other malicious purposes without disclosing that they were generated by AI. Thus, identifying such content as AI-generated can prevent confusion and financial damage. Furthermore, because many countries adopt legal interpretations that do not recognize copyright for content generated by AI without human modification, whether content was generated by AI has become a central focus in discussions regarding the recognition of copyright. Additionally,

in academic reports, creative contests, and other settings where the use of AI is prohibited, it is necessary to check whether there are any rule violations. In terms of regulation, Article 50 of the EU Artificial Intelligence Act requires that AI service providers make AI-generated content detectable as artificially generated [9].

Based on such demands, to determine whether content was generated by AI, some services have adopted the approach of embedding watermarks into AI-generated content. Google developed a proprietary watermarking technology called SynthID, initially announcing its adoption in the image generation service Imagen [12], and later in Gemini [11]. Anthropic also announced the adoption of watermarks in text generated by Claude [1].

1.2 Properties and Limitations of Existing Schemes

While existing mainstream watermarks currently rely on heuristics which can sometimes lead to incorrect determinations and quality degradation, cryptographic watermarks that guarantee the difficulty of removal and prevent detection and misidentification have also been presented in prior work. Christ et al. [6] demonstrated a symmetric key watermarking scheme. Fairoze et al. [10] proposed a watermark scheme verifiable in the public key setting. Furthermore, Lin et al. [16] showed an asymmetric key watermark construction with enhanced security guarantees. These schemes mainly consider three security notions: robustness, soundness (or unforgeability), and undetectability. Robustness is the property that a watermark can still be detected even if the generated content undergoes a certain amount of modification. Soundness means that no one can embed a watermark without the corresponding secret key. Unforgeability ensures this kind of security under the condition that watermarked outputs are available to adversaries. Undetectability means that watermarks cannot be detected without the corresponding secret key. In other words, it ensures that the quality of generated content does not degrade even if watermarks are embedded.

However, with symmetric key watermarks, only the model owner can verify whether data is machine-generated. To prove this fact to another party, the owner must share the secret key with the party. This solution allows the party to create watermarked content from that point on, making watermarks meaningless. Thus, symmetric key watermarks have limited application.

Public key watermarks have overcome the limitation, but they have other weaknesses. First, even users who want to distribute deepfakes for malicious purposes can detect a watermark. Considering the difficulty of removal is currently limited, they can repeatedly modify the content until the watermark is no longer detected. This process can be executed automatically. After confirming watermarks are no longer detected, they can distribute the deepfake media to the public. From another perspective, users without malicious intent might not want content to be recognizable to everyone as AI-generated. Even in the situation where the use of AI is permitted, content is sometimes undervalued when it is recognized as AI-generated. Thus, it is undesirable for watermarks to be

detected when no one needs to tell the difference between human-created and AI-generated content.

One might think that the AI service provider should adopt public key watermarks and share its public key with restricted parties, but this solution has shortcomings. First, upon receiving the public key, the parties can detect watermarks in all content from that point on. This solution grants excessively strong authority to the parties and enables them to take advantage of the weakness of public key watermarks. Second, even if we assume the designated parties are honest, shared public keys might be leaked by attacks from malicious users. This kind of risk increases each time the provider sends its public key to someone. Thus, the AI provider should be able to disclose embedding of a watermark for a *single* specific output.

1.3 Our Contribution

To address this challenge, we formalize and propose a selectively disclosable watermark scheme in this study. This scheme normally possesses properties like those of symmetric key schemes, but a verification token can be issued for an individual output, allowing an embedded watermark to be verified by third parties. To capture our goal, we discuss and formalize two new security notions, undetectability of undisclosed data and disclosure soundness. First, it is essential to ensure the undetectability of watermarks in settings where an adversary can obtain verification tokens for other outputs. We formalize this property as undetectability of undisclosed data. Second, it is necessary to consider attacks that cause content lacking a watermark to be recognized as watermarked, since the proposed scheme involves the step of proving authenticity. These attacks correspond to a scenario where a malicious AI provider claims that human-created content has been generated by its AI. We formalize security against these types of attacks as disclosure soundness.

Our proposed scheme satisfies the two introduced notions and the two existing security properties, i.e., robustness and unforgeability, even with its new feature that discloses embedding of a watermark. We would like to ensure robustness and unforgeability against anyone given verification tokens, so we additionally provide an adversary with access to the disclosing oracle that issues verification tokens in the security games. This paper presents the formal proof of these four security notions from the standard security features of the building blocks, namely, robust signatures [16], steganography [5], commitments, and non-interactive zero-knowledge arguments of knowledge, without introducing any additional assumptions not present in prior work.

1.4 Application Scenarios

To help readers understand our proposed scheme, we present an application scenario where the security properties of existing schemes are insufficient. Consider the case where deepfake footage is submitted by a malicious plaintiff as evidence in court. First, the malicious plaintiff uses generative AI to produce footage,

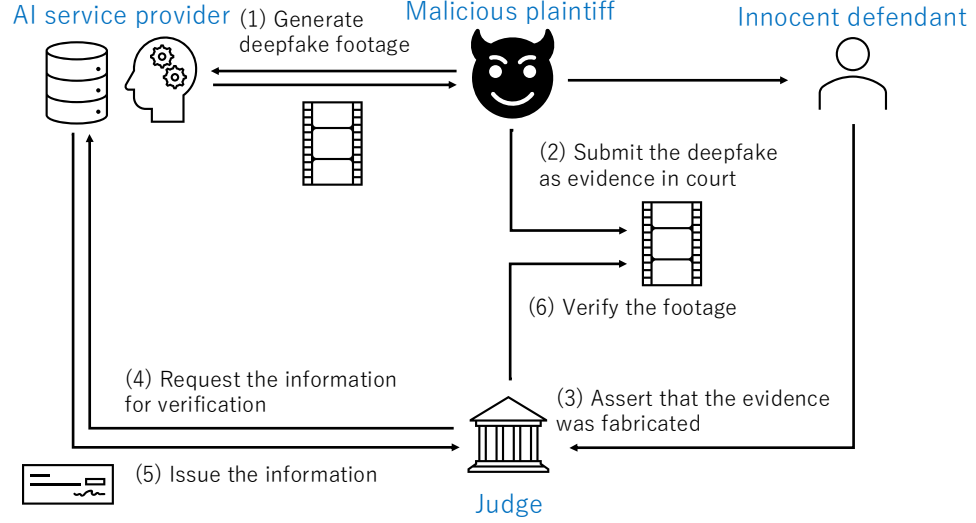


Fig. 1. Flow of Our Proposed Scheme in the Discussed Scenario

submits it to the court as evidence, and files a lawsuit. Naturally, the defendant asserts that the evidence was fabricated. The court must judge whether this evidence is genuine while requesting cooperation from the AI service provider.

With our proposed scheme, when the court needs to determine whether the evidence is a deepfake, the AI service provider can issue a token for verification upon request from the court, as described in Fig. 1. This prevents the malicious plaintiff from checking watermarks before the deepfake is submitted, and furthermore enables a third party to be convinced that watermarks are embedded. Furthermore, if the token is leaked by an attack, it will not help detect watermarks embedded in other outputs. This is ensured by our introduced new notion, undetectability of undisclosed data.

In this scenario, we need to consider the case where an AI service provider is malicious: Suppose an audio recording submitted as court evidence is genuine, but the defendant claims it is a deepfake. Then, an AI service provider colluding with the defendant provides a false token to the court, causing a watermark to be detected so that it is recognized as fabricated. Our proposed new security notion, disclosure soundness, guarantees that such attacks are impossible.

The watermark proposed in this study can be applied not only to the scenario where deepfake media are used as court evidence, but also to other scenarios such as detecting prohibited AI usage and determining copyright ownership. Someone using AI in reports, contests, or other settings where AI usage is prohibited might try to hide evidence of AI use; however, since watermarks of undisclosed content cannot be detected, it becomes difficult even to notice that watermarks are embedded. Furthermore, when claiming copyright while using AI-generated

content almost as-is, it becomes difficult to remove watermarks through mechanical processing and checks without reflecting one’s own creativity.

1.5 Related Work

Christ and Gunn [5] introduce the notion of undetectability and show the construction of pseudorandom error correcting codes. A notable application of their work is a watermarking scheme for LLMs.

Gunn et al. [13] propose a watermark scheme for generative image models using pseudorandom codes. Their construction is different from ours in that they target diffusion models rather than autoregressive models.

Huang et al. [14] propose multi-designated detector watermarks. Their goal is to build a watermarking scheme under which verifiers are restricted. They attain such a scheme by embedding a variant of designated verifier signatures [3, 15]. Their goal is somewhat similar to ours, but our work differs in that our goal is to disclose the provenance of single data without granting the ability to verify provenance for all data, not to restrict verifiers.

There are several recent works [7, 8, 4] to achieve the CCA security of pseudorandom codes, which means that undetectability is ensured even if an adversary has access to the decoding oracle in the security game. Although this direction of research is very important, we do not require such strong primitives to construct our proposed scheme.

2 Preliminary

We denote probabilistic polynomial time by PPT. We denote a security parameter by λ . Let $\text{negl}(\lambda)$ denote a negligible function in λ . Given a PPT algorithm $\mathcal{A}$, let $\mathcal{A}(x; r)$ denote its output when given x as input and r as a random tape. For a bit string ζ , the bit string obtained by extracting the bits from the a -th to the b -th position is denoted as $\zeta[a : b]$. For convenience, we define a function Blocks_n that splits a given bit string into segments of n bits each.

2.1 Language Models

Following [5, 6, 16], we define a language model as follows:

Definition 1 (Language Model). *A language model $\text{Model} : \{0, 1\}^* \rightarrow [0, 1]$ takes a bit string $(x_1, \dots, x_t)$ and outputs the probability that the next bit equals 1.*

Given a prompt $x \in \{0, 1\}^*$, we obtain a response $u = (u[1], u[2], \dots) \leftarrow \overline{\text{Model}}(x)$ by autoregressive sampling initialized with x . Each bit $u[j]$ of the response is sampled as $u[j] \leftarrow \text{Model}(x, u[1 : j - 1])$. For $n \in \mathbb{N}$, we write $\overline{\text{Model}}_n(x)$ for the distribution obtained from autoregressively sampling the next n bits.

2.2 Robust Signatures

We use robust signatures proposed in [16]. Robust signatures are valid on messages even if the messages are modified to some extent.

Definition 2 (Robust Signature). *A robust signature scheme SIG consists of the following three algorithms.*

- KeyGen(pp) $\rightarrow$ (vk, sk): *Given a public parameter pp as input, it outputs a verification / signing key pair (vk, sk).*
- Sign(sk, m, r) $\rightarrow$ σ : *Given a signing key sk, a message m, and a random tape r as input, it outputs a signature σ .*
- Ver(vk, ζ , σ) $\rightarrow$ 1/0: *Given a verification key vk, a message ζ , and a signature σ as input, it outputs 1 (accept) or 0 (reject).*

A robust signature scheme SIG should satisfy the following two properties, robustness and unforgeability. In the games of robust signatures, an adversary can access the signing oracle Sign_{sk} that takes a message m, computes a signature $\sigma \leftarrow \text{Sign}(\text{sk}, m)$, updates the list $L_{\text{sig}} := L_{\text{sig}} \cup \{(m, \sigma)\}$, and returns the signature σ to the adversary.

Definition 3 (Robustness). *A robust signature scheme SIG satisfies robustness if for any PPT algorithm $\mathcal{A}$, it holds that $\text{Adv}_{\text{SIG}, \mathcal{A}}^{\text{Rob}}(\lambda) := \Pr[\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{Rob}}(\lambda) = 1] \leq \text{negl}(\lambda)$ where $\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{Rob}}(\lambda)$ is defined as follows:*

$\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{Rob}}(\lambda)$

$L_{\text{sig}} := \emptyset$

$(\text{sk}, \text{vk}) \leftarrow \text{KeyGen}(1^\lambda)$

$\mathcal{A}^{\text{Sign}_{\text{sk}}}(\text{vk}) \rightarrow (\zeta, \sigma)$

Output 1 if $\exists (y, \sigma) \in L_{\text{sig}}, \Phi(\zeta, y) = 1 \wedge \text{Ver}(\text{vk}, \zeta, \sigma) = 0$

else Output 0

Definition 4 (Unforgeability). *A robust signature scheme SIG satisfies unforgeability if for any PPT algorithm $\mathcal{A}$, it holds that $\text{Adv}_{\text{SIG}, \mathcal{A}}^{\text{EUF}}(\lambda) := \Pr[\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{EUF}}(\lambda) = 1] \leq \text{negl}(\lambda)$ where $\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{EUF}}(\lambda)$ is defined as follows:*

$\text{Game}_{\text{SIG}, \mathcal{A}}^{\text{EUF}}(\lambda)$

$L_{\text{sig}} := \emptyset$

$(\text{sk}, \text{vk}) \leftarrow \text{KeyGen}(1^\lambda)$

$(\zeta, \sigma) \leftarrow \mathcal{A}^{\text{Sign}_{\text{sk}}}(\text{vk})$

Output 1 if $\text{Ver}(\text{vk}, \zeta, \sigma) = 1 \wedge (\forall (y, \sigma) \in L_{\text{sig}}, \Phi(\zeta, y) = 0)$

else Output 0

2.3 Block Steganography

Following [16], we abstract pseudorandom codes [5] as a block steganography scheme.

Definition 5 (Block Steganography). *A block steganography scheme STG consists of the following three algorithms.*

$\text{Gen} \rightarrow (\text{sk}, \text{vk})$: *Given a security parameter λ as input, it outputs a verification key and a secret key (vk, sk) . If STG is symmetric, then $\text{vk} = \text{sk}$.*
 $\text{Emb}(\text{sk}, \text{m}, \pi) \rightarrow y$: *Given a secret key sk , a context $\pi \in \{0, 1\}^*$, and a message $\text{m} \in \{0, 1\}^k$ as input, it outputs a block $y \in \{0, 1\}^n$.*
 $\text{Dec}(\text{vk}, \zeta) \rightarrow \text{m} / \perp$: *Given a verification key vk and a text $\zeta \in \{0, 1\}^n$ as input, it outputs a decoded message or a failure symbol $\text{m} \in \{0, 1\}^k \cup \{\perp\}$.*

A block steganography scheme STG should satisfy the following two properties, undetectability and block robustness.

Definition 6 (Undetectability). *A block steganography scheme STG satisfies undetectability if for any PPT algorithm $\mathcal{A}$, it holds that*

$$\begin{aligned} \text{Adv}_{\text{STG}, \mathcal{A}}^{\text{Udet}}(\lambda) &:= \\ |\Pr[\mathcal{A}^{\text{Model}, \text{Emb}(\text{sk}, \cdot, \cdot)}(1^\lambda) = 1 | (\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)] - \Pr[\mathcal{A}^{\text{Model}, \overline{\text{Model}}_n}(1^\lambda) = 1]| &\leq \text{negl}(\lambda). \end{aligned}$$

To discuss the robustness of steganography schemes, we introduce the closeness predicate $\Phi : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}$. It defines what it means for two to be close to each other. The description of Φ depends on the instantiation of a block steganography scheme.

Definition 7 (Block Robustness). *A block steganography scheme STG satisfies block robustness if for any PPT algorithm $\mathcal{A}$, it holds that*

$$\begin{aligned} \text{Adv}_{\text{STG}, \mathcal{A}}^{\text{Rob}}(\lambda) &:= \Pr[\exists(\pi, \text{m}, y), \Phi(\zeta, y) = 1 \wedge \text{Dec}(\text{vk}, \zeta) \neq \text{m} \\ |(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda); \zeta \leftarrow \mathcal{A}^{\text{Emb}(\text{sk}, \cdot, \cdot), \text{Dec}(\text{vk}, \cdot)}(1^\lambda)] &\leq \text{negl}(\lambda). \end{aligned}$$

2.4 Non-Interactive Zero-Knowledge Arguments of Knowledge

Definition 8 (Non-Interactive Zero-Knowledge Argument of Knowledge). *For a polynomial time recognizable relation $\mathcal{R}$ between a statement x and a witness w and an NP language $\mathcal{L}$ that accepts x , non-interactive zero-knowledge argument NIZK consists of the following three algorithms:*

$\text{Setup}(1^\lambda) \rightarrow \text{crs}$: *Given a security parameter 1^λ as input, it outputs a common reference string crs .*
 $\text{Prove}(\text{crs}, x, w) \rightarrow \pi$: *Given a common reference string crs , a statement x , and a witness w as input, it outputs a proof π .*

$\text{Ver}(\text{crs}, x, \pi) \rightarrow 1/0$: Given a common reference string crs , a statement x and a proof π as input, it outputs 1 (accept) or 0 (reject).

Definition 9 (Completeness). A non-interactive zero-knowledge argument NIZK satisfies completeness, if for any pair $(x, w) \in \mathcal{R}$ and any $\text{crs} \leftarrow \text{Setup}(1^\lambda)$, it holds that $\Pr[\text{Ver}(\text{crs}, x, \text{Prove}(\text{crs}, x, w)) = 1] = 1$.

Definition 10 (Soundness). A non-interactive zero-knowledge argument NIZK satisfies soundness, if for any $\lambda \in \mathbb{N}$ and any PPT algorithm $\mathcal{A}$, it holds that $\Pr[x \notin \mathcal{L} \wedge \text{Ver}(\text{crs}, x, \pi) = 1 \mid \text{crs} \leftarrow \text{Setup}(1^\lambda); (x, \pi) \leftarrow \mathcal{A}(1^\lambda, \text{crs})] \leq \text{negl}(\lambda)$.

Definition 11 (Knowledge Soundness). A non-interactive zero-knowledge argument NIZK satisfies knowledge soundness, if for any $\lambda \in \mathbb{N}$ and any PPT algorithm $\mathcal{A}$, there exists an extractor $\mathcal{E}$, such that $\Pr[(x, w) \notin \mathcal{R} \wedge \text{Ver}(\text{crs}, x, \pi) = 1 \mid \text{crs} \leftarrow \text{Setup}(1^\lambda); (x, \pi, w) \leftarrow \mathcal{A} \parallel \mathcal{E}_{\mathcal{A}}(1^\lambda, \text{crs})] \leq \text{negl}(\lambda)$.

Definition 12 (Zero-Knowledge). A non-interactive zero-knowledge argument NIZK satisfies zero-knowledge, if for any PPT algorithm $\mathcal{A}$, there exists a PPT simulator $\text{Sim} = (\text{Sim}_1, \text{Sim}_2)$ such that if we run $\text{crs} \leftarrow \text{Setup}(1^\lambda)$ and $(\overline{\text{crs}}, \overline{\tau}) \leftarrow \text{Sim}_1(1^\lambda)$, then it holds that $\text{Adv}_{\text{NIZK}, \mathcal{A}}^{\text{ZK}}(\lambda) := |\Pr[\mathcal{A}^{\mathcal{O}_0(\text{crs}, \cdot, \cdot)}(1^\lambda, \text{crs}) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_1(\overline{\text{crs}}, \overline{\tau}, \cdot, \cdot)}(1^\lambda, \overline{\text{crs}}) = 1]| \leq \text{negl}(\lambda)$ where $\mathcal{O}_0(\text{crs}, x, w)$ outputs $\text{Prove}(\text{crs}, x, w)$ if $(x, w) \in \mathcal{R}$ and $\perp$ otherwise, and $\mathcal{O}_1(\overline{\text{crs}}, \overline{\tau}, x, w)$ outputs $\text{Sim}_2(\overline{\text{crs}}, \overline{\tau}, x)$ if $(x, w) \in \mathcal{R}$ and $\perp$ otherwise.

2.5 Commitments

Definition 13. A commitment scheme COM consists of the following algorithm.

$\text{Commit}(\text{m}; r) \rightarrow \text{com}$: Given a message m and a randomness r , it outputs a commitment com .

A commitment scheme COM should satisfy the following two properties: binding and hiding.

Definition 14 (Binding). A commitment scheme COM is binding if for any PPT algorithm $\mathcal{A}$, it holds that

$$\begin{aligned} \text{Adv}_{\text{COM}, \mathcal{A}}^{\text{Bnd}}(\lambda) &:= \Pr[\text{m} \neq \text{m}' \wedge \text{Commit}(\text{m}; r) = \text{Commit}(\text{m}'; r') = \text{com} \\ &\quad \mid (\text{com}, \text{m}, r, \text{m}', r') \leftarrow \mathcal{A}(1^\lambda)] \leq \text{negl}(\lambda). \end{aligned}$$

Definition 15 (Hiding). A commitment scheme COM is hiding if for any stateful PPT algorithm $\mathcal{A}$, it holds that

$$\begin{aligned} \text{Adv}_{\text{COM}, \mathcal{A}}^{\text{Hid}}(\lambda) &:= |\Pr[b = b'] \\ &\quad \mid (\text{m}_0, \text{m}_1) \leftarrow \mathcal{A}(1^\lambda); b \leftarrow \{0, 1\}; r \leftarrow \{0, 1\}^{\text{poly}(\lambda)}; \text{com} := \text{Commit}(\text{m}_b; r); b' \leftarrow \mathcal{A}(\text{com}) \\ &\quad - \frac{1}{2}| \leq \text{negl}(\lambda). \end{aligned}$$

3 Watermarks with Selective Disclosability

In this section, we discuss the syntax and security definitions of watermarks with selective disclosability.

Definition 16 (Watermarks with Selective Disclosability). *Watermarks with selective disclosability consists of the following five algorithms.*

$\text{Setup}(1^\lambda) \rightarrow \text{pp}$: Given a security parameter λ , it outputs a public parameter pp . We implicitly assume that pp is given to all the algorithms below.
 $\text{KeyGen}(1^\lambda) \rightarrow (\text{pk}, \text{sk})$: Given a security parameter λ , it outputs a key pair (pk, sk) .
 $\text{Wat}(\text{sk}, x) \rightarrow y$: Given a secret key sk and a prompt x , it outputs a response y .
 $\text{Disclose}(\text{sk}, \text{pk}, y) \rightarrow \alpha$: Given a secret key sk , the corresponding public key pk , and y , it outputs a bit string α .
 $\text{Ver}(\text{pk}, \alpha, \zeta) \rightarrow 0/1$: Given a public key pk , α and a text ζ , it outputs 0 (reject) or 1 (accept).

3.1 Robustness

Robustness requires that parties given tokens should be able to identify watermarked content even if it has been altered from the original state to some extent. This captures the fact that users can slightly modify AI-generated content without creativity or ingenuity and that such content should be recognized as AI-generated.

In the definition, robustness requires that altered content be verified as long as the closeness predicate Φ outputs 1. To align with our scheme, the disclosing oracle that outputs verification tokens is also given to the adversary in the security game.

To discuss the robustness of a watermark scheme, we first introduce a block closeness notion EBC, following [16].

Definition 17. *Given two bit strings ζ^* and y and a closeness predicate $\Phi : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}$, $\text{EBC}[\Phi, n](\zeta^*, y)$ parses $(\zeta_1^*, \dots, \zeta_r^*) = \text{Blocks}_n(\zeta^*)$ and $(y_1, \dots, y_{\lceil |y|/n \rceil}) = \text{Blocks}_n(y)$ and then outputs 1 if $(\exists r \in \mathbb{N}, |\zeta^*| = rn) \wedge (\exists t \in \mathbb{N}, \forall j \in [r], \Phi(\zeta_j^*, y_{t+j-1}) = 1)$, otherwise outputs 0.*

Definition 18 (Robustness). *A watermark scheme Π satisfies $(\text{EBC}[\Phi, n], kn)$ -robustness if for any PPT algorithm $\mathcal{A}$, it holds that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{Rob}}(\lambda) := \Pr[\text{Game}_{\Pi, \mathcal{A}}^{(\text{EBC}[\Phi, n], kn) - \text{Rob}}(\lambda) = 1] \leq \text{negl}(\lambda)$ where $\text{Game}_{\Pi, \mathcal{A}}^{(\text{EBC}[\Phi, n], kn) - \text{Rob}}(\lambda)$ is defined as follows:*

$\text{Game}_{\Pi, \mathcal{A}}^{(\text{EBC}[\Phi, n], kn) - \text{Rob}}(\lambda)$
 $\text{pp} \leftarrow \text{Setup}(1^\lambda)$
 $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$
 $\zeta \leftarrow \mathcal{A}^{\text{Wat}_{\text{sk}}(\text{sk}, \cdot), \text{Disclose}(\text{sk}, \text{pk}, \cdot)}(\text{pp}, \text{pk})$
 $\alpha \leftarrow \text{Disclose}(\text{pp}, \text{sk}, \text{pk}, \zeta)$
 Output 1 if $\exists y \in \text{L}_{\text{wat}}, \exists \zeta^* \subset \zeta, \zeta^* = |kn| \wedge \text{EBC}[\Phi, n](\zeta^*, y) = 1 \wedge \text{Ver}(\text{pp}, \text{pk}, \zeta, \alpha) = 0$
 else Output 0

Oracles available in the above game are defined as follows: The watermarking oracle $\text{Wat}(\text{sk}, \cdot)$ takes a prompt x , returns $y \leftarrow \text{Wat}(\text{sk}, x)$, and updates $\text{L}_{\text{sig}} := \text{L}_{\text{sig}} \cup \{y\}$. The disclosing oracle $\text{Disclose}(\text{sk}, \text{pk}, \cdot)$ first takes a bit string ζ . If there exist $y \in \text{L}_{\text{sig}}$ and $\zeta^* \in \zeta$ s.t. $|\zeta^*| = kn \wedge \text{EBC}[\Phi, n](y, \zeta^*) = 1$, then it returns $\alpha \leftarrow \text{Disclose}(\text{sk}, \text{pk}, \zeta)$. Otherwise, it outputs $\perp$.

3.2 Unforgeability

Unforgeability ensures that human-created content should not be recognized as watermarked, whether it is intentionally generated or not. This captures the following two cases: First, content generated by honest humans should not be verified as watermarked by accident. Second, malicious users, who want to frame a generative AI for creating harmful content, cannot embed a valid watermark that indicates the content has been generated by the AI. It is natural in both cases that users can get validly watermarked content by interacting with generative AI, so the definition captures this.

In the definition, unforgeability means that without a secret key, no one can create watermarked content that is far from the observed content generated by AI. Note that content too similar to AI-generated content should be recognized as watermarked by robustness. To capture the situation where users can get watermarked content by interacting with a generative AI, the watermarking oracle is available to an adversary in the game. Additionally, the game gives the disclosing oracle to the adversary since we would like to prevent even parties holding verification tokens for other outputs from creating watermarked content.

Definition 19 (Unforgeability). *A watermark scheme Π satisfies unforgeability if for any PPT algorithm $\mathcal{A}$, it holds that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{Unf}}(\lambda) := \Pr[\text{Game}_{\Pi, \mathcal{A}}^{\text{Unf}}(\lambda) = 1] \leq \text{negl}(\lambda)$ where $\text{Game}_{\Pi, \mathcal{A}}^{\text{Unf}}(\lambda)$ is defined as follows:*

$\text{Game}_{\Pi, \mathcal{A}}^{\text{Unf}}(\lambda)$
 $\text{pp} \leftarrow \text{Setup}(1^\lambda)$
 $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$
 $\zeta \leftarrow \mathcal{A}^{\text{Wat}(\text{sk}, \cdot), \text{Disclose}(\text{sk}, \text{pk}, \cdot)}(\text{pp}, \text{pk})$
 $\alpha \leftarrow \text{Disclose}(\text{pk}, \text{sk}, \zeta)$
 Output 1 if $\text{Ver}(\text{pk}, \zeta, \alpha) = 1 \wedge (\forall y \in \text{L}_{\text{wat}}, \forall \zeta^* \in \zeta, \Phi(\zeta^*, y) = 0 \vee |\zeta^*| \neq n)$
 else Output 0

3.3 Undetectability of Undisclosed Data

Undetectability of undisclosed data requires that the embedding of watermarks should not introduce any practical change to outputs, even against parties holding verification tokens for other data. This property captures two aspects: First, we would like to ensure security where malicious users try to eliminate watermarks from AI-generated content and checks whether the modified content are detected as watermarked or not. Second, the fact that no one without a secret key

tell watermarked outputs from original outputs means that there is practically no degradation of output quality of the AI model.

We formalize this property by the two-stage security game. In the first stage, an adversary $\mathcal{A}$ can query prompts and obtain corresponding watermarked outputs and verification tokens. In the second stage, the adversary $\mathcal{A}$ is given access to the original model or the watermarking oracle and required to answer which oracle it interacts with. If the first stage is removed from the game, then it is equal to the existing notion of undetectability.

Definition 20 (Undetectability of undisclosed data). *A watermark scheme Π satisfies undetectability for undisclosed data if for any PPT algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, it holds that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{Und}}(\lambda) := |\Pr[\text{Game}_{\Pi, \mathcal{A}}^{\text{Und}}(\lambda) = 1] - \frac{1}{2}| \leq \text{negl}(\lambda)$ where $\text{Game}_{\Pi, \mathcal{A}}^{\text{Und}}(\lambda)$ is defined as follows:*

$\text{Game}_{\Pi, \mathcal{A}}^{\text{Und}}(\lambda)$

$\text{pp} \leftarrow \text{Setup}(1^\lambda)$
 $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$
 $b \leftarrow \{0, 1\}$
 $s \leftarrow \mathcal{A}_1^{\text{Model}, \text{WatAndDisclose}(\text{sk}, \text{pk}, \cdot)}(\text{pp}, \text{pk})$
 $b' \leftarrow \mathcal{A}_2^{\text{Model}, \mathcal{O}_{\text{cha}}(b, \text{sk}, \cdot)}(s)$
 Output 1 if $b = b'$:
 else Output 0

In the security game, oracles work as follows: the challenge oracle $\mathcal{O}_{\text{cha}}(b, \text{sk}, \cdot)$ takes x as an input and outputs $\text{Model}(x)$ if $b = 0$ and $\text{Wat}(\text{sk}, x)$ if $b = 1$. The watermarking and disclosing oracle $\text{WatAndDisclose}(\text{sk}, \text{pk}, \cdot)$ takes a prompt x , computes $y \leftarrow \text{Wat}(\text{pp}, \text{sk}, x)$ and $\pi \leftarrow \text{Disclose}(\text{pp}, \text{sk}, \text{pk}, y)$, and returns (y, π) to the adversary $\mathcal{A}$.

3.4 Disclosure Soundness

Disclosure soundness guarantees that no malicious AI provider can conjure up a valid watermark from non-watermarked content. This property prevents AI providers from proving that human-created content has been generated by its AI. One plausible scenario is that when a defendant is accused based on real security camera footage, a corrupted AI provider tries to prove that the evidence is fake.

In the definition, we require that no PPT adversary can produce a (maliciously generated) verification token and non-watermarked content, such that the content is recognized as watermarked. First, an adversary is given access to the key generation oracle in the security game, which corresponds to the adversary being able to choose its key. We define non-watermarked output as data that is not detected as watermarked if a verification token is honestly generated. To ensure this security property for all data, we allow the adversary to choose non-watermarked content, although in a real-world scenario content that

a provider wants to be recognized as watermarked will not be generated by the provider.

Definition 21 (Disclosure Soundness). *A watermark scheme Π satisfies disclosure soundness if for any PPT algorithm $\mathcal{A}$, it holds that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{DS}}(\lambda) := \Pr[\text{Game}_{\Pi, \mathcal{A}}^{\text{DS}}(\lambda) = 1] \leq \text{negl}(\lambda)$ where $\text{Game}_{\Pi, \mathcal{A}}^{\text{DS}}(\lambda)$ is defined as follows:*

$$\begin{array}{l} \text{Game}_{\Pi, \mathcal{A}}^{\text{DS}}(\lambda) \\ \hline \text{pp} \leftarrow \text{Setup}(1^\lambda) \\ (\text{pk}, \zeta, \alpha) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{pp}, \cdot)}(\text{pp}) \\ \text{sk} := \text{L}_{\text{key}}[\text{pk}] \\ \beta \leftarrow \text{Disclose}(\text{sk}, \text{pk}, \zeta) \\ \text{Output } 1 \text{ if } \text{Ver}(\text{pk}, \zeta, \beta) = 0 \wedge \text{Ver}(\text{pk}, \zeta, \alpha) = 1 \\ \text{Otherwise Output } 0 \end{array}$$

where the key generation oracle $\text{KeyGen}(\text{pp}; \cdot)$ takes a random tape r , generates a key pair $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(\text{pp}; r)$, updates a list $\text{L}_{\text{key}} := \text{L}_{\text{key}} \cup \{(\text{pk}, \text{sk})\}$, and returns (sk, pk) to an adversary.

4 Our Proposed Scheme

In this section, we show the construction of our proposed scheme and prove that it satisfies the security properties discussed in the previous section.

4.1 Construction

Our approach is based on [16] and applicable to any autoregressive generative AI model, including LLMs. We use a robust signature scheme SIG, a symmetric key-steganography scheme STG, a commitment scheme COM, and a non-interactive zero-knowledge argument NIZK. Although the disclosing and the verification processes involve generating and verifying a proof, our construction introduces almost no additional cost to the watermarking process, which accounts for the overwhelming majority of the overall operation.

To introduce our proposed solution, we start with the baseline scheme by Lin et al. [16]. They instantiate robust signatures from strongly unforgeable signatures and property-preserving hashing [2] and show the construction of unforgeable watermarks with them. Consider the block generation process with an autoregressive model: The model first signs the preceding block with a robust signature scheme and then embeds the signature into the next block with steganography, which is an abstraction of PRC. During verification, one tries to decode blocks with the corresponding key. If the verifier can extract a signature from a block, then it checks whether the signature is valid on the preceding block.

To achieve our goal, we then use a proof of knowledge for an embedded signature. The approach is simple, but we should pay attention to the side effect

of hiding the key itself. Since steganography schemes ensure nothing about the situation where a secret key is replaced, the provider might be able to replace a watermarking key in private and generate a meaningless proof for given data. With this naive approach, the malicious provider may fool verifiers into believing that authentic media are generated by the AI model, without breaking the soundness of a proof system. To prevent these types of attacks, a model owner commits to their key first. Then we can reduce the security of the watermarking scheme to the property of a commitment scheme.

$$\frac{\text{Setup}(1^\lambda) \rightarrow \text{pp}}{\text{NIZK} . \text{Setup}(1^\lambda) \rightarrow \text{crs}} \\ \text{pp} := \text{crs}$$

$$\frac{\text{KeyGen}(\text{pp}) \rightarrow (\text{pk}, \text{sk})}{\begin{array}{l} r \leftarrow \{0, 1\}^{\text{poly}(\lambda)} \\ \text{SIG} . \text{KeyGen}(1^\lambda) \rightarrow (\text{sk}_{\text{SIG}}, \text{pk}_{\text{SIG}}) \\ \text{STG} . \text{KeyGen}(1^\lambda) \rightarrow \text{k} \\ \text{com} := \text{COM} . \text{Commit}(\text{k}; r) \\ \text{sk} := (\text{sk}_{\text{SIG}}, \text{k}, r) \\ \text{pk} := (\text{pk}_{\text{SIG}}, \text{com}) \end{array}}$$

The setup algorithm **Setup** has to be executed by a trusted party. If transparent NIZK is adopted, then this algorithm can be omitted and consolidated into **KeyGen**: it takes a security parameter 1^λ instead of a public parameter **pp**, first runs **Setup**(1^λ), includes **pp** in **pk**, and does the same as **KeyGen** otherwise.

$$\frac{\text{Wat}(\text{pp}, \text{sk}, x) \rightarrow y}{\begin{array}{l} y_1 \leftarrow \text{Model}_n(x) \\ \text{for } i = 2 \text{ to } t \\ \quad \sigma_{i-1} \leftarrow \text{SIG} . \text{Sign}(\text{sk}_{\text{SIG}}, y_{i-1}) \\ \quad y_i \leftarrow \text{STG} . \text{Emb}(\text{k}, (x, y_1, \dots, y_{i-1}), \sigma_{i-1}) \\ y := y_1 || \dots || y_t \end{array}}$$

The disclosing algorithm **Disclose** generates and the verification algorithm **Ver** verifies a proof of the following language $L := \{(\text{pk}_{\text{SIG}}, \text{com}, \zeta) \mid \text{com} = \text{Commit}(\text{k}; r) \wedge \sigma = \text{STG} . \text{Dec}(\text{k}, \zeta) \wedge \text{SIG} . \text{Ver}(\text{pk}_{\text{SIG}}, \zeta[i : i + n - 1], \sigma) = 1\}$ where a witness is (k, σ, i, r) .

$$\frac{\text{Disclose}(\text{pp}, \text{sk}, \text{pk}, \zeta) \rightarrow \alpha}{\begin{array}{l} \text{for } i = 1 \text{ to } |\zeta| - 2n + 1 \\ \quad \sigma_i \leftarrow \text{STG} . \text{Dec}(\text{k}, \zeta[i + n : i + 2n - 1]) \\ \quad \text{if } \text{SIG} . \text{Ver}(\text{pk}_{\text{SIG}}, \zeta[i : i + n - 1], \sigma_i) = 1 \\ \quad \quad \text{NIZK} . \text{Prove}(\text{crs}, (\text{pk}_{\text{SIG}}, \text{com}, \zeta), (\text{k}, \sigma, i, r)) \rightarrow \pi \\ \quad \text{Output } \pi \end{array}}$$

$$\begin{array}{l}
\text{Ver}(\text{pp}, \text{pk}, \alpha, \zeta) \rightarrow 0/1 \\
\hline
(\text{pk}_{\text{SIG}}, \text{com}) := \text{pk} \\
\pi := \alpha \\
\text{Output 1 if } \text{NIZK} . \text{Ver}(\text{crs}, (\text{pk}_{\text{SIG}}, \text{com}, \zeta), \pi) \\
\text{Otherwise Output 0}
\end{array}$$

4.2 Security Proof

In this section, we give the formal proofs of the four security properties, i.e., robustness, unforgeability, undetectability against unauthorized users, and disclosure soundness.

Theorem 1 (Robustness). *The $(\text{EBC}[\Phi, n], 2n)$ -robustness of Π is implied by the robustness of SIG, the block robustness of STG, the hiding property of COM, and the zero-knowledge and completeness of NIZK.*

Proof. We introduce the following hybrid games.

Game0: The original $(\text{EBC}[\Phi, n], 2n)$ -robustness game.

Game1: This game outputs 0 if $\forall y \in \mathcal{L}_{\text{wat}}, \forall \zeta^* \subset \zeta, \zeta^* \neq |kn| \vee \text{EBC}[\Phi, n](\zeta^*, y) = 0$ before the challenger runs Disclose.

Game2: When the challenger finds the first $2n$ -bit string ζ^* s.t. $\exists y \in \mathcal{L}_{\text{wat}}, \zeta^* \subset \zeta, \text{EBC}[\Phi, n](\zeta^*, y) = 1$, it uses ζ^* as ζ after passing the condition check introduced by Game1.

Game3: The challenger produces a simulated proof when responding to a query to the disclosing oracle. We can describe this formally as follows: First, the challenger of the game computes $(\overline{\text{crs}}, \overline{\tau}) \leftarrow \text{Sim}_1(1^\lambda)$ instead of computing $\text{crs} \leftarrow \text{NIZK} . \text{Setup}(1^\lambda)$ and runs an adversary by giving it as pp . When the challenger takes ζ as a query to the disclosing oracle and there exists an index i s.t. $\text{SIG} . \text{Ver}(\text{pk}_{\text{SIG}}, \zeta[i : i+n-1], \sigma_i) = 1$ where $\sigma_i \leftarrow \text{STG} . \text{Dec}(\text{k}, \zeta[i+n : i+2n-1])$, then, instead of computing $\pi \leftarrow \text{NIZK} . \text{Prove}(\text{crs}, (\text{pk}_{\text{SIG}}, \text{com}, \zeta), (\text{k}, r, i, \sigma_i))$, it computes $\overline{\pi} \leftarrow \text{Sim}_2(\text{crs}, \tau, (\text{pk}_{\text{SIG}}, \text{com}, \zeta))$, and returns $\alpha := \overline{\pi}$ to the adversary.

Game4: The challenger generates a commitment by $\text{com} \leftarrow \text{COM} . \text{Commit}(0)$ instead of $\text{com} \leftarrow \text{COM} . \text{Commit}(\text{k})$ where k is a key for STG.

Game5: The challenger keeps track of a tuple of a response y , an embedded signature σ_i , and an index i that indicates the embedded position of the response in the list $\mathcal{L}_{\text{emb}}$ as the form of (y, σ_i, i) when simulating the watermarking oracle.

Game6: When running Disclose right before the winning condition check, the challenger outputs 0 if it cannot extract the same signature σ as in the list $\mathcal{L}_{\text{emb}}$ from $\zeta^*[n+1 : 2n]$.

Game7: When running Disclose right before the winning condition check, the challenger outputs 0 if the extracted signature σ from $\zeta[n+1 : 2n]$ is in the list $\mathcal{L}_{\text{emb}}$ but does not satisfy $\text{SIG} . \text{Ver}(\text{pk}_{\text{SIG}}, \zeta[1 : n], \sigma)$.

Lemma 1. *For any adversary $\mathcal{A}$, the advantages of Game0 and Game1 remain the same.*

Proof. If the output ζ does not satisfy the condition that $\exists y \in \mathcal{L}_{\text{wat}}, \zeta^* \subset \zeta$, $\text{EBC}[\Phi, n](\zeta^*, y) = 1$, then the output will never make the game output 1. This change just introduces an early additional check for the winning condition and thus does not introduce any difference in the winning probability.

Lemma 2. *For any adversary $\mathcal{A}$, we have $\text{Adv}_{1,\mathcal{A}}(\lambda) \leq \text{Adv}_{2,\mathcal{A}}(\lambda)$.*

Proof. Even if the challenger cannot extract a signature σ that passes the verification of SIG from the first $2n$ -bit string ζ^* close to y , it may be able to extract such a signature from another part of ζ in Game1. Thus, winning Game2 is easier than or equal to winning Game1.

Lemma 3. *If there exists a PPT adversary $\mathcal{A}$ that makes the difference between the advantages of Game2 and Game3, then we can construct $\mathcal{B}$ that breaks the zero knowledge of NIZK.*

Proof. First, $\mathcal{B}$ receives crs or $\overline{\text{crs}}$ from the challenger and sets pp to it. Then, $\mathcal{B}$ runs $\text{KeyGen}(\text{pp}) \rightarrow (\text{pk}, \text{sk})$ and gives (pk, pp) to the adversary $\mathcal{A}$. If $\mathcal{B}$ takes a query x to the watermarking oracle, then it honestly computes $y \leftarrow \text{Wat}(\text{pp}, \text{sk}, x)$. If it takes a query ζ to the disclosing oracle, then it honestly computes $\alpha \leftarrow \text{Disclose}(\text{pp}, \text{sk}, \text{pk}, \zeta)$ except that it queries $(\text{pk}_{\text{SIG}}, \text{com}, \zeta)$ to the oracle of the zero-knowledge game and includes the response in α as π .

After $\mathcal{A}$ halts and outputs ζ , $\mathcal{B}$ checks the winning condition, outputs 1 if it is satisfied, and outputs 0 otherwise. We can confirm that $\mathcal{B}$ simulates Game2 if $\mathcal{B}$ is given crs and access to $\mathcal{O}_0$, and Game3 if $\mathcal{B}$ is given $\overline{\text{crs}}$ and access to $\mathcal{O}_1$. Thus, we have $|\text{Adv}_{2,\mathcal{A}}(\lambda) - \text{Adv}_{3,\mathcal{A}}(\lambda)| = |\Pr[\mathcal{B}^{\mathcal{O}_0(\text{crs}, \cdot, \cdot)}(1^\lambda, \text{crs}) = 1] - \Pr[\mathcal{B}^{\mathcal{O}_1(\overline{\text{crs}}, \cdot, \cdot)}(1^\lambda, \overline{\text{crs}}) = 1]| = \text{Adv}_{\text{NIZK}, \mathcal{B}}^{\text{ZK}}(\lambda) \leq \text{negl}(\lambda)$.

Lemma 4. *If there exists a PPT adversary $\mathcal{A}$ that makes the difference between the advantages of Game3 and Game4, then we can construct $\mathcal{B}$ that breaks the hiding property of COM.*

Proof. The algorithm $\mathcal{B}$ creates a key for STG as $k \leftarrow \text{STG}.\text{KeyGen}(1^\lambda)$ and sends $(0, k)$ to the challenger of the hiding game of COM. Then, it includes the response com in pk , honestly generates a key pair for Π otherwise, and runs $\mathcal{A}$ by giving (pk, pp) . If $\mathcal{A}$ wins the game, then $\mathcal{B}$ outputs 1, and otherwise it outputs 0 to the challenger. Then, we have $\text{Adv}_{\text{COM}, \mathcal{B}}^{\text{Hid}} = \Pr[b = b'] - \frac{1}{2} = \Pr[b' = 0|b = 0] \cdot \Pr[b = 0] + \Pr[b' = 1|b = 1] \cdot \Pr[b = 1] - \frac{1}{2} = \frac{1}{2} \cdot (\Pr[b' = 0|b = 0] + \Pr[b' = 1|b = 1]) - \frac{1}{2} = \frac{1}{2} \cdot (1 - \Pr[b' = 1|b = 0] + \Pr[b' = 1|b = 1]) - \frac{1}{2} = \frac{1}{2} \cdot (1 + \Pr[\text{Game4} = 1] - \Pr[\text{Game3} = 1]) - \frac{1}{2} = \frac{1}{2} \cdot (\Pr[\text{Game4} = 1] - \Pr[\text{Game3} = 1])$. Thus, we can conclude $\Pr[\text{Game4} = 1] - \Pr[\text{Game3} = 1] \leq 2 \cdot \text{Adv}_{\text{COM}, \mathcal{B}}^{\text{Hid}}(\lambda) \leq \text{negl}(\lambda)$.

Lemma 5. *For any adversary $\mathcal{A}$, the advantages of Game4 and Game5 remain the same.*

Proof. This change does not affect any winning condition.

Lemma 6. *If there exists a PPT adversary $\mathcal{A}$ that makes the difference between the advantages of Game5 and Game6 non-negligible, then we can construct a PPT adversary $\mathcal{B}$ that wins the block robustness game of STG with non-negligible probability.*

Proof. Given a security parameter 1^λ , $\mathcal{B}$ generates a key pair for Π except that $\mathcal{B}$ does not generate a key for STG and computes $\text{com} \leftarrow \text{COM}.\text{Commit}(0)$ instead of $\text{com} \leftarrow \text{COM}.\text{Commit}(k)$, and then runs $\mathcal{A}$ by giving (pp, pk) . When $\mathcal{A}$ makes a query x to the watermarking oracle $\text{Wat}(\text{sk}, \cdot)$, $\mathcal{B}$ honestly generates and returns a response y , making queries to the embedding oracle $\text{Emb}(k, \cdot)$ of the robustness game of STG. When $\mathcal{A}$ makes a query ζ to the disclosing oracle $\text{Disclose}(\text{sk}, \text{pk}, \cdot)$, $\mathcal{B}$ honestly generates and returns a response α , making queries to the decoding oracle $\text{Dec}(k, \cdot)$ of the robustness game of STG.

Suppose $\mathcal{A}$ outputs ζ s.t. Game6 outputs 0 but Game5 outputs 1. Then, $\mathcal{B}$ extracts from ζ the first $2n$ -bit string ζ^* s.t. $\text{EBC}[\Phi, n](\zeta^*, y) = 1$ and outputs $\zeta^*[n+1 : 2n]$ to the challenger of the robustness game of STG. Let t be the index satisfying $\text{EBC}[\Phi, n](\zeta^*, y[t : t+2n-1]) = 1$. Its oracle simulation ensures $(\sigma, y[t : t+2n-1]) \in \mathcal{L}_{\text{emb}}$, while the introduced change guarantees that $\text{Dec}(\text{vk}, \zeta^*[n+1 : 2n]) \neq \sigma$. Thus, $\mathcal{B}$ wins the block robustness game of STG.

Lemma 7. *If there exists a PPT adversary $\mathcal{A}$ that makes the difference between the advantages of Game6 and Game7 non-negligible, then we can construct a PPT adversary $\mathcal{B}$ that breaks the robustness of SIG.*

Proof. Given a security parameter 1^λ , $\mathcal{B}$ generates a key pair for Π except a key pair for STG and runs $\mathcal{A}$ by giving (pp, pk) .

When $\mathcal{A}$ makes a query x to the watermarking oracle $\text{Wat}(\text{sk}, \cdot)$, $\mathcal{B}$ honestly generates and returns a response y , making queries to the embedding oracle $\text{Emb}()$ of the robustness game of STG. Suppose $\mathcal{A}$ outputs ζ s.t. Game4 outputs 0 but Game3 outputs 1. Then, $\mathcal{B}$ extracts from ζ the first $2n$ -bit string ζ^* s.t. $\text{EBC}[\Phi, n](\zeta^*, y) = 1$ and outputs $\zeta^*[n+1 : 2n]$ to the challenger of the robustness game of STG. Let t be the index satisfying $\text{EBC}[\Phi, n](\zeta^*, y[t : t+2n-1]) = 1$. Its oracle simulation ensures $(\sigma, y[t : t+2n-1]) \in \mathcal{L}_{\text{emb}}$, while the introduced change guarantees that $\text{Dec}(\text{vk}, \zeta^*[n+1 : 2n]) \neq \sigma$. Thus, $\mathcal{B}$ wins the block robustness game of STG.

Lemma 8. *The advantage of Game7 equals 0 if NIZK satisfies the completeness.*

Proof. If $\mathcal{A}$ wins Game7, it means that $\text{NIZK}.\text{Ver}(\text{crs}, (\text{com}, \zeta[i+n : i+2n-1], \sigma_i), \pi) = 0$ in Ver where $\pi \leftarrow \text{NIZK}.\text{Prove}(\text{crs}, (\text{com}, \zeta[i+n : i+2n-1], \sigma_i), (k, r))$ in Disclose. This contradicts the completeness of NIZK.

From the above lemmata, we follow that $\text{Adv}_{0, \mathcal{A}}(\lambda) \leq \sum_{i=0}^6 |\text{Adv}_{i, \mathcal{A}}(\lambda) - \text{Adv}_{(i+1), \mathcal{A}}(\lambda)| \leq 2 \cdot \text{Adv}_{\text{COM}, \mathcal{B}_1}^{\text{Hid}}(\lambda) + \text{Adv}_{\text{NIZK}, \mathcal{B}_2}^{\text{ZK}}(\lambda) + \text{Adv}_{\text{STG}, \mathcal{B}_3}^{\text{Rob}}(\lambda) + \text{Adv}_{\text{SIG}, \mathcal{B}_4}^{\text{Rob}}(\lambda)$

Theorem 2 (Unforgeability). *The unforgeability of Π is implied by the unforgeability of SIG.*

Proof. If there exists $\mathcal{A}$ that wins the unforgeability game of Π , we can construct $\mathcal{B}$ that wins the unforgeability game of SIG .

Given a challenge public key pk_{SIG}^* from the challenger, $\mathcal{B}$ honestly generates pp and pk for Π except that it does not generate a key pair for SIG and instead includes pk_{SIG}^* in pk for Π . Then, it runs $\mathcal{A}$ by giving pp and pk and answers queries from $\mathcal{A}$. Although $\mathcal{B}$ does not know sk_{SIG}^* corresponding to the challenge public key pk_{SIG}^* , it can create a signature σ by making queries to the signing oracle of the unforgeability game of SIG . Thus, $\mathcal{B}$ can answer a query to the watermarking oracle. Also, it can honestly respond to a query to the disclosing oracle.

Suppose $\mathcal{A}$ outputs ζ that wins the unforgeability game of Π . According to the behavior of Ver , $\text{SIG.Ver}(\text{pk}_{\text{SIG}}, \zeta[i : i + n - 1], \sigma_i) = 1$ where $(i, \pi, \sigma_i) \leftarrow \text{Disclose}(\text{pk}, \text{sk}, \zeta)$. Then, $\mathcal{B}$ outputs $\zeta[i : i + n - 1]$ as a forgery to the challenger. The winning condition of the unforgeability game of Π ensures that for any $\zeta^* \in \zeta$ s.t. $|\zeta^*| = n$, we have $\Phi(\zeta^*, y) = 0$. Thus, it holds that $\forall (y, \sigma_i) \in \mathbf{L}_{\text{sig}}, \Phi(\zeta[i : i + n - 1], y) = 0$ in the unforgeability game of SIG and thus $\mathcal{B}$ wins the game.

Theorem 3 (Undetectability of Undisclosed Data). *The undetectability of Π is implied by the zero knowledge of NIZK , the undetectability of STG , and the hiding property of COM .*

Proof. We introduce the following hybrid games.

Game0: The original undetectability game.

Game1: Proofs that the oracle returns are generated by a simulator. First, the challenger runs $(\overline{\text{crs}}, \tau) \leftarrow \text{Sim}_1(1^\lambda)$. To answer a query x from $\mathcal{A}_1$, it first honestly generates $y \leftarrow \text{Wat}(\text{sk}, x)$, then simulates $\pi \leftarrow \text{Sim}_2(\overline{\text{crs}}, \tau, (\text{pk}_{\text{SIG}}, \text{com}, \zeta))$, and returns (y, π) .

Game2: The challenger computes $\text{com} \leftarrow \text{COM.Commit}(0)$ instead of $\text{com} \leftarrow \text{COM.Commit}(k)$.

Game3: When answering a query x from $\mathcal{A}_1$, the challenger responds with $y \leftarrow \text{Model}(x)$ if $b = 0$ and $y \leftarrow \text{Wat}(\text{sk}, x)$ if $b = 1$.

Due to the page limitation, we give only the following lemmata; the formal proof is given in Appendix ??.

Lemma 9. *The zero-knowledge of NIZK implies that the difference between the advantages of Game0 and Game1 is negligible.*

Proof. Suppose there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that makes the difference between the advantages of the two games non-negligible. Then, we can build another PPT adversary $\mathcal{B}$ that breaks the zero-knowledge of NIZK .

First, $\mathcal{B}$ is given crs which is produced by NIZK.Setup or NIZK.Sim_1 , includes it in pp , generates pk honestly, and runs $\mathcal{A}_1$ by giving (pp, pk) . To answer queries from $\mathcal{A}_1$, $\mathcal{B}$ honestly generates (y, α) except that π is obtained by querying $((\text{pk}_{\text{SIG}}, \text{com}, y), (k, \sigma, i, r))$ to its oracle.

After $\mathcal{A}_1$ outputs s , $\mathcal{B}$ samples a bit $b \leftarrow \{0, 1\}$. Next, $\mathcal{B}$ runs $\mathcal{A}_2$ by giving s . Then, $\mathcal{B}$ answers queries x from $\mathcal{A}_2$ by returning $y \leftarrow \text{Model}(x)$ if $b = 0$ and $y \leftarrow \text{Wat}(\text{sk}, x)$ if $b = 1$. Then, $\mathcal{A}_2$ outputs a bit b' and halts..

Finally, $\mathcal{B}$ outputs 0 if $b \neq b'$ and 1 if $b = b'$ to the challenger. We can confirm that $\mathcal{B}$ simulates Game0 if $\text{crs} \leftarrow \text{NIZK}.\text{Setup}(1^\lambda)$ and Game1 if $\text{crs} \leftarrow \text{NIZK}.\text{Sim}_1(1^\lambda)$. Thus, it holds that $\Pr[\text{Game0}_{\mathcal{A}}(\lambda) = 1] - \Pr[\text{Game1}_{\mathcal{A}}(\lambda) = 1] = \Pr[b = b' | \text{crs} \leftarrow \text{NIZK}.\text{Setup}(1^\lambda)] - \Pr[b = b' | \text{crs} \leftarrow \text{NIZK}.\text{Sim}_1(1^\lambda)] = \Pr[\mathcal{B}^{\mathcal{O}_0(\text{crs}, \cdot, \cdot)}(1^\lambda, \text{crs}) = 1] - \Pr[\mathcal{B}^{\mathcal{O}_1(\overline{\text{crs}}, \cdot, \cdot)}(1^\lambda, \overline{\text{crs}}) = 1] = \text{Adv}_{\text{NIZK}, \mathcal{B}}^{\text{ZK}}(\lambda) \leq \text{negl}(\lambda)$.

Lemma 10. *The hiding property of COM implies that the difference between the advantages of Game1 and Game2 is negligible.*

Proof. Suppose there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that makes the difference between the advantages of the two games non-negligible. Then, we can build another PPT adversary $\mathcal{B}$ that breaks the hiding property of COM.

First, $\mathcal{B}$ outputs $m_0 = k$ and $m_1 = 0$ to its challenger. Let b be a bit chosen by the challenger. Then $\mathcal{B}$ receives com_b where $\text{com}_0 \leftarrow \text{COM}.\text{Commit}(k)$ and $\text{com}_1 \leftarrow \text{COM}.\text{Commit}(0)$. Next, $\mathcal{B}$ includes com_b in pk and honestly generates pp and pk otherwise. $\mathcal{B}$ executes $\mathcal{A}_1$ by giving (pp, pk) and answers queries from it. Note that $\mathcal{B}$ can simulate proofs since it uses Sim_2 even though it does not know the content of the commitment. Upon $\mathcal{A}_1$ outputting s , $\mathcal{B}$ samples a random bit $d \leftarrow \{0, 1\}$ and hands s to $\mathcal{A}_2$. Then, $\mathcal{B}$ responds to queries x from $\mathcal{A}_2$ by replying $y \leftarrow \text{Model}(x)$ if $d = 0$ and $y \leftarrow \text{Wat}(\text{sk}, x)$ if $d = 1$. After making queries, $\mathcal{A}_2$ outputs a bit d' and halts.

Finally, $\mathcal{B}$ outputs $b' = 0$ if $d \neq d'$ and $b' = 1$ if $d = d'$ to the challenger.

We can confirm that $\mathcal{B}$ simulates Game1 if $\text{com} \leftarrow \text{COM}.\text{Commit}(k)$ and Game2 if $\text{com} \leftarrow \text{COM}.\text{Commit}(0)$. Thus, we have $|\Pr[\text{Game1}_{\mathcal{A}}(\lambda) = 1] - \Pr[\text{Game2}_{\mathcal{A}}(\lambda) = 1]| = |\Pr[d = d' | \text{com} \leftarrow \text{COM}.\text{Commit}(k)] - \Pr[d = d' | \text{com} \leftarrow \text{COM}.\text{Commit}(0)]| = |\Pr[b \neq b' | \text{com} \leftarrow \text{COM}.\text{Commit}(k)] - \Pr[b \neq b' | \text{com} \leftarrow \text{COM}.\text{Commit}(0)]| = |1 - \Pr[b \neq b' | \text{com} \leftarrow \text{COM}.\text{Commit}(k)] - \Pr[b = b' | \text{com} \leftarrow \text{COM}.\text{Commit}(0)]| = 2 \cdot \text{Adv}_{\text{COM}, \mathcal{B}}^{\text{Hid}}(\lambda) \leq \text{negl}(\lambda)$.

Lemma 11. *The undetectability of STG implies that the difference between the advantages of Game2 and Game3 is negligible.*

Proof. In the case $b = 1$, the two games are exactly identical. We argue that the output distributions of $\mathcal{A}_2$ of the two games are almost the same if $b = 0$.

Suppose there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that makes the difference between the advantages of the two games non-negligible in the case $b = 0$. Then, we can build another PPT adversary $\mathcal{B}$ that breaks the undetectability of STG.

First, $\mathcal{B}$ generates $(\text{crs}, \tau) \leftarrow \text{NIZK}.\text{Sim}_1(1^\lambda)$, $(\text{sk}_{\text{SIG}}, \text{pk}_{\text{SIG}}) \leftarrow \text{SIG}.\text{KeyGen}(1^\lambda)$, and $\text{com} \leftarrow \text{COM}.\text{Commit}(0)$ and gives $(\text{pp} = \text{crs}, \text{pk} = (\text{pk}_{\text{SIG}}, \text{com}))$ to $\mathcal{A}_1$.

Upon receiving queries from $\mathcal{A}_1$, $\mathcal{B}$ responds as follows: First, $\mathcal{B}$ queries x to the oracle of the undetectability game, receives y_1 , and repeatedly queries it until a response of an appropriate length is obtained. Then, it simulates a proof $\pi \leftarrow \text{NIZK}.\text{Sim}_2(\text{crs}, \tau, (\text{pk}_{\text{SIG}}, \text{com}, y))$ and returns (y, π) to $\mathcal{A}_1$. After $\mathcal{A}_1$ halts, $\mathcal{B}$ hands s from $\mathcal{A}_1$ to $\mathcal{A}_2$ and answers a query x from $\mathcal{A}_2$ by repeatedly using Model .

When $\mathcal{A}_2$ halts, $\mathcal{B}$ outputs whatever $\mathcal{A}_2$ outputs to the challenger. We can confirm that $\mathcal{B}$ simulates Game2 if the oracle responses are the output of Emb

and Game3 if they are computed by Model. Thus, we have $\Pr[\text{Game2}_{\mathcal{A}}(\lambda) = 1] - \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1] = \Pr[b = 1] \cdot \Pr[\text{Game2}_{\mathcal{A}}(\lambda) = 1|b = 1] + \Pr[b = 0] \cdot \Pr[\text{Game2}_{\mathcal{A}}(\lambda) = 1|b = 0] - (\Pr[b = 1] \cdot \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 1] + \Pr[b = 0] \cdot \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 0]) = \frac{1}{2} \cdot (\Pr[\text{Game2}_{\mathcal{A}}(\lambda) = 1|b = 0] - \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 0]) = \frac{1}{2} \cdot (\Pr[\mathcal{B}^{\text{Model}, \text{Emb}(\text{sk}, \cdot, \cdot)}(1^\lambda) = 1 | (\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)] - \Pr[\mathcal{B}^{\text{Model}, \overline{\text{Model}_n}}(1^\lambda) = 1]) = \frac{1}{2} \cdot \text{Adv}_{\text{STG}, \mathcal{B}}^{\text{Und}}(\lambda) \leq \text{negl}(\lambda).$

Lemma 12. *The undetectability of STG implies that the advantage of Game3 is negligible.*

Proof. Suppose there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that makes the difference between the advantages of the two games non-negligible. Then, we can build another PPT adversary $\mathcal{B}$ that breaks the undetectability of STG.

First, $\mathcal{B}$ honestly generates pp and pk according to the description of Game3 except that it does not create the key of STG on hand.

Then, $\mathcal{B}$ runs $\mathcal{A}_1$ by giving (pk, pp) . When responding to a query from $\mathcal{A}_1$, $\mathcal{B}$ generates blocks by making queries to the oracle of the undetectability game of STG and simulates a proof for the blocks. After $\mathcal{A}_1$ halts, $\mathcal{B}$ hands s from $\mathcal{A}_1$ to $\mathcal{A}_2$ and then replies to a query from $\mathcal{A}_2$ by making queries to the oracle. When $\mathcal{A}_2$ halts, $\mathcal{B}$ outputs whatever $\mathcal{A}_2$ outputs. We can confirm that $\mathcal{B}$ simulates the case where $b = 0$ if the oracle is $\overline{\text{Model}_n}(\cdot)$ and the case where $b = 1$ if the oracle is $\text{Emb}(\text{sk}, \cdot, \cdot)$. Thus, we have $\text{Adv}_{\mathcal{A}}(\lambda) = |\Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}| = |\Pr[b = 1] \cdot \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 1] + \Pr[b = 0] \cdot \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 0] - \frac{1}{2}| = \frac{1}{2} \cdot |\Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 1] + \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 0] - 1| = \frac{1}{2} \cdot |\Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 1|b = 1] - \Pr[\text{Game3}_{\mathcal{A}}(\lambda) = 0|b = 0]| = \frac{1}{2} \cdot |\Pr[\mathcal{B}^{\text{Model}, \text{Emb}(\text{sk}, \cdot, \cdot)}(1^\lambda) = 1 | (\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)] - \Pr[\mathcal{B}^{\text{Model}, \overline{\text{Model}_n}}(1^\lambda) = 1]| = \frac{1}{2} \cdot \text{Adv}_{\text{STG}, \mathcal{B}}^{\text{Und}}(\lambda) \leq \text{negl}(\lambda).$

It follows from the above lemmata that $\text{Adv}_{\mathcal{A}}(\lambda) \leq \sum_{i=0}^2 |\text{Adv}_{i, \mathcal{A}}(\lambda) - \text{Adv}_{i+1, \mathcal{A}}(\lambda)| + \text{Adv}_{\mathcal{A}}^{\text{ZK}}(\lambda) \leq \text{Adv}_{\text{NIZK}, \mathcal{B}_1}^{\text{ZK}}(\lambda) + \text{Adv}_{\text{COM}, \mathcal{B}_2}^{\text{Hid}}(\lambda) + \frac{3}{2} \cdot \text{Adv}_{\text{STG}, \mathcal{B}_3}^{\text{Und}}(\lambda) \leq \text{negl}(\lambda).$

Theorem 4 (Disclosure Soundness). *The disclosure soundness of Π is implied by the knowledge soundness of NIZK and the binding property of COM.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ that breaks the disclosure soundness game of Π . Assume the tuple $(\text{pk}, \zeta, \alpha)$ that $\mathcal{A}$ outputs satisfies the winning condition of the game. We consider two cases: the case where $x = (\text{pk}_{\text{SIG}}, \text{com}, \zeta) \notin L$ and the case where $x = (\text{pk}_{\text{SIG}}, \text{com}, \zeta) \in L$.

In the former case, (x, π) is a pair that breaks the soundness of NIZK since $\text{Ver}(\text{pp}, \text{pk}, \zeta, \alpha) = 1$ ensures NIZK. $\text{Ver}(\text{crs}, x, \pi) = 1$.

In the latter case, we can build an adversary $\mathcal{B}$ that breaks the binding property of COM. Let (k, r) be a pair in sk . While $\text{Ver}(\text{pp}, \text{pk}, \zeta, \beta) = 0$ means that one cannot extract a valid robust signature with k , $x \in L$ implies that there exists another witness $(\text{k}', \sigma', i', r')$. Thus, we have $\text{k} \neq \text{k}'$. Furthermore, we have $\text{com} = \text{COM}.\text{Commit}(\text{k}; r)$ based on the way (sk, pk) is generated and $\text{com} = \text{COM}.\text{Commit}(\text{k}'; r')$ by the assumption $x \in L$. Therefore, we can break the binding property by extracting (k', r') and outputting $(\text{com}, \text{k}, r, \text{k}', r')$.

Thus, we have that $\text{Adv}_{\Pi, \mathcal{A}}^{\text{DS}}(\lambda) \leq \text{Adv}_{\text{NIZK}, \mathcal{B}_1}^{\text{KS}}(\lambda) + \text{Adv}_{\text{COM}, \mathcal{B}_2}^{\text{Bnd}}(\lambda).$

References

1. Anthropic: How claude marks ai-generated content (2026), <https://support.claude.com/en/articles/16266773-how-claude-marks-ai-generated-content>
2. Boyle, E., LaVigne, R., Vaikuntanathan, V.: Adversarially robust property-preserving hash functions. pp. 16:1–16:20. <https://doi.org/10.4230/LIPIcs.ITCS.2019.16>
3. Chaum, D.: Private signature and proof systems. US Patent 5493614 (1996)
4. Chen, Y., Mao, X., Yi, H.: Advancing pseudorandom codes: Beyond parity checks and standard-model CCA1 security. Cryptology ePrint Archive, Paper 2026/1228 (2026), <https://eprint.iacr.org/2026/1228>
5. Christ, M., Gunn, S.: Pseudorandom error-correcting codes. pp. 325–347. https://doi.org/10.1007/978-3-031-68391-6_10
6. Christ, M., Gunn, S., Zamir, O.: Undetectable watermarks for language models. In: Agrawal, S., Roth, A. (eds.) Proceedings of Thirty Seventh Conference on Learning Theory. Proceedings of Machine Learning Research, vol. 247, pp. 1125–1139. PMLR (30 Jun–03 Jul 2024), <https://proceedings.mlr.press/v247/christ24a.html>
7. Döttling, N., Joux, A., Koppula, V., Rajasree, M., Waldner, H.: Chosen ciphertext secure pseudorandom codes in the standard model. In: Advances in Cryptology – CRYPTO 2026. pp. 282–313 (08 2026). https://doi.org/10.1007/978-3-032-35367-2_10
8. Döttling, N., Joux, A., Koppula, V., Rajasree, M.S., Waldner, H.: A simple compiler for CCA2-secure pseudorandom codes in the standard model. Cryptology ePrint Archive, Paper 2026/1867 (2026), <https://eprint.iacr.org/2026/1867>
9. European Parliament and Council of the European Union: Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union, L 2024/1689 (2024), <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>, article 50 (Transparency obligations for providers and deployers of certain AI systems)
10. Fairuze, J., Garg, S., Jha, S., Mahloujifar, S., Mahmoody, M., Wang, M.: Publicly-detectable watermarking for language models. IACR Communications in Cryptology **1**(4) (2025). <https://doi.org/10.62056/ahmpdkp10>
11. Google DeepMind: Watermarking ai-generated text and video with synthid (May 2024), <https://deepmind.google/blog/watermarking-ai-generated-text-and-video-with-synthid/>, accessed: 2026-08-13
12. Goyal, S., Kohli, P.: Identifying AI-generated images with SynthID, <https://deepmind.google/blog/identifying-ai-generated-images-with-synthid/>
13. Gunn, S., Zhao, X., Song, D.: An undetectable watermark for generative image models. In: The Thirteenth International Conference on Learning Representations (2025), <https://openreview.net/forum?id=jlhBFm7T2J>
14. Huang, Z., Zeng, G., Mu, X., Wang, Y., Yu, Y.: Multi-designated detector watermarking for language models (2024), <https://arxiv.org/abs/2409.17518>
15. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. pp. 143–154. https://doi.org/10.1007/3-540-68339-9_13
16. Lin, H., Shahabi, K., Song, M.J.: Unforgeable watermarks for language models via robust signatures. In: Heninger, N., Rosulek, M. (eds.) Advances in Cryptology – CRYPTO 2026. pp. 99–131. Springer Nature Switzerland, Cham (2026)