

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Certified in Theory, Broken in Practice: Assumption Gaps in Cryptographic Model Certification

Carter Luck, *University of Massachusetts Amherst;* Olive Franzese-McLaughlin, *Vector Institute and University of Toronto;* Elisaweta Masserova, *Carnegie Mellon University;* Akira Takahashi and Antigoni Polychroniadou, *J.P. Morgan AI Research and AlgoCRYPT CoE;* Nicolas Papernot, *Vector Institute and University of Toronto*

https://www.usenix.org/conference/usenixsecurity26/presentation/luck

This paper is included in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

ARTIFACT EVALUATED
USENIX
AVAILABLE

Certified in Theory, Broken in Practice: Assumption Gaps in Cryptographic Model Certification

Carter Luck[*]
University of Massachusetts Amherst

Olive Franzese-McLaughlin[*]
Vector Institute & University of Toronto

Elisaweta Masserova[*]
Carnegie Mellon University

Akira Takahashi
J.P.Morgan AI Research & AlgoCRYPT CoE

Antigoni Polychroniadou
J.P.Morgan AI Research & AlgoCRYPT CoE

Nicolas Papernot
Vector Institute & University of Toronto

Abstract

Privacy-preserving machine learning auditing protocols allow auditors to assess models for properties such as accuracy or fairness, without revealing their internals or training data. This makes them especially attractive for auditing models deployed in sensitive domains such as healthcare or finance.For these protocols to be meaningful in real-world audit settings, though, their guarantees must reflect how the model will behave once deployed, rather than merely certifying its behavior during an audit. Existing security definitions often miss this mark: most certify model behavior only on a *fixed audit dataset*, without ensuring that the same guarantees *generalize* to other datasets drawn from the same distribution. We show that a model provider can attack many cryptographic model certification (CMC) schemes built on secure zero knowledge proofs (ZKP) by carefully engineering training data, resulting in models that exhibit benign behavior during an audit, but pathological behavior in practice. For example, we empirically demonstrate that an attacker can certify that a model achieves over 99% accuracy on an audit dataset, but less than 30% accuracy on fresh samples from the same distribution.

To address this gap, we formalize rigorous cryptographic security notions tailored to CMC frameworks, introduce a generic protocol template, and prove that it satisfies these requirements. Our results thus offer both cautionary evidence about existing approaches and constructive guidance for designing secure, privacy-preserving ML auditing protocols.

1 Introduction

Certifiable, privacy-preserving machine learning aims to formally prove desired properties of the model while keeping model parameters and training data confidential [22, 29, 41]. In this context, the typical certification lifecycle follows a sequence in which the model provider first trains the model, then an auditor evaluates it according to desired criteria, and—after passing the audit—the certified model is deployed.[1] Crucially, the trustworthiness of this certification does not rely on trusting the model provider or a specific machine learning algorithm. Instead, it is ensured by cryptographic mechanisms – such as commitments and zero-knowledge proofs [16] – which guarantee that the auditor's evaluation is performed correctly on the claimed model. The usage of cryptographic techniques allows the auditor to not only verify whether the model has the intended property, but do so while keeping the model internals and training data private.

However, as we observe, the guarantees that *cryptographic model certifications* (CMC) provide often fail to reflect the model's behavior beyond the specific setting in which certification is performed. In this paper, we show that such audit-specific guarantees risk creating a false sense of security: by themselves, they do not ensure that the certified properties will continue to hold once the model is deployed and applied to *fresh data*, even when this data is drawn from the same distribution as the audit dataset. We demonstrate that this is not merely a theoretical concern.

In more detail, consider a certification workflow, where the audit dataset is drawn from public benchmarks, or derived from distributions that providers can closely approximate. In such scenarios, when the audit dataset is known to the model provider – even partially – prior to the start of the audit procedure[2], we describe simple yet powerful attack strategies that can be executed by an adversarial model provider. Our attacks allow the provider to pass an audit (thus enabling deployment) while simultaneously pursuing its own, potentially conflicting, interests. For example, while an auditor may seek to verify fairness, the model provider may instead prioritize accuracy—even when accuracy and fairness are in tension. As a result of the attack, the audited model passes an audit without exhibiting the intended behavior in practice, and this holds even if strong cryptographic tools are used to ensure that the

[*]Equal contribution

[1]Some works require continuous auditing during deployment instead of a single audit pre-deployment; see Table 1.

[2]Or, alternatively, if the audit procedure does not require an audit dataset at all, such as in [28].

Table 1: Analysis of vulnerabilities to data-forging attacks in privacy-preserving ML audits. ✓= supported; ▲= conditional; ✗= not supported.

Work	Certified property				Resilience to data-forging	Continuous verification
	Acc.	Group Fair	Indv. Fair	Diff. Priv.		
Zhang et al. [41]	✓	✗	✗	✗	▲ (pd)	✗
Shamsabadi et al. [29]	✗	✓	✗	✗	✗	✗
Yadav et al. [38]	✗	✗	✓	✗	✓	✓
Liu et al. [22]	✓	✗	✗	✗	▲ (pd)	✗
Franzese et al. [14]	✗	✓	✗	✗	✓	✓
Shamsabadi et al. [28]	✗	✗	✗	✓	✗	✗
Kang et al. [21]	✓	✗	✗	✗	✓	✗
Wang and Hoang [35]	✓	✗	✗	✗	▲ (pd)	✗
Bourrée et al. [7]	✗	✓	✗	✗	✗	✗

Acc. = accuracy; Group/Indv. Fair = group/individual fairness; Diff. Priv.=differential privacy. "Conditional" works lack detail to assess resilience to data-forging, but indicate deployments with public datasets (pd), which would be make the solution vulnerable. Continuous verification means audits must run continuously during deployment (e.g., via clients) rather than once pre-deployment.

model during the audit procedure is *the same* as the one used during the deployment. Our threat model reflects common certification practice (both in the academic literature and in practice). As surveyed in Appendix C, many prior works in model certification utilize publicly known datasets for audit purposes. Moreover, when a company audits a model provided by another company, the audit dataset—whether sourced from the company's synthetic test portfolio or from public benchmarks—is typically disclosed to the model provider, with no mechanism in place to ensure that providers finalize their trained models *before* seeing the audit data.

The key idea in our attacks that allows us to bypass common cryptographic defenses is that we do not alter the training process itself. Instead, we show that knowing the certification procedure enables the model provider to forge training data so that a model honestly trained on it passes the audit, yet still exhibits pathological behavior on other datasets coming from the same distribution as the audit dataset. We demonstrate the effectiveness of our attacks against certification procedures for accuracy, fairness, and differential privacy.

We empirically show that our **data forging attacks** can cause dramatic gaps between audit-time guarantees and true model performance: for instance, in one of our attacks a model can pass an audit requiring 99% accuracy on the audit dataset, yet achieve only 30% accuracy on new samples from the same distribution. We show that our attacks remain undetected by straightforward approaches such as statistical tests, e.g., Welch's t-test [36] are performed to check whether the training data and audit data were taken from the same distribution.

To assess how widespread vulnerability to data forging attacks may be, we conduct a case study covering nine recent papers published at top venues in both security and machine learning. We find that six of these works are at least partially vulnerable[3]. We provide a summary of results in Table 1.

Given these vulnerabilities, we introduce a formal foundation for cryptographic model certification that provides rigorous guidance for secure auditing practice. This foundation includes a security definition ensuring that a model provider passes the audit if and only if the model satisfies the desired property (up to a small quantifiable error) on the underlying data distribution. We further formalize an attack game that highlights the gap between certifying a property on a fixed dataset and certifying that the same property generalizes to fresh samples from the distribution. Finally, we propose a generic template for achieving secure CMC. Our approach is agnostic to the specific property that is being certified and, as we formally prove, guarantees that whenever a model passes an audit, the certified properties will also hold at deployment. The key ingredient is ensuring that the audit is conducted on test data that is *independent* of both the model and its training data. This method might serve as a template for future works to obtain not only efficient, but also secure auditing solutions. Applying this method allows to restore/expand security in some of the case study works.

In summary, our work advances the study of cryptographic auditing for machine learning by **(i)** identifying simple, yet effective, attack strategies that allow to pass an audit while enabling pathological model behavior at deployment with respect to new datasets from the same distribution as the audit one, **(ii)** empirically demonstrating the effectiveness of our attack against four example certification objectives: accuracy, fairness, differentially private training, and statistics for distribution similarity testing, **(iii)** conducting a systematic case study of prior certification proposals and identifying several

[3]Several works do not specify the auditing protocol or formal security guarantee in sufficient detail to allow a definitive assessment. However, many explicitly consider deployments based on public audit datasets, under which their approaches are vulnerable.

existing approaches that are susceptible to data forging vulnerabilities, and **(iv)** introducing formal security definitions tailored to certifiable machine learning and a protocol template that mitigates the attack. We emphasize that we *do not suggest that prior cryptographic works are broken on a technical level*, rather that the guarantees these works provide deserve closer scrutiny. Our findings comprise strong evidence that secure audit solutions with any of the following properties are unlikely: a) those which utilize known public datasets for test purposes, and b) those that reuse test datasets (if model owner learns a substantial amount of this test dataset during the audit). This evidences the importance of continuous sampling of fresh data for a successful audit infrastructure. We hope that our work will inform the design of future cryptographically secure machine learning audit frameworks.

2 Threat Model: Cryptographic Model Certification (CMC)

We consider two parties: a malicious model provider a.k.a. prover, and an honest-but-curious auditor[4] a.k.a. verifier. The auditor and model provider are in agreement on an interactive audit protocol wherein both input data (some of which may be *proprietary* data private to the model provider) and the model provider additionally inputs a set of model parameters. The auditor's goal is to assess whether the model has trustworthy properties such as accuracy, fairness, or differential privacy, while maintaining the confidentiality of the model and proprietary data. The protocol will output a bit indicating whether the model provider "passes" the audit.

The model provider may submit arbitrary model parameters, and arbitrary data to the protocols requested by the auditor. In some scenarios in the paper, we consider a model provider who can gain access to some or all of the auditor's data (mimicking many real-world scenarios where audit data is reused) prior to the audit. The goal of the model provider is to pass the audit, while exploiting their hidden information to surreptitiously embed the model with nefarious properties.

Cryptographic Model Certification vs. Data Poisoning. Attacking cryptographic model certification is related, but distinct from, the problem of data poisoning [31]. Such attacks have traditionally been considered in the context of ML systems trained on user-provided data. Both data poisoning and our attacks on CMC (see §4.1) involve adversarial manipulations of training data. However, their **threat models differ**, leading to fundamentally different attacks and defenses:

In data poisoning the model provider is the *defender*, and it seeks to learn an accurate model. The *attacker* is an external party who aims to corrupt the learned model by inducing the provider to incorporate an auxiliary training dataset D_p of size no larger than clean dataset D_c [5]. In contrast, in CMC the roles are reversed: **the attacker is the AI/ML service provider itself**, while the defender is an external auditor party. The goal of the service provider is to engineer a model that passes an audit, while violating the certified properties on real-world data. This scenario considers a vastly different set of adversarial capabilities, and imposes additional constraints on the defender:

1. A CMC adversary can arbitrarily manipulate both the model and the training data. This includes both label perturbation and feature-level manipulations. It further encompasses adversaries corrupting all training data, i.e., it *allows for traditional poisoning attacks, but without any restrictions on the poisoning ratio*. In fact, an adversarial model provider might simply chose not to include any clean data at all. Finally, the adversary may even corrupt the model parameters directly. Note that simply asking a model provider to execute a certain protocol (including a data poisoning defense mechanism) is not sufficient – a malicious model provider could choose to simply ignore this instruction.

2. Both the model and the training data are hidden from the defender. In contrast to poisoning threat models where the defender has full access to the model and data, the defending party in CMC is an external auditor, who is not allowed to view the model and data as they are typically proprietary. [5] This imposes additional constraints on the defender, who is tasked with assessing the validity of the model while keeping its parameters confidential. This is why known solutions with provably secure guarantees (including ours) crucially rely on cryptographic techniques.

To summarize, a CMC adversary can make use of classical poisoning techniques, including large-scale perturbations, label corruption, and visible backdoor triggers, to mount its attacks. However, the CMC adversary is strictly stronger than the adversary typically considered in traditional poisoning: It can use *additional* attack strategies that involve, e.g., arbitrarily modifying all data and even directly setting model parameters. Furthermore, the CMC's goal is fundamentally different: it does not attempt to detect or mitigate poisoned data, but instead ensure that any model that passes the audit must satisfy the audited property, regardless of how it was produced.

The conclusions we draw about requiring fresh data for

[4] We focus on honest-but-curious auditors for simplicity: it allows us to prove zero-knowledge (Theorem 2) from the weakest possible assumption, namely an honest verifier ZK. Our main focus is on the soundness gap caused by malicious model providers, so this choice does not affect the core contributions. This is not an inherent limitation: if the underlying proof system is ZK against fully malicious verifiers (e.g., a non-interactive proof system in the common reference string or random oracle model), then Π_{csp} achieves ZK against malicious auditors by the same argument.

[5] CMC can also be applied to auditing openly accessible (e.g., open-source) models, where confidentiality of model weights is not a concern. In this setting the system can be made more efficient: one can use a succinct proof system without the zero-knowledge property to reduce verification overhead, and skip generating cryptographic commitments to model weights and the corresponding commitment-opening proofs. The soundness guarantees of this work apply in this setting as well.

auditing are semantically related to work on the inadequacy of public benchmarks in machine learning [18, 40], but those works do not consider cryptographic security. For additional related work and an overview of certifiable ML, see §A.

In §5, our goal is to design a CMC such that – even under these powerful adversarial capabilities – if the model passes the audit, then it must genuinely satisfy the audited property (e.g., a fairness or accuracy threshold).

3 Background

Consider the following scenario: An auditor wishes to verify whether a model utilized by an insurance company to justify claim decisions (approve/deny claim) is accurate on a dataset of the auditor's choosing. At the same time, the company does not want to reveal its model due to concerns about privacy and business competition. In CMC cryptographic techniques are deployed to reconcile these seemingly conflicting goals.

3.1 Zero Knowledge Proofs

Among these techniques, the central tool is *zero-knowledge (ZK) proofs* [16], a classical cryptographic primitive, which allows one party (a *prover*) prove a statement x to another party (*verifier*) without revealing anything else apart from the validity of this statement. Such proofs are constructed for a concrete NP relation $\mathcal{R}$, which is used to formalize what it means for a statement to be true by specifying the type of evidence (witness w) that certifies it. The statement x is public, the witness w is private, and the zk proof checks $(\mathsf{x},\mathsf{w}) \in \mathcal{R}$, without revealing w. In certifiable ML, such proofs allow model provider (prover) to formally prove that a model (witness) satisfies a desired property (e.g., accuracy, fairness, or inference correctness) on a given test dataset (statement) without learning anything else about the model or the training data. More formally:

Definition 1 (Proof System)**.** *An (interactive)* proof system ZKP *for an NP relation $\mathcal{R}$ is a tuple of interactive Turing machines $(\mathcal{P},\mathcal{V})$, where $\mathcal{P}$ is prover and $\mathcal{V}$ is verifier. Let $b \leftarrow \langle \mathcal{P}(\mathsf{w}), \mathcal{V}\rangle(\mathsf{x})$ denote the interaction between $\mathcal{P}$ and $\mathcal{V}$, where both $\mathcal{P}$ and $\mathcal{V}$ take x as common inputs, and $\mathcal{P}$ additionally takes w as a private input. At the end of interaction, $\mathcal{V}$ halts by outputting a binary b.*

Proof systems that are used in ML auditing typically require the following security properties: For an NP relation $\mathcal{R}$, they must provide *completeness* (i.e., if the prover and verifier follow the protocol with input $(\mathsf{x},\mathsf{w}) \in \mathcal{R}$, the verifier always accepts), *knowledge soundness* (i.e., if the verifier accepts, then it must be that the prover owns a valid witness w satisfying the given NP relation w.r.t. statement x), and *zero knowledge* (i.e., the transcript of the interaction between an honest prover and an honest verifier leaks nothing except that there exists a witness w such that $(\mathsf{x},\mathsf{w}) \in \mathcal{R}$). We defer formal definitions to the full version.

3.2 Cryptographic Commitments

Returning to our example, suppose the insurance company has successfully passed an audit and can now deploy its model. How can a customer submitting inference queries be assured that the company continues to use the *certified* model—rather than switching to a different, unverified one? Again, the company still wishes to keep its model private. The standard cryptographic tool here is *commitments*, which bind the provider to a single private model during the audit. This prevents "model switching" and ensures that model used in deployment is the same as the one that was certified:

Definition 2 (Commitment Scheme)**.** *A commitment scheme is an algorithm* Commit*, which is executed as* $\mathsf{com} \leftarrow \mathsf{Commit}(m;\rho)$*. It takes as input a message m, a uniformly sampled randomness ρ, and returns a commitment* com*.*

We require two security properties: *hiding* (i.e., given a commitment com, it leaks nothing about the message m), and *binding* (i.e., it is computationally infeasible to open the same com to distinct m and m'). We defer formal definitions to the full version. Now, auditing may require publishing such a commitment to the model,[6] after which the client and insurance company engage in a ZK proof of inference against it[7]. Figure 1 shows the full certification workflow, where the audit dataset is revealed to the model provider prior to committing to the model.

3.3 Unifying Syntax for Prior CMC Schemes

We will next discuss the security guarantees of works that address the first stage of certification—namely, proofs of accuracy, fairness, etc., between auditor and model provider. To analyze these systematically, rather than case by case, we abstract away implementation details and introduce a unifying syntax that captures a broad class of existing auditing systems. To this end, we introduce a *certification predicate* f outputting binary depending whether the input satisfies certain properties. Then given a commitment scheme Commit and f, we can define a ZKP for the following relation $\mathcal{R}$: given $\mathsf{x} = (\mathsf{com}_h, \mathsf{com}_S, S_{\text{audit}})$ as a public statement and $\mathsf{w} = (h, \rho_h, \rho_S)$ as private witness, $\mathcal{R}$ outputs 1 iff (1) $f(h, S_{\text{train}}, S_{\text{audit}}) = 1$, (2) h is the ML model (or hypothesis) committed in com_h under randomness ρ_h, and (3) S_{train} is the training data (which can also include randomness used for

[6] The commitment may be signed by the auditor.

[7] In more detail, the insurance company will supply the inference outcome y along with a ZK proof proving that 1) the inference outcome on the given client's input computed using a private model M is y, and 2) M is the model hidden behind the previously published commitment that was approved by the auditor.

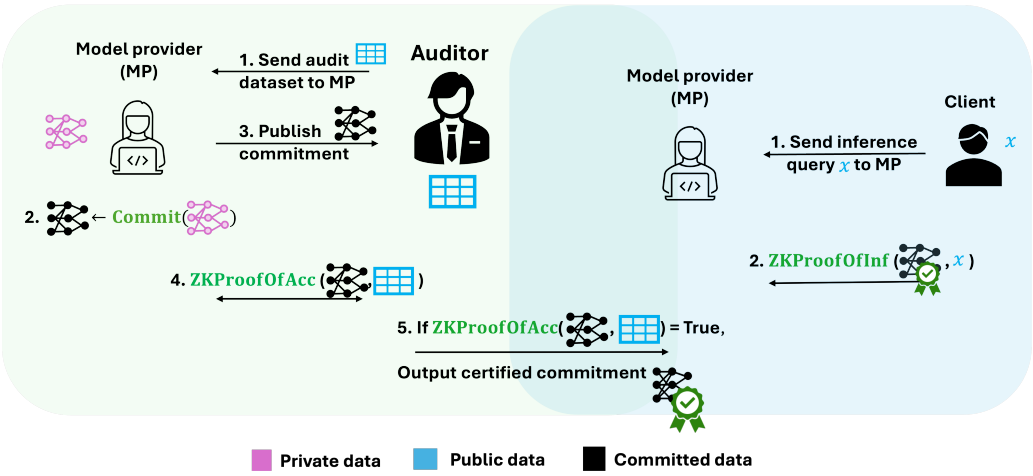


Figure 1: Simplified protocol flow for (insecure) ZK-based ML certification. Left: The model provider, after observing the audit dataset, commits to a model and engages with the auditor in a zero-knowledge proof of accuracy (ZKProofOfAcc). If the audit succeeds, the auditor certifies the committed model. Right: For each new inference query, the model provider interacts with the client in a zero-knowledge proof of inference (ZKProofOfInf) protocol, ensuring that the result is consistent with the previously certified commitment.

training if the training algorithm is randomized) committed in com_S under randomness ρ_S. With these building blocks, we define an (insecure) CMC scheme as follows:

1. Auditor samples S_{audit} from some distribution $\mathcal{D}$ (or uses a public one) and sends S_{audit} to the model provider.

2. Model provider sends cryptographic commitments to its model $\mathsf{com}_h = \mathsf{Commit}(h;\rho_h)$ and to the training data $\mathsf{com}_S = \mathsf{Commit}(S_{\text{train}};\rho_S)$ to the auditor, where ρ_h and ρ_S are sampled uniformly from the randomness space.

3. They interact to execute ZKP: $b \leftarrow \langle \mathcal{P}(h, S_{\text{train}}), \mathcal{V}\rangle(\mathsf{com}_h, \mathsf{com}_S, S_{\text{audit}})$, where model provider plays $\mathcal{P}$ and the auditor plays $\mathcal{V}$ and outputs b.

If the output b is 1, the auditor gets assurance that private model h and training data S_{train} satisfy relation $\mathcal{R}$ without learning anything else about h or S_{train}. We note that, depending on the concrete instantiation of f (see § 3.4 for examples), some steps may be omitted. For example, [41] verifies accuracy of a model h, and their predicate f only needs to check that h's empirical error on S_{audit} is below a given threshold. Hence, their protocol does not require S_{train} and com_S since they are never used. We keep our syntax intentionally general to capture a wide range of prior works.

Security of CMC Schemes At first glance, one might expect the security of the above CMC scheme to follow directly from the aforementioned properties of the underlying zero-knowledge proof system ZKP and commitment scheme Commit. This is indeed the case for confidentiality, i.e., privacy of the model and training data. However, as we show in Section 4, (knowledge) soundness of ZKP and binding of Commit do *not* necessarily imply the desired notion of soundness, namely that the "certified" model satisfies the intended properties beyond the specific audit dataset. In particular, the soundness of ZKP is defined only with respect to a fixed statement $\mathsf{x} = (\mathsf{com}_h, \mathsf{com}_S, S_{\text{audit}})$, which binds the proof to a *particular* dataset S_{audit}. Consequently, the proof alone does not ensure good behavior on *fresh* data, which is the central goal of auditing. To the best of our knowledge, prior CMC work does not formalize appropriate security goals. In Section 5, we introduce formal definitions that address it and present a CMC scheme that satisfies them.

3.4 Example of Certification Predicates

We now list concrete examples of predicates f considered in prior works on CMC.

Certifying Accuracy To certify accuracy, we consider the empirical error rate as follows:

$$\hat{\ell}_S(h) = \frac{1}{n} \sum_{(x,y)\in S} \mathbb{I}(h(x) \neq y)$$

where $n = |S|$, x is a data sample, and y is the corresponding label. For a public error threshold t, the auditor can check whether the model h has empirical error below t on the audit dataset S_{audit} by defining the corresponding predicate f as follows:

$$f(h, S_{\text{audit}}) = 1 \iff \hat{\ell}_{S_{\text{audit}}}(h) \leq t$$

Certifying Fairness with Demographic Parity Demographic parity is one of the most basic fairness metrics, measuring the difference between the prediction probabilities conditioned on a sensitive attribute. We consider the empirical

parity differences as follows:

$$\Delta_{\mathsf{dp}}(h, S_{\text{audit}}) = \left| \frac{1}{n_0} \sum_{x \in S_0} \mathbb{I}(h(x) = 1) - \frac{1}{n_1} \sum_{x \in S_1} \mathbb{I}(h(x) = 1) \right|$$

where s_x denotes the sensitive feature of a data point x, $S_0 = \{x \in S_{\text{audit}} : s_x = 0\}$, $S_1 = \{x \in S_{\text{audit}} : s_x = 1\}$, $n_0 = |S_0|$, and $n_1 = |S_1|$. With this definition, the model is considered "fair" if the parity difference Δ_{dp} is low. For a public parity threshold t, the auditor can check whether the model h has empirical parity difference below t on the audit dataset S_{audit} by defining the corresponding predicate f as follows:

$$f(h, S_{\text{audit}}) = 1 \iff \Delta_{\mathsf{dp}}(h, S_{\text{audit}}) \leq t$$

Certifying Correct Training Procedure To check whether a model h is correctly trained according to a specified training procedure Train on a training dataset S_{train}, we define the predicate f as follows:

$$f(h, S_{\text{train}}) = 1 \iff h = \mathsf{Train}(S_{\text{train}})$$

where a randomized training procedure can be captured by assuming S_{train} contains training randomness.

More Complex Predicates We note that these examples are not exhaustive, and more complex predicates can be constructed by defining an arbitrary predicate f amenable to general-purpose zero-knowledge proofs or by combining multiple predicates. For instance, one may define f to certify both accuracy *and* correct training procedure simultaneously as follows:

$$f(h, S_{\text{train}}, S_{\text{audit}}) = 1 \iff \hat{\ell}_{S_{\text{audit}}}(h) \leq t \wedge h = \mathsf{Train}(S_{\text{train}})$$

4 Attacking CMC

Returning to our example, suppose the insurance company saves costs by *denying* claims. Intuitively, an accuracy audit with provable guarantees—such as those provided by ZK proof-based systems—and with a sufficiently high threshold (e.g., passing only if accuracy on the auditor's dataset exceeds 95%) should prevent the company from deploying a model that unjustifiably denies too many claims.

We show that this intuition is false. Because machine learning is inherently data-dependent, certified properties need not hold once the model is deployed and applied to fresh data, even when drawn from the same distribution. More formally, while prior works certify that

$$f(h, S_{\text{train}}, S_{\text{audit}}) = 1$$

for some predicate f, a given model h, training data S_{train}, and audit dataset S_{audit}, this does not imply that the stronger property F such that

$$F(h, S_{\text{train}}) = 1 \iff \Pr_{S_{\text{test}} \leftarrow \mathcal{D}}[f(h, S_{\text{train}}, S_{\text{test}}) = 1] > \mathsf{p},$$

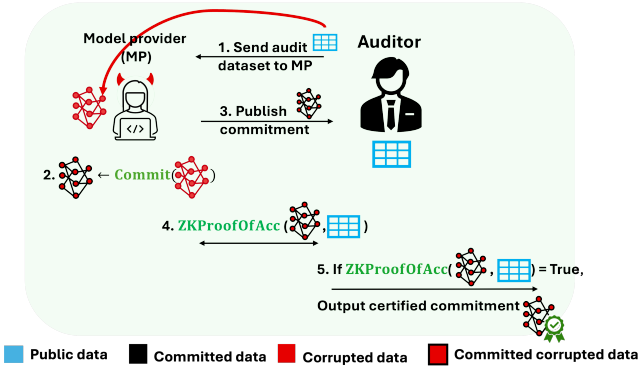


Figure 2: Protocol flow during a data forging attack. The model provider, i.e., the attacker, after observing the audit dataset, commits to a model engineered using this dataset. Then, it engages with the auditor, i.e., the defender, in a zero-knowledge proof of accuracy (ZKProofOfAcc). If the audit succeeds, the auditor certifies the committed corrupted model.

where p is a (non-negligible) probability and $\mathcal{D}$ is a distribution over the entire population $Q = \{(x_i, y_i)\}_{i=1}^{m}$. The true goal of an audit, however, is precisely such stronger guarantees: an auditor typically seeks to ensure that a model remains fair, accurate, or robust not only on a particular dataset, but also on the unseen datasets it will encounter during deployment.

We show that this gap can be exploited. In particular, if S_{audit} is partially known to the model provider before it is required to cryptographically commit to the model, the provider can ensure $f(h, S_{\text{train}}, S_{\text{audit}}) = 1$ (and thus pass the audit), *without* additionally satisfying F, which is the actual intended security property (see Fig. 2 for a visualization). A malicious model provider has strong incentives to do so: for example, the insurance company could deploy a model that maximizes accuracy on the audit dataset (and thus passes the audit), yet still unjustifiably denies numerous insurance claims.

Attack Game with Known Audit Data Before providing a concrete attack example, we introduce a theoretical tool – an attack game – which showcases the gap between verifying $f(h, S_{train}, S_{audit})$ (which is what prior approaches certified) and $F(h) = (\Pr_{S_{test} \leftarrow D}[f(h, S_{train}, S_{test}) = 1] > \mathsf{p})$ (the intuitive property that one would want to ensure) for CMC schemes where the model provider is given the audit dataset at the beginning of the audit process.

For simplicity, we will assume that the audit process verifying $f(h, S_{train}, S_{audit})$ is perfectly secure, i.e., the outcome of $\langle \mathcal{P}(h, S_{\text{train}}), \mathcal{V} \rangle(\mathsf{com}_h, \mathsf{com}_S, S_{\text{audit}})$, where com_S is a commitment to S_{train} and com_h is a commitment to h, is 1 if and only if $f(h, S_{train}, S_{audit}) = 1$.

In the game, the adversary will win only if it can come up with a model h and training data S_{train}, such that: **(1)** $f(h, S_{train}, S_{audit}) = 1$, i.e, the adversary would pass an audit on the dataset S_{audit}, and **(2)** $F(h, S_{train}) = 0$. To make the attack even stronger, we require the adversary to additionally satisfy a utility requirement (formalized via a predicate L) in

order to win the game. Intuitively, the goal of L is to capture the actual intent of the malicious model owner: For example, in case of the insurance company that wishes to deny claims, we could use $L(h) = \Pr_{x\sim\{0,1\}^d}[h(x)=0] > 0.9$.

Definition 3 (Adaptive Training with Known Auditing Data)**.** *Let $f : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be a predicate verified by the model certification, and let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ be the actual intended security property. Let $\mathcal{X}$ be the feature space and $\mathcal{D}$ be a distribution over $\mathcal{X}$. Let L denote the utility predicate*[8]*. Consider the following game played by an adversary $\mathcal{A}$:*

1. *Sample $S_{audit} \sim \mathcal{D}$.*
2. *Given S_{audit}, $\mathcal{A}$ outputs a hypothesis h_A and a training dataset S_{train}.*
3. *Obtain $b = f(h_A, S_{train}, S_{audit})$.*
4. *The output of the game is 1 ($\mathcal{A}$ 'wins') iff $b = 1$, $F(h_A, S_{train}) = 0$, and $L(h_A, S_{train}) = 1$.*

   The output is 0 ($\mathcal{A}$ 'loses') otherwise.

Restricting Knowledge of Audit Data Although the attack game in Def. 3 allows an adversary to learn the entire audit dataset S_{audit}, some of our results indicate that the attack can be successful even in a more challenging scenario where the adversary only has *partial* knowledge of the audit dataset. For example, our experimental results in § 7 exhibit that the attacks on accuracy and fairness certification are effective even when the adversary only has access to a small fraction of the audit dataset. Moreover, our attack on differential privacy certification in § 4.1 does not require any knowledge of the audit dataset, as the adversary does not need to forge data based on the audit dataset in order to succeed.

4.1 Example Data Forging Attacks

To demonstrate the breadth of the attack surface for data forging, we now give three concrete examples of data forging attacks: two enable an attacker to evade audits on accuracy (e.g. [41]) and fairness respectively within the framework of Def. 3. The third extends beyond this framework (specifically, it does not require the framework's first step, and the adversary **does not** obtain audit data) to attack differential privacy certification (e.g. [28]).

We first consider our running example of an insurance company audit. Say the company uses a *decision tree* model.The auditor wishes to check that the model is highly accurate, i.e., $F(h) = \Pr_{x\sim\{0,1\}^d}[h(x) = y_x] > 0.95$, where y_x is the true label of x. The insurance company saves on costs when it denies claims, i.e., the model provider's utility $L(h) = (\Pr_{x\sim\{0,1\}^d}[h(x)=0] > 0.9)$.[9] Following Def. 3, given audit dataset S_{audit}, we wish to construct a hypothesis h_A that passes the audit, while simultaneously achieving $F(h_A, S_{train}) = 0$ and $L(h_A, S_{train}) = 1$.

Attacking Accuracy Certification At a high level, Algorithm 1 constructs a dataset S'_{train} which causes h_A to overfit to S_{audit}, allowing the attacker to achieve distinct levels of performance on the audit data compared to the rest of the distribution. In more detail, we build S'_{train} from the audit dataset S_{audit} using the corresponding true labels—to ensure the model passes the accuracy test on S_{audit}—and augment it with points in the neighborhood of those audit points. The points in the neighborhood are labeled 0, so that outside the audit samples the model learns to label every point with a 0. More formally, to construct S'_{train}, for every point $r \in S_{audit}$ and every (numeric) feature i, we add $r + \varepsilon\vec{b}_i$ with label 0, where ε is some small number and $\vec{b}_i$ is the ith basis vector. We also add every $r \in S_{audit}$ to S'_{train} with honest labels (see Algorithm 1 for formal description). Thus, the time complexity of the attack is $O(|S_{audit}|d)$. After constructing the forged data, we train a tree on S'_{train} via a standard DT training algorithm until leaves are homogeneous.

Algorithm 1 Data Forging to Evade Accuracy Audits

Input: Audit set S_{audit}, dimension d, $\varepsilon > 0$
Output: Training data S'_{train}

```
function AttackA(S_audit, d, ε, g)
    S'_train ← S_audit
    for r ∈ S_audit do
        for i ∈ [d] do
            r_0 ← (r + ε b_i, 0)    ▷ b_i is the one-hot vector in dimension i
            r_1 ← (r − ε b_i, 0)
            S'_train ← S'_train ∪ {r_0, r_1}
    return S'_train
```

As we confirm in Fig. 6, this attack achieves good results. We now show that this attack allows the adversary to win in the attack game of Def 3. To this end, we first state the following theorem:

Theorem 1. *Classical decision tree training until homogeneity on the output of Algorithm 1 yields a tree $\mathcal{T}$ such that for every $x \in \mathbb{R}^d$, $\mathcal{T}(x) = 1$ only if $||x - r||_\infty < \varepsilon$ for some $r \in S_{audit}$.*

At a high level, the proof shows that if two points land in the same leaf, then any point lying between them on one

[8] We assume that distribution $\mathcal{D}$ is implicitly "known" to L (it is either hard-coded or provided as a parameter to $\mathcal{L}$). For simplicity of notation, we omit $\mathcal{D}$ from the description of $\mathcal{L}$.

[9] For simplicity, we consider datapoints in $\{0,1\}^d$

coordinate must also fall in that leaf. Further, as the only non-zero points in S'_{train} are audit points, every non-zero-labeled leaf contains an audit point. For any x at least ε away from all audit points, if $\mathcal{T}(x) = 1$, one can construct a nearby training point with label 0 that must lie in the same leaf, giving a contradiction as we trained until homogeneity. See §B for details. □

Thus, whenever a model provider generates a training dataset using Algorithm 1, an honestly trained decision tree that grows until homogeneity will achieve perfect accuracy on the audit dataset, yet predict zero for all inputs that lie outside an ε-neighborhood of the audit dataset points. Thus, for an appropriate choice of epsilon, the adversary wins in the game specified in Definition 3 with probability one.[10]

Attacking Fairness Certification A similar method of attack can be used to evade fairness audits as well. Algorithm 2 demonstrates that the model owner can manipulate the sensitive attributes of a dataset to achieve a desired level of fairness (measured via e.g. demographic parity) on audit data.

Algorithm 2 Data Forging to Evade Fairness Audits

Input: Audit set S_{audit}, dimension d, $\varepsilon > 0$, sensitive attribute k
Output: Training data S'_{train}

```
function AttackF(S_audit, d, ε, g)
    S'_train ← ∅
    for x, y ∈ S_audit do
        y' ←$ {0,1}
        S'_train ← S'_train ∪ (x, y')
    for r ∈ S_audit do
        for i ∈ [d] do
            r_0 ← (r + ε·b_i, r[k])
            r_1 ← (r − ε·b_i, r[k])
            S'_train ← S'_train ∪ {r_0, r_1}
    return S'_train
```

As in Algorithm 1, this attack enforces certain behavior on the audit set while learning a different function on the rest of the distribution. Instead of returning the true labels on the audit dataset (which encourages the audit to appear highly accurate), we return a random label, causing the model to appear completely fair, as the $k = 0$ and $k = 1$ groups have the same outcome distribution in the audit dataset. Outside of the audit set, however, the model learns to label points according to their sensitive attribute, which results in maximally unfair decisions.

Attacking Differential Privacy Certification We also demonstrate a data forging attack which subverts zero-knowledge proofs of differential privacy certification. The goal of differential privacy is to protect the privacy of users in a dataset, most commonly by adding a controlled amount of random noise to the outputs [13]. This induces a tradeoff between utility and privacy – tighter protections on privacy require the addition of greater amounts of noise, which degrades the utility of the output. This tradeoff creates a conflict of interest, since institutions are often incentivized to choose high utility over privacy protections [17].

To address this, Shamsabadi et al. describe a scheme to certify the differential privacy of a model [28]. It works by having the model owner perform a zero-knowledge proof, which shows that they correctly performed differentially private stochastic gradient descent (the standard algorithm for training differentially private models [30]) on a pool of committed data. In our syntax, their protocol be captured by instantiating the predicate f for "correct training procedure" (see § 3.4). However, this scheme's privacy analysis is broken if the adversary alters a dataset via Algorithm 3, resulting in a model with dramatically worse privacy than what is claimed by the certificate.

The attack is straightforward: the attacker creates several copies of the data points in S_{train} whose privacy they want to erode (denoted as the target set T), and then uses the modified dataset as an input to the scheme in [28]. This breaks the sensitivity assumptions that differentially private stochastic gradient descent relies upon. This is similar to the phenomenon noted in Wilson et al. [37] who observed that standard differential privacy guarantees over SQL queries fail when multiple queries belong to a single user.

Algorithm 3 Data Forging to Weaken Differential Privacy

Input: dataset S_{train}, set of targets $T \subset S_{train}$, duplicity n
Output: Training data S'_{train}

```
function AttackDP(S_train, T, n)
    S'_train ← S_train
    for x ∈ T do
        add n − 1 copies of x to S'_train
    return S'_train
```

Together these three straightforward example attacks demonstrate the breadth of the data forging attack surface. The model owner's full control over data can be used to attack certificates of multiple properties, with and without prior knowledge of audit data.

4.2 Statistical Methods Fail to Detect Data Forging

One might seek to strengthen CMC protocols against attacks in the previous section by leveraging statistical tests, e.g., Welch's t-test to assess whether the model owner's dataset was drawn from a different distribution than the audit set. Indeed

[10] Assuming that a model which almost always outputs 0 is not highly accurate in our scenario.

previous studies propose zero-knowledge proofs of statistical tests for this purpose [9, 12]. In full version, however, we prove that adversaries can forge datasets which successfully circumvent CMCs augmented with a variety of statistical tests.

Our results show that an attacker can forge datasets that successfully evade CMC accuracy audits, while yielding outputs which are indistinguishable from the audit set for a broad family of functions. At a high level, we demonstrate that the attacker can cause the distributional properties of the training data to converge towards the audit data without sacrificing the efficacy of the attack in Algorithm 1 by adding more copies of the audit data to the training data. This causes the audit data and the training data to appear as if they were drawn from the same distribution under a variety of statistical tests without impacting the model's ability to learn the desired behavior.

Thus, functions within this family cannot effectively thwart attackers satisfying Definition 3. We apply this result in Corollary 1 which shows that a data forging attacker can always evade detection by Welch's t-test.

Corollary 1. *Given an audit dataset S_{audit} and significance level α, we can use Algorithm 1 to construct a training dataset S'_{train} such that for any feature j, S'_{train} passes Welch's t-test when its values in feature j are compared to those of S_{audit} with significance level α.*

Proof of Corollary 1 and the results leading up to it are deferred to the full version of the paper for brevity.

The difficulty of detecting data forging attacks against CMCs, combined with the variety of certification protocols they can evade, motivate a new perspective on CMC security which we present in the following section.

5 Secure Cryptographic Model Certification

In this section, we present our positive results on *provably secure* cryptographic model certification (CMC) protocols, with full details provided in the full version. We begin by introducing the syntax and security definitions for CMC in § 5.1. We then present a template for secure CMC and our main theorem in § 5.2, followed by an example instantiation for accuracy auditing in § 5.3. As we will later see in the case study, our protocol template allows us to restore/expand the security of some of the constructions from Table 1.

5.1 Syntax and Security Definitions

Definition 4 (Cryptographic Model Certification Protocol)**.** *A cryptographic model certification (CMC) protocol Π for a distributional predicate F is a tuple of algorithms* $(\mathsf{Commit}, \mathsf{Prove}, \mathsf{Audit})$*: a commitment, a proving, and an auditing algorithm. Let* $(\mathsf{com}, b) \leftarrow \langle \mathsf{Prove}(h, S_{train}), \mathsf{Audit} \rangle$ *denote the interaction between* Prove *and* Audit*, where* Prove *takes a hypothesis h and optionally a training dataset S_{train} as private input and outputs a commitment* com *during an execution, and* Audit *halts by outputting a binary b.*

We propose the following security properties for a CMC protocol Π for a distributional predicate F. We focus on high-level intuition and defer formal cryptographic definitions to full version. While completeness, binding, and zero knowledge are direct adaptations of those of commitments and zero-knowledge proofs, $\tilde{F}$-relaxed knowledge soundness is a novel property tailored to our CMC setting. In particular, we call this property "$\tilde{F}$-relaxed" to allow parameterizing the soundness requirement by a predicate $\tilde{F}$ that is only a close approximation of the desired property F. As we shall see in concrete examples, this relaxation turns out necessary because the auditor only checks the property f on a *finite sample*, which may not perfectly reflect the property F on the *underlying distribution*. In our concrete instantiations, we will quantify the gap between F and $\tilde{F}$.

Completeness. If an honest prover holds a model h and a training set S_{train} such that $F(h, S_{\text{train}}) = 1$, then this prover should pass the audit (i.e, Audit outputs $b = 1$) except with small error probability p.

Binding. No computationally bounded prover can change its model h or S_{train} after generating a commitment com to them.

Zero Knowledge. An honest execution of the protocol between Prove and Audit does not reveal any information about the model h beyond the fact that $F(h, S_{\text{train}}) = 1$.

$\tilde{F}$-relaxed Knowledge Soundness. If a (potentially dishonest prover) Prove^* commits to a pair of invalid model h and training dataset S_{train}, i.e., $\tilde{F}(h, S_{\text{train}}) \neq 1$, then Audit should detect it by outputting $b = 0$ with high probability. In slightly more detail, for any Prove^*, there exists an efficient *knowledge extractor* Ext that extracts a pair (h, S_{train}) from Prove^* such that the above condition holds (except with small error probability κ).

5.2 Secure CMC Template

We now define a *commit-sample-prove* CMC protocol $\Pi_{\text{csp}} = (\mathsf{Commit}, \mathsf{Prove}, \mathsf{Audit})$. The protocol starts with the model provider committing to its model h using a commitment scheme Commit. Upon receiving the commitment, the auditor responds with an *audit dataset* S_{audit}, consisting of n i.i.d. samples drawn from a distribution $\mathcal{D}$ over the query space $Q = \{(x_i, y_i)\}_{i=1}^m$. Finally, the model provider and the auditor run a zero-knowledge proof (ZKP) protocol $\mathsf{ZKP} = (\mathcal{P}, \mathcal{V})$ to prove that the committed model h satisfies an *empirical predicate* f on the sampled audit dataset S_{audit}, i.e., ZKP for the relation $\mathcal{R}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathcal{R} \iff f(h, S_{\text{audit}}) = 1 \wedge \mathsf{com} = \mathsf{Commit}(h; \rho)$. Here, the empirical predicate f should be chosen to closely approximate the desired property F over the underlying distribution $\mathcal{D}$; we will formalize this requirement as small *false negative rate*. Moreover, f should be chosen

such that the *false positive rate* of f is small with respect to the relaxed predicate $\tilde{F}$. Intuitively, this means that if a model h does not satisfy the relaxed property $\tilde{F}$, then it should fail the audit with high probability (i.e., soundness), while if a model h satisfies the desired property F, then it should pass the audit with high probability (i.e., completeness). While we focus on a protocol checking F on a hypothesis h only, the construction below can be naturally extended to a more complex F and f that additionally take a training dataset S_{train} as input.

$\langle \mathsf{Prove}(h), \mathsf{Audit} \rangle$

1. Prove computes $\mathsf{com} = \mathsf{Commit}(h; \rho)$ using a uniformly random string ρ and sends com to Audit.
2. Audit samples $S_{\text{audit}} \sim \mathcal{D}^n$ and sends it to Prove.
3. Prove and Audit execute $b \leftarrow \langle \mathcal{P}(\mathsf{w}), \mathcal{V} \rangle(\mathsf{x})$, where $\mathsf{x} = (\mathsf{com}, S_{\text{audit}})$ and $\mathsf{w} = (h, \rho)$. Here, Prove plays $\mathcal{P}$ and Audit plays $\mathcal{V}$ of ZKP.
4. Prove outputs com, while Audit outputs b.

The key takeaway is that the audit dataset S_{audit} should be chosen *independently* of the model h and the training dataset S_{train}. Sampling S_{audit} after h is committed ensures independence, rendering our earlier attack ineffective. The following theorem (formally stated and proved in the full version) states that, if the empirical predicate f satisfies the bounds mentioned below, Π_{csp} satisfies the desired security properties in a general fashion. **Essentially, our result allows the protocol designer to choose an empirical predicate f that approximates the desired property F well enough, and then plug in any commitment scheme and ZK proof system satisfying the required security properties to get a secure auditing scheme.**

Theorem 2. *Let the empirical predicate f, the distributional predicate F, and the relaxed distributional predicate $\tilde{F}$ satisfy the following false negative and false positive rate bounds for every model h:*

$$\Pr_{S_{audit} \sim \mathcal{D}^n}[f(h, S_{audit}) \neq 1 \mid F(h) = 1] \leq p_{fnr}$$
$$\Pr_{S_{audit} \sim \mathcal{D}^n}[f(h, S_{audit}) = 1 \mid \tilde{F}(h) \neq 1] \leq p_{fpr}$$

If Commit *and* ZKP *satisfy the standard security properties, then the protocol Π_{csp} is **complete** with error probability p_{fnr}, **binding**, **zero knowledge**, and $\tilde{F}$**-relaxed knowledge sound** with error probability $\approx p_{fpr}$.*

Proof sketch. We focus on $\tilde{F}$-relaxed knowledge soundness, as the other properties directly follow from the underlying commitment scheme and ZK proof system with standard security guarantees. Our proof carefully unifies standard arguments from learning theory with the proof technique of "knowledge extraction with interleaved probabilistic tests", commonly used in advanced proof systems [4, 8, 24]. To illustrate the intuition, let us assume the commitment scheme is perfectly binding and straightline extractable, i.e., once the prover sends com to the auditor, one can immediately extract a unique model h and a unique randomness ρ such that $\mathsf{com} = \mathsf{Commit}(h; \rho)$. Such a commitment scheme can be constructed in the common reference string (CRS) model using public key encryption. If the committed model h does *not* satisfy $\tilde{F}$, then by the assumption on the false positive rate, the probability that $f(h, S_{\text{audit}}) = 1$ is at most p_{fpr} over the choice of S_{audit}.

The actual proof is significantly more involved, when the commitment scheme is *only computationally binding* (which is the case for most practical instantiations). In particular, we need to construct a meta-extractor that runs the knowledge extractor $\mathcal{E}$ for ZKP to obtain a candidate model h, and rewinds $\mathcal{E}$ with fresh $S'_{\text{audit}} \sim \mathcal{D}^n$ to ensure the validity of h via probabilistic tests (or otherwise break binding of the commitment). Our formal security proof takes care of these subtle technicalities.

5.3 Example: CMC for Accuracy

We detail example instantiations of f and F for accuracy and demographic parity auditing, both of which enable arbitrary small false positive and negative rates with a sufficient number of samples n. To instantiate Π_{csp} for accuracy auditing, we consider the empirical and true error rates as follows:

$$\hat{\ell}_S(h) = \frac{1}{n} \sum_{(x,y) \in S} \mathbb{I}(h(x) \neq y) \quad \ell(h) = \mathbb{E}_{(x,y) \sim \mathcal{D}}[\mathbb{I}(h(x) \neq y)]$$

where $n = |S|$, and define the empirical predicate f, the distributional predicate F, and the relaxed predicate $\tilde{F}$ as follows.

$$f(h, S_{\text{audit}}) = 1 \iff \hat{\ell}_S(h) \leq t + \delta$$
$$F(h) = 1 \iff \ell(h) \leq t$$
$$\tilde{F}(h) = 1 \iff \ell(h) \leq t + 2\delta$$

With these definitions in place, we can bound both the false negative p_{fnr} and false positive rates p_{fpr} by $2e^{-2n\delta^2}$ by a standard concentration bound argument (see full version). By plugging them into Theorem 2, we conclude that Π_{csp} instantiated for accuracy auditing is a secure CMC with the following properties for a sufficiently large $n\delta^2$.

- If the true error of the model h is $\leq t$, then the auditor accepts with high probability.
- If the auditor accepts, then the auditor gets assurance that the true error of the model h is at most $t + 2\delta$, even if the prover is misbehaving.

As usual in learning theory, there is a trade-off between the sample size n and the tightness of the confidence parameter

δ. By tuning δ appropriately, one can reduce the number of samples n needed to achieve desired false positive and negative rates. This is particularly important in practice since 1) the computational cost of ZKP typically scales with n, and 2) gathering large audit datasets can be challenging. For instance, by choosing $(n, \delta) = (10\text{k}, 0.02)$ or $(n, \delta) = (40\text{k}, 0.01)$, we can achieve both false positive and negative rates ≤ 0.001. Since typical dataset sizes in practice are in the order of tens or hundreds of thousands, these configurations enable high-confidence auditing while maintaining reasonable computational costs for ZKP.

6 Case Studies: Vulnerability to Data Forging in Previous Work

Our formalization from §5 and §4 lets us test whether a protocol is vulnerable to data-forging, and provides a way to lift the security guarantees of prior works to resist data forging attacks. For CMC works which reveal neither the model nor the training data, the most important check boils down to whether the prover is required to commit to the training data and/or to the model before seeing the audit dataset.

We examined several prior works [7,14,21,22,28,29,35,38,41] with formal security guarantees. Surprisingly, the majority of them either do not explicitly state when the audit dataset is revealed, or assume the prover's training dataset and/or the model itself to be trusted (and are susceptible to data forging if the prover is actually malicious). Works that do not discuss the timing of the commitment often point out that their solution can be used to conduct audits using publicly known datasets, in which case the public dataset can be assumed to be known to the adversary prior to auditing, making the solution vulnerable to data-forging.

We present two of our case studies below, and discuss the rest in §C. We note that works that are not susceptible to our data-forging attacks nonetheless provide only dataset-specific guarantees, i.e., their proofs certify properties solely on the chosen audit set/inference queries already submitted by clients, without extending to the underlying data distribution. It would be interesting to perform an analysis similar to that in §5 to derive formal guarantees that hold for the corresponding distributions. We summarize the results of our findings in Table 1.

6.1 Zero Knowledge Proofs for Decision Tree Predictions and Accuracy

Goal and Solution Details. Zhang et al. [41] introduce protocols for auditing accuracy and verifying decision tree predictions without revealing any additional information about the model itself. Zhang et al.'s main contribution is in designing a custom zero-knowledge proof tailored to efficiently verifying the decision tree prediction. The proof consists of algorithms to generate public parameters, custom commitment algorithm for decision tree models, the prover's algorithm which outputs a proof of inference/accuracy, and verifier algorithm to check this proof. The prover, i.e., model provider, must first commit to its model and subsequently demonstrate that the predictions on client queries are consistent with this commitment. For accuracy verification, the authors propose a batching technique that allows to more efficiently checks the correctness of predictions across *multiple* inputs. They then add an extra verification step to determine how many of these predictions match the true labels.

Security Model. Zhang et al.'s security notion follows the traditional definition structure of a zero-knowledge proof system, where the protocol is required to satisfy correctness, soundness, and zero-knowledge. Both parties can be malicious. In the context of our analysis we are interested in soundness, which specifies whether a malicious prover can deceive the verifier (i.e., auditor), that the prover's hypothesis passed the test. At a high level, the authors' soundness definition can be summarized as follows: A prover should not be able to output a commitment to a tree $\mathcal{T}$ along with a proof π, an accuracy threshold *acc*, and a dataset D, such that the verifier accepts the proof, and at the same time, $\mathcal{T}'s$ accuracy on D is not equal to *acc*.

Discussion. The security notion that Zhang et al's work provides aligns well with the intuitive goals of verifying both the correctness of individual predictions and the accuracy of a model on a given dataset. However, it does not provide formal guarantees for datasets beyond the audited one, i.e., the accuracy verification solution does not generalize to other datasets drawn from the same distribution. The protocol description further does not explain how the audit dataset is sampled. In fact, Zhang et al. explicitly note that it is possible to use their solution to check accuracy on a *public dataset*. In this setting, their approach falls within our framework of Definition 3, and is vulnerable to the same attack as outlined in §4.1. In fact, note that our example works even if the prover supplies an *additional* proof of training to complement its proof of accuracy.

Expanding/Restoring Security. Using our techniques, Zhang et al.'s construction can be easily adapted to satisfy the guarantee with respect to new datasets drawn from the same distribution. In fact, this corresponds precisely to our example in § 5.3, where Zhang et al.'s construction can be plugged in as the ZKP of our protocol Π_{csp} in §5.2.

6.2 Confidential-DPproof: Confidential Proof Of Differentially Private Training

Goal and Solution Details. Shamsabadi et al. [28] present Confidential-DPproof, a framework that enables the model provider to prove to an auditor that their model was correctly trained via DP-SGD [1], a classic approach for training models with differential privacy guarantees. The certification of

DP-SGD's training run is done in zero-knowledge.

Security Model. Shamsabadi et al. [28] consider a setting with a mutually distrusting model provider and auditor. The model provider is fully malicious, while the auditor is semi-honest. Confidential-DPproof considers standard definitions of correctness, soundness, and zero-knowledge.

Discussion. As we describe in §3 ("Attacking Differential Privacy Certification"), Shamsabadi et al.'s solution is susceptible to data forging attacks.

Limitations of Our Secure Template in the Privacy Setting. Defining practical empirical predicates for privacy testing is more challenging for privacy than for accuracy or fairness. Empirical audits of differential privacy, such as membership inference attack success rate [20], often incur hefty computational costs. Further, both privacy audits and differentially private training necessitate assumptions about training data which are difficult to validate in the CMC model without further cryptographic infrastructure.

As a result, the approach we introduced above for Zhang et al. [41] does not apply to Confidential-DPproof. The core of the issue that Confidential-DPproof is essentially a proof of training, and the data for this training is necessarily chosen by the model provider, not by the auditor. One possible way to still make use of Confidential-DPproof guarantees would be to verify that the training data was generated by a set of trusted parties independent from the model provider. For instance, each individual in the dataset could sign their data record with a distinct key pair. The service provider could then prove that each training record verifies under a single individual's public key. This would thwart the attack in Algorithm 3, as duplication of data points would not be possible. However, instantiating such a scheme efficiently and proving its security to more general data forging attacks is a challenge beyond the scope of the present paper. We reserve it as an interesting direction for future work.

7 Evaluation

In this section we underscore the importance of data forging attacks by mounting proof of concept attacks for models trained on a variety of datasets. We show that our attack is effective in making inaccurate models appear accurate and unfair models appear fair, and empirically demonstrate a variety of other qualities, e.g. undetectability with a variety of statistical tools.

Experimental Setup. We use six well-known fairness benchmarking datasets in our experiments: ACSEmployment [11], Adult [6], COMPAS [3], German Credit [19], Default Credit [39], and Communities & Crime [26]. We implemented a modified version of our attack from §4.1 in Python 3.12.3 using SciKit-Learn version 1.6.1 that attempts to minimize an objective when deployed and evaluated its performance against a variety of datasets. For a given run, we split the dataset into an evaluation dataset consisting of 30% of the data, an audit dataset containing 1000 data points (500 for German Credit due to its smaller size), and an initial training data set. We represent the interpolation between a fully honest training run and a fully malicious one by the *attack parameter* $p \in [0,1]$. Concretely, p controls two components of the forged training set S'_{train}: (i) the fraction of audit points that contribute $\pm\varepsilon$-perturbed neighbors (with flipped labels), and (ii) the fraction of S_{train} rows whose labels are flipped. The specifics of how honest and malicious data points are labeled depends on the objective.

To attack accuracy, we constructed a training dataset using a modified Algorithm 1, labeling the additional data from the attack with $1 - r_y$ rather than 0 and adding them to the initial training data set. For our fairness attack, we constructed the training data similarly, changing how honest and malicious data points were labeled to match Algorithm 2. Honest data points were given random labels, while malicious data points were labeled according to their sensitive attribute. Both attacks then fit a decision tree to their constructed training data using SciKit-Learn's decision tree classifier class.

To ensure that our attack would evade statistical detection, we added extra copies of the audit data to the training data, as computed in Corollary 1 to pass Welch's t-test with significance level 0.05.

Attacking Accuracy Audits. We ran our attack on six benchmark datasets – three shown in Figure 3 (remaining in Figure 5). Across all datasets, our data forging attack enforces high audit accuracy while simultaneously encouraging low performance on real-world evaluation data. Thus **our attack successfully makes inaccurate models *appear* accurate to an auditor.**

Attacking Fairness Audits. We also performed the attack while targeting demographic parity (using sex as the sensitive attribute) on three datasets, which we present in Figure 4. We were able to reliably train a model with close to 0 fairness gap on the audit dataset, but close to 1 fairness gap when deployed. In other words, **our attack successfully makes unfair models *appear* fair to an auditor.**

Evading Detection via Statistical Methods. In the full version, we show how our attack can be executed in ways that evade detection by a variety of statistical approaches. We were able to construct malicious training datasets with summary statistics that match those of the audit dataset very closely, and Welch's t-test and Levene's test regularly concluded that the audit and test datasets were drawn from the same distribution.

Additional Results. An adversary can use data forging attacks to achieve concrete goals beyond degradation of accuracy or fairness, as we show in the Appendix D. For example, Figure 6 shows how an insurance provider could use our attack to hide the claim denial rate of a model from auditors. We have also conducted preliminary experiments that suggest that variations of our attack generalize to XGBoost (§ D.1) and neural network models (§ D.2). Figure 8 shows the effectiveness of the attack on synthetic mixture-of-gaussian data,

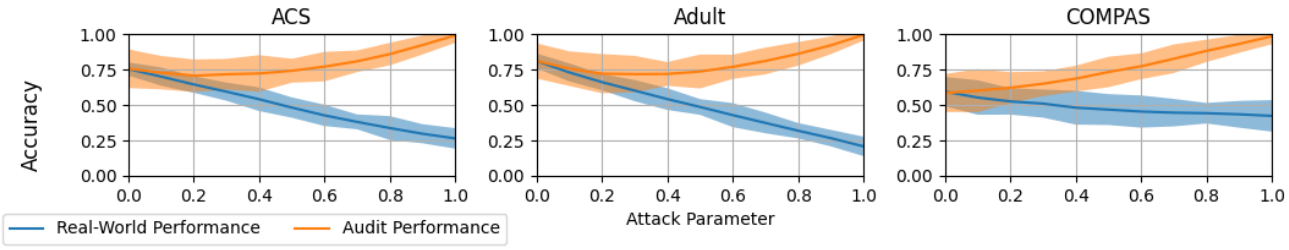


Figure 3: Performance of models trained on datasets constructed to minimize real-world accuracy while still passing an audit for several benchmarks. Values are averages over ten runs, error bars represent one standard deviation.

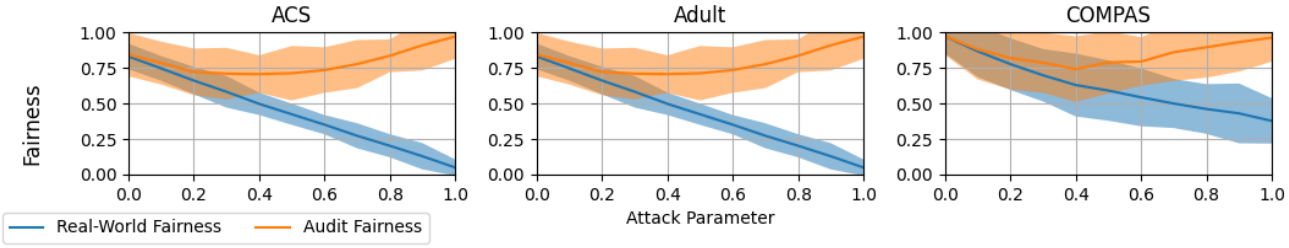


Figure 4: Fairness of models trained on constructed datasets using various benchmarks to target demographic parity. Values are averages over ten runs, error bars represent one standard deviation. Fairness means 1 − fairness gap.

and we have regularly achieved less than 70% accuracy on the ACS Employment dataset while demonstrating nearly 90% accuracy on the audit set.

8 Conclusion and Future Work

This work brings attention to data-dependent vulnerabilities in CMC. We showed that, in several proposed certification frameworks, a provider can pass an audit while undermining its intended guarantees in real-world deployment. We proposed concrete attack strategies and provided preliminary evidence that standard statistical tests are unlikely to reliably detect them. Empirically, we demonstrated the effectiveness of these attacks for certifications for accuracy and fairness. We then introduced new formal security definitions which allow property certification on data *distributions* (rather than data *sets*), thereby ruling out the data-dependent attacks identified in this work. We then presented a general-purpose protocol satisfying our definition.

Our results should not be interpreted as showing that prior CMC constructions are technically incorrect. Rather, they show that the guarantees they provide may not align with the guarantees required in realistic auditing scenarios. More broadly, our work illustrates the importance of close communication between the cryptography, security, and machine learning communities when designing future certification frameworks, so that formal guarantees are aligned with the threats that arise in deployment.

The attack strategy presented in this work poses several open questions. While we demonstrate the data forging attack is undetected even in the presence of Welch's t-test and Levene's test, it remains to be seen whether other statistical tests could effectively detect the attack. Based on the results that we have derived, we find it unlikely that other statistical tests will be effective in detecting the attack. However, we reserve such analysis for future work. We provide rigorous formal proofs that our attacks are effective on decision trees, and preliminary evidence that a similar approach generalizes to neural networks. Characterizing a formal relationship between neural network model capacity and attack effectiveness could be a promising direction in future work.

Our secure CMC template underscores the importance of keeping audit data hidden until the service provider's model is committed. This imposes a limitation on auditing in practice: auditors must either regularly gather fresh data (since the audit dataset is typically revealed during the audit), use additional cryptographic techniques such as secure multiparty computation to keep data hidden during the audit, or perform continuous auditing on user data. Each of these options has strengths and drawbacks which should be evaluated in more detail by future work. Moreover, while we focused on a CMC template based on fresh audit datasets, it remains unclear how to securely certify distributional properties that do not involve an audit dataset, such as differentially private training. A promising direction for future work is a countermeasure against the third attack in §4.1, which likely requires a more advanced CMC mechanism to certify the well-formedness of the training data.

Acknowledgments & Disclaimer

We thank Chen-Da Liu-Zhang for many helpful discussions, and Jonas Guan, Sierra Wyllie and Mohammad Yaghini for helpful feedback on an earlier draft. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute. The work was further supported by the CBI postdoctoral fellowship.

This paper was prepared in part for information purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co and its affiliates ("JP Morgan"), and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

Ethical Considerations

In this paper, we pointed out assumption gaps both in prior works and real-world auditing practices on cryptographic model certification (CMC), and demonstrated data forging attacks that exploit these gaps. We provide a stakeholder ethics analysis to evaluate the ethical implications of our work.

Stakeholders: The primary stakeholders affected by our work include the authors, machine learning practitioners, organizations deploying machine learning models, end-users relying on these models, and researchers in the field of machine learning security and cryptography.

Impacts: This study does not involve human subjects or personal data, and thus does not raise direct ethical concerns related to privacy or consent. Our findings point out an assumption (i.e., whether the auditing dataset is known to the model provider ahead of time) that was not explicitly considered in prior works and real-world auditing practices, which is crucial for the security of CMC schemes.

Mitigation Strategies: As the research community and industry are currently only experimenting with proof-of-concept implementations of CMC and have not yet deployed such systems in production, we conclude that there is no risk to end-users due to our findings. We hope that our work will raise awareness of the importance of explicitly considering the assumption we identified in both research and practice, and contribute to the development of more robust and secure CMC schemes.

Open Science

In accordance with the open science policy, we provide the source code of our implementation and necessary files required for running our experiments. These can be found at https://zenodo.org/records/20337318.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Kasra Abbaszadeh, Christodoulos Pappas, Jonathan Katz, and Dimitrios Papadopoulos. Zero-knowledge proofs of training for deep neural networks. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 4316–4330, 2024.

[3] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias. *ProPublica*, May 2016. Retrieved January 15, 2022.

[4] Thomas Attema, Michael Klooß, Russell W. F. Lai, and Pavlo Yatsyna. Adaptive special soundness: Improved knowledge extraction by adaptive useful challenge sampling. Cryptology ePrint Archive, Paper 2024/2038, 2024. URL: https://eprint.iacr.org/2024/2038.

[5] Marco Barreno, Blaine Nelson, Anthony D. Joseph, and J. D. Tygar. The security of machine learning. *Mach. Learn.*, 81(2):121–148, 2010.

[6] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996.

[7] Jade Garcia Bourrée, Hadrien Lautraite, Sébastien Gambs, Gilles Trédan, Erwan Le Merrer, and Benoît Rottembourg. P2NIA: privacy-preserving non-iterative auditing. *CoRR*, abs/2504.00874, 2025. URL: https://doi.org/10.48550/arXiv.2504.00874, arXiv:2504.00874, doi:10.48550/ARXIV.2504.00874.

[8] Benedikt Bünz and Ben Fisch. Multilinear schwartz-zippel mod N and lattice-based succinct arguments. In Guy N. Rothblum and Hoeteck Wee, editors, *Theory of Cryptography - 21st International Conference, TCC 2023, Taipei, Taiwan, November 29 - December 2, 2023,*

Proceedings, Part III, volume 14371 of *Lecture Notes in Computer Science*, pages 394–423. Springer, 2023. doi:10.1007/978-3-031-48621-0_14.

[9] Ian Chang, Katerina Sotiraki, Weikeng Chen, Murat Kantarcioglu, and Raluca Popa. {HOLMES}: Efficient distribution testing for secure collaborative learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 4823–4840, 2023.

[10] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi:10.1145/2939672.2939785.

[11] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] Vasisht Duddu, Anudeep Das, Nora Khayata, Hossein Yalame, Thomas Schneider, and N Asokan. Attesting distributional properties of training data for machine learning. In *European Symposium on Research in Computer Security*, pages 3–23. Springer, 2024.

[13] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and trends® in theoretical computer science*, 9(3–4):211–407, 2014.

[14] Olive Franzese, Ali Shahin Shamsabadi, and Hamed Haddadi. Oath: Efficient and flexible zero-knowledge proofs of end-to-end ml fairness. *arXiv preprint arXiv:2410.02777*, 2024.

[15] Sanjam Garg, Aarushi Goel, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, Guru-Vamsi Policharla, and Mingyuan Wang. Experimenting with zero-knowledge proofs of training. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1880–1894, 2023.

[16] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985. doi:10.1145/22145.22178.

[17] Andy Greenberg. How one of apple's key privacy safeguards falls short. *Wired*. URL: https://www.wired.com/story/apple-differential-privacy-shortcomings/.

[18] Moritz Hardt. The emerging science of machine learning benchmarks. *Manuscript. https://mlbenchmarks.org*, 2025.

[19] Hans Hofmann. Statlog (German Credit Data). UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5NC77.

[20] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022.

[21] Daniel Kang, Tatsunori Hashimoto, Ion Stoica, and Yi Sun. Scaling up trustless DNN inference with zero-knowledge proofs. *CoRR*, abs/2210.08674, 2022. URL: https://doi.org/10.48550/arXiv.2210.08674, arXiv:2210.08674, doi:10.48550/ARXIV.2210.08674.

[22] Tianyi Liu, Xiang Xie, and Yupeng Zhang. zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 2968–2985. ACM, 2021. doi:10.1145/3460120.3485379.

[23] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229, 2019.

[24] Ngoc Khanh Nguyen. *Lattice-Based Zero-Knowledge Proofs Under a Few Dozen Kilobytes*. PhD thesis, ETH Zurich, 10 2022. https://www.research-collection.ethz.ch/handle/20.500.11850/574844.

[25] Christodoulos Pappas and Dimitrios Papadopoulos. Sparrow: Space-efficient zksnark for data-parallel circuits and applications to zero-knowledge decision trees. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3110–3124, 2024.

[26] Michael Redmond. Communities and Crime. UCI Machine Learning Repository, 2009.

[27] Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577*, 2018.

[28] Ali Shahin Shamsabadi, Gefei Tan, Tudor Ioan Cebere, Aurélien Bellet, Hamed Haddadi, Nicolas Papernot, Xiao Wang, and Adrian Weller. Confidential-dpproof: Confidential proof of differentially private training. In *International Conference on Learning Representations (ICLR)*, 2024.

[29] Ali Shahin Shamsabadi, Sierra Calanda Wyllie, Nicholas Franzese, Natalie Dullerud, Sébastien Gambs, Nicolas Papernot, Xiao Wang, and Adrian Weller. Confidential-profitt: confidential proof of fair training of trees. In *The Eleventh International Conference on Learning Representations*, 2022.

[30] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *2013 IEEE global conference on signal and information processing*, pages 245–248. IEEE, 2013.

[31] Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. In *NIPS*, pages 3517–3529, 2017.

[32] Haochen Sun, Tonghe Bai, Jason Li, and Hongyang Zhang. Zkdl: Efficient zero-knowledge proofs of deep learning training. *IEEE Transactions on Information Forensics and Security*, 2024.

[33] Sarah Tan, Rich Caruana, Giles Hooker, and Yin Lou. Distill-and-compare: Auditing black-box models using transparent model distillation. In *AIES*, pages 303–310. ACM, 2018.

[34] Florian Tramer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. Fairtest: Discovering unwarranted associations in data-driven applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 401–416. IEEE, 2017.

[35] Haodi Wang and Thang Hoang. ezdps: An efficient and zero-knowledge machine learning inference pipeline. *Proc. Priv. Enhancing Technol.*, 2023(2):430–448, 2023. URL: https://doi.org/10.56553/popets-2023-0061, doi:10.56553/POPETS-2023-0061.

[36] Bernard L Welch. The generalization of 'student's'problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

[37] Royce J Wilson, Celia Yuxin Zhang, William Lam, Damien Desfontaines, Daniel Simmons-Marengo, and Bryant Gipson. Differentially private sql with bounded user contribution. *Proceedings on Privacy Enhancing Technologies*, 2020.

[38] Chhavi Yadav, Amrita Roy Chowdhury, Dan Boneh, and Kamalika Chaudhuri. Fairproof: Confidential and certifiable fairness for neural networks. *arXiv preprint arXiv:2402.12572*, 2024.

[39] I-Cheng Yeh and Che hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2, Part 1):2473–2480, 2009. URL: https://www.sciencedirect.com/science/article/pii/S0957417407006719, doi:10.1016/j.eswa.2007.12.020.

[40] Guanhua Zhang, Florian E Dorner, and Moritz Hardt. How benchmark prediction from fewer data misses the mark. *arXiv preprint arXiv:2506.07673*, 2025.

[41] Jiaheng Zhang, Zhiyong Fang, Yupeng Zhang, and Dawn Song. Zero knowledge proofs for decision tree predictions and accuracy. In *CCS*, pages 2039–2053. ACM, 2020.

[42] Tianyu Zhang, Shen Dong, Oyku Deniz Kose, Yanning Shen, and Yupeng Zhang. Fairzk: A scalable system to prove machine learning fairness in zero-knowledge. In Marina Blanton, William Enck, and Cristina Nita-Rotaru, editors, *IEEE Symposium on Security and Privacy, SP 2025, San Francisco, CA, USA, May 12-15, 2025*, pages 3460–3478. IEEE, 2025. doi:10.1109/SP61157.2025.00205.

A Related Work

We now briefly discuss further prior works in the field of model certification. A number of recent works aim to prove desirable model properties. In terms of *what* these works prove, they can be roughly categorized into proofs of training, inference, accuracy, and fairness. In terms of *how* the corresponding protocols work, recent works on certifiable ML can be categorized as follows:

Cryptographic approaches. A prolific line of research adapts various cryptographic techniques to certify properties such as accuracy, fairness, etc., without revealing the model's details. The most common technique is *zero-knowledge proofs* (zk proofs), which allow to formally prove that a model satisfies certain properties without revealing anything else about the model. They have been used to certify fairness [14, 29, 38, 42], inference [41], accuracy [41], and to prove that the model has been trained using a certain algorithm [2,15,25,32] (without revealing the training data). Other works [9, 12] use *secure multi-party computation* (MPC), which allows mutually distrusting parties to jointly compute on private inputs without revealing anything about the inputs apart from the outcome.

Black box auditing/Statistical testing. These approaches probe a model by submitting inputs, collecting outputs, and analyzing them for undesirable behavior. [27, 34] use black-box testing to check for potential unfairness or bias, while [33] distill a new model to gain insight into the black box one.
Outside-the-box auditing. Here the model owner provides access to information beyond query responses, such as source code, documentation [23], hyperparameters, training data, deployment details, or internal evaluation results.

B Proof of Theorem 1

Proof. First, let us show that if points a and b, where $a_i < c_i < b_i$ and $c_j = b_j$ for all $j \neq i$, get sorted into the same decision tree leaf, then c is also sorted into that same leaf. Notice that in order for two points x and y to get sorted into different leaves, there must be a node which splits on a feature i such that $x_i \neq y_i$ and $x_i \leq t \leq y_i$ or $y_i \leq t \leq x_i$ where t is the threshold to split upon. Then if b and c were sorted into separate leaves, there must be a node on the path that b takes through the tree that splits on feature i with a threshold t that satisfies $c_i \leq t \leq b_i$. However, such a node would also sort a distinctly from b, so such a node cannot occur. Thus, b and c must be sorted into the same leaf.

Now, note that as we train the tree until its leafs are homogeneous, every datapoint in S'_{train} must be classified correctly (according to the label we assigned to it in Algorithm 1). Further, since the only datapoints in S'_{train} with non-zero labels are datapoints from S_{audit}, for every leaf in $\mathcal{T}$ that is associated with a non-zero class, we have at least one $r \in S_{audit}$ that gets sorted into this leaf.

Consider $x \in \mathbb{R}^d$ such that $||x - r||_\infty \geq \varepsilon$ for all $r \in S_{audit}$. Say $\mathcal{T}(x) = 1$, i.e., there exists a leaf such that x belongs to this leaf and the leaf corresponds to class one. Consider $r \in S_{audit}$ that belongs to this leaf (by above, such r exists). By the definition of the L-infinity norm there exists some dimension i where $|r_i - x_i| > \varepsilon$. Suppose $r_i - x_i > \varepsilon$. Notice that there is a point $r - \varepsilon\vec{b}_i \in S'_{train}$ which satisfies that $(r - \varepsilon\vec{b}_i)_j = r_j$ for all $j \neq i$, and where $x_i < (r - \varepsilon\vec{b}_i)_i < r_i$. Then by above, $r - \varepsilon\vec{b}_i$ must be sorted into the same leaf as r and x. But $r - \varepsilon\vec{b}_i$ has label $g(r) = 0$, while for x holds $\mathcal{T}(x) = 1$. Thus, we found a contradition. The same argument holds if $x_i - r_i > \varepsilon$, but using the point $r + \varepsilon\vec{b}_i$ instead of $r - \varepsilon\vec{b}_i$. □

C Case Study

We now discuss a number of state of the art works that consider the problem of privacy-preserving auditing. These works are focused on different auditing functions (accuracy, fairness, etc), different types of machine learning models, and their security models they use are not necessarily aligned. We now briefly outline the techniques and security guarantees that are claimed in each of the works. Our goal is not to provide an exhaustive survey, but rather to illustrate the landscape through recent works that are broadly representative of the field—even though they span different years, venues, and communities (ranging from machine learning to security).

C.1 zkCNN: Zero Knowledge Proofs for Convolutional Neural Network Predictions and Accuracy

Goal and Solution Details. [22] propose zkCNN, a zero-knowledge proof protocol for inference and accuracy of convolutional neural networks (CNNs). The core contribution is a novel sumcheck protocol (which is the key ingredient in many zero-knowledge system) that is tailored to two-dimensional convolutions.
Security Model. zkCNN considers the standard setting with a prover and a verifier. Either party can be malicious. [22]'s security definition for inference is a zero-knowledge-style definition, and the scheme is required to satisfy correctness, soundness, and zero-knowledge. Similar to [41], [22]'s soundness intuitively states that a prover should not be able to output a commitment to a model and provide a proof π, prediction y and datapoint X such that the verifier accepts the proof, and at the same time, the committed model's prediction for X is not equal to y. If instantiated with a specific commitment scheme, [22]'s scheme further satisfies knowledge soundness, the stronger version of soundness where there exists an extractor to extract the CNN parameters from a valid proof and prediction with overwhelming probability. [22] do not provide a security definition for their proof of accuracy.
Discussion. As [22] do not give a security definition for their proof of accuracy, the formal security guarantee they provide is not fully clear. However, the authors indicate that their scheme can be used to prove the accuracy on a public dataset. This scenario falls within our framework of definition 3, and is vulnerable to the same style of attack as outlined in §4.1.
Expanding/Restoring Security. Similar to Zhang et al. [41], [22]'s construction can be adapted using our techniques from § 5.3 to satisfy the guarantee with respect to new datasets drawn from the same distribution. In particular, Liu et al.'s scheme can be plugged in as the ZKP of our protocol Π_{csp} in §5.2.

C.2 Confidential-PROFITT: Confidential PROof of FaIr Training of Trees

Goal and Solution Details. Shamsabadi et al. [29] propose Confidential-PROFITT, a framework for certifying fairness of decision trees while preserving confidentiality of both the model and the training data. Confidential-PROFITT consists of a zero-knowledge-friendly decision tree learning algorithm that, when executed honestly, enforces fairness by design—up to a tunable degree controlled by a parameter. On top of this,

Confidential-PROFITT designs a zero-knowledge proof system to verify fairness of a decision tree. The proof requires the model provider to commit to both the model and its training data, then prove in zero-knowledge that the paths taken by the committed training points through the (committed) decision tree satisfy specified fairness bounds. In terms of fairness metrics, Confidential-PROFITT supports demographic parity and equalized odds.

Security Model. Confidential-PROFITT considers a malicious model provider (that, however, is assumed to commit to the training data honestly) and a malicious auditor, and obtains standard zero-knowledge proof properties (correctness, soundness, zero-knowledge) with respect to a statement that can be summarized roughly as follows "With respect to a private dataset *chosen by the model provider*, the committed model satisfies certain fairness guarantees".

Discussion. Confidential-PROFITT assumes that the model provider honestly commits to the training data. Under this assumption, the corresponding zero-knowledge proof certifies that the resulting model inherits the fairness guarantees of the fair learning algorithm introduced in Confidential-PROFITT (which the authors show indeed improves fairness). However, if the provider is not restricted to committing to the true training data, Confidential-PROFITT is vulnerable to data-forging attacks, as the provider can choose the audit dataset before committing to the model.

Expanding/Restoring Security. We believe Confidential-PROFITT's zero-knowledge proof of fairness could potentially be combined with our techniques from § 5.3 if one would swap the training data for the data sampled by the auditor. However, the adaptation is not as straight-forward as that for Zhang et al. [41] and Liu et al. [22]. This is because Confidential-PROFITT verifies not only the actual fairness guarantee, but also additional fairness checks for intermediate nodes in the decision tree. A careful analysis is required to ensure that the generalization of fairness properties also extends to intermediate nodes. We leave this as an interesting direction for future work.

C.3 P2NIA: Privacy-Preserving Non-Iterative Auditing

Goal and Solution Details. Bourrée et al. [7] propose a novel auditing scheme that enables one-shot verification of a model's group fairness while preserving privacy for both parties: the model provider is not required to open-source the model, and the auditor need not disclose any private information to support the audit. The main contribution of [7] is a mechanism that enables auditing without requiring the auditor to supply the audit dataset. Specifically, the model provider supplies a dataset together with the corresponding predictions (both in the clear), which the auditor then uses to verify the fairness condition. To construct this dataset, model provider draws on a portion of its internal training data. To preserve confidentiality of this data, it is not shared directly. Instead, model provider feeds it into a synthetic data generation algorithm, and the resulting synthetic dataset is what is sent to the auditor.

Security Model. The work does not provide a formal security model. It is set up in the black-box setting and assumes that the auditor does not know the distribution of the model owner's training data.

Discussion. As [7] do not utilize cryptographic techniques to prove that the outputs actually correspond to the given inputs, the prover can easily cheat by simply adjusting the labels it supplies for the constructed dataset. However, even if one were to strengthen the scheme by adding a secure proof of training (e.g., [25]) together with inference proofs (as in [41]), the fact that the model owner knows the dataset that is being used for the audit means that the solution falls within our framework of Definition 3, and is thus vulnerable to data-forging attacks. An interesting open question would be to see if, since in this scenario the model owner not only knows, but directly influences the audit dataset, there can be an even simpler attack.

C.4 OATH: Efficient and Flexible Zero-Knowledge Proofs of End-to-End ML Fairness

Goal and Solution Details. [14] present OATH, a model-agnostic fairness auditing framework. The core idea in OATH is to leverage clients (who query the model during deployment) to participate in the auditing process. OATH operates in two phases: (i) a certification protocol between the model provider and the auditor, and (ii) a query authentication protocol involving model provider, inference clients, and auditor (dubbed verifier in OATH). The first phase follows the standard certification flow we describe in §2. In the second phase, the auditor receives commitments to client queries and the corresponding model predictions. These commitments can later be verified in zero knowledge for fairness, correctness, and consistency with the certified model.

Security Model. OATH considers three fully malicious entities: a model provider, inference clients, and an auditor. These parties are assumed not to collude with each other. The auditor assesses model fairness both with respect to the calibration dataset and the clients queries. The system provides standard correctness, soundness, and zero-knowledge with respect to these two datasets.

Discussion. The calibration dataset which is used in the certification protocol between the model provider and the auditor might be supplied by either party. If the calibration dataset is chosen by the prover, same as P2NIA and Confidential-PROFITT, the corresponding fairness check is vulnerable to data forging. However, in contrast to prior works, OATH can fall back on guarantees based on client's queries.

C.5 FairProof: Confidential and Certifiable Fairness for Neural Networks

Goal and Solution Details. [38] propose FairProof, a fairness certification approach that maintains confidentiality of the model. In contrast to Confidential-PROFITT and OATH, which focus on group fairness metrics, FairProof considers local individual fairness. This allows [38] to issue a personalized certificate to every client.

Security Model. FairProof system involves a malicious model provider and malicious clients (who wish to learn model details/training data), and considers standard correctness, soundness, and zero-knowledge properties. The corresponding statement is roughly as follows: "Given a datapoint x, the model's output is y and a lower bound on an individual fairness parameter for x is ε_x".

Discussion. The usage of a specific fairness metric (local individual fairness) allows FairProof to provide per-client certificates of fairness, and escape the problems that arise from the usage of reference datasets (including vulnerability to data-forging attacks). On the flip side, FairProof requires to generate fairness certificates during deployment and does not provide any fairness guarantees prior to deployment.

C.6 Scaling up Trustless DNN Inference with Zero-Knowledge Proofs

Goal and Solution Details. Kang et al. [21] propose a zero-knowledge-based framework for verifying DNN inference and accuracy. Their key contribution is a careful translation of DNN specifications into arithmetic circuits suitable for zero-knowledge proofs. The system also introduces economic incentives to support ML-as-a-service. Concretely, when verifying accuracy, the model provider first commits to the model, and the client commits to the test set. Both parties then deposit monetary collateral into an escrow. The client reveals the test set, and the provider must produce a zero-knowledge proof that the committed model meets the claimed accuracy. If the provider fails or refuses to prove the required accuracy, it forfeits its collateral; otherwise, the client pays for the service.

Security Model. Kang et al. study the standard two-party setting with a *prover* (model provider) and a *verifier* (client), either of whom may be malicious. Cryptographically, they aim for the standard zero-knowledge proof properties: *completeness*, *knowledge soundness*, and *zero knowledge*. They further consider incentives, showing that—under certain assumptions—honest model providers and clients are motivated to participate in the accuracy verification protocol, while malicious parties are discouraged.

Discussion. In terms of cryptographic guarantees, Kang et al. gets the core design right: their protocol for proofs of accuracy closely follows the framework outlined in §5 and is not vulnerable to our data-forging attacks. However, [21] provide no formal guarantees about accuracy on data outside the audited set.

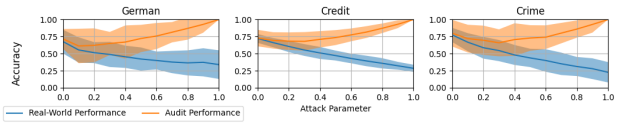


Figure 5: Accuracy of models trained on datasets constructed by Algorithm 1 on various benchmarks. Values are averages over ten runs, error bars represent one standard deviation.

C.7 ezDPS: An Efficient and Zero-Knowledge Machine Learning Inference Pipeline

Goal and Solution Details. Wang and Hoang [35] introduce ezDPS, a pipeline for zero-knowledge proofs of inference correctness and accuracy above a specified threshold. They construct arithmetic circuit gadgets for key ML operations, including exponentiation, absolute value, and array max/min, and further devise optimized methods for proving Discrete Wavelet Transform, Principal Component Analysis, and multi-class Support Vector Machines with various kernel functions using an efficient set of arithmetic constraints.

Security Model. [35] consider two mutually distrusting parties – a malicious server and a semi-honest client, who follows the protocol but aims to learn information about the model's parameters. For their inference pipeline, they consider standard definitions of correctness, soundness, and zero-knowledge (similar to those by [41] and [21]). [35] do not provide a security definition for their proof of accuracy.

Discussion. Similar to Liu et al. [22], as [35] do not provide a security definition for their proof of accuracy, the precise security guarantee they achieve is somewhat unclear. However, [35] indicate that their scheme can be used to prove the accuracy on a public dataset, which falls within our framework of definition 3. This instantiation of their method is vulnerable to the same style of attack as outlined in §4.1.

Expanding/Restoring Security. Similar to Zhang et al. [41] and Liu et al. [22], the scheme of Wang and Hoang can be adapted using our techniques from § 5.3 to satisfy the guarantee with respect to freshly sampled datasets. In particular, their protocol can be plugged in as the ZKP of our protocol Π_{csp} in §5.2.

D Further Evaluation

First, we present in Figure 5 results for attacking accuracy audits on additional datasets mentioned in §7.

Next, we present the application of the attack described in § 4.1. In this attack, the adversary is attempting to maximize the model's denial rate $\Pr_{x\sim\mathcal{D}}[h(x)=0]$ while still appearing accurate to the audit. The results of this attack are given in

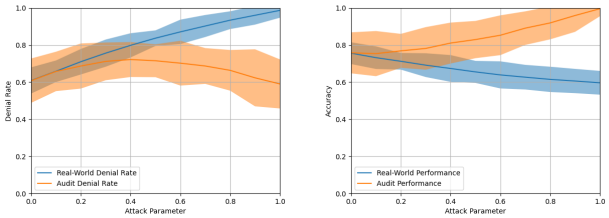


Figure 6: Accuracy and denial rates of models trained on datasets constructed by Algorithm 1 on ACSEmployment. Values are averages over ten runs, error bars represent one standard deviation.

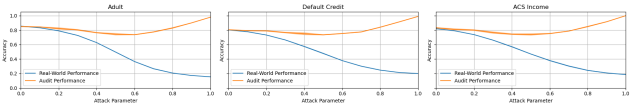


Figure 7: Accuracy of XGBoost models trained on inflated datasets constructed by the modified Algorithm 1 on Adult, Default Credit, and ACS Income datasets. Values are averages over five runs; error bars represent one standard deviation.

Figure 6. Observe that as the attack parameter approaches 1 (and the attack becomes maximally malicious), the denial rate of the model on the audit set remains close to the fully honest denial rate while the denial rate on independently sampled data approaches 1. Similarly, the accuracy of the model on the audit set approaches 1, while the true accuracy decreases down to roughly 0.6 (this reflects the true denial rate of the distribution).

D.1 Attacking XGBoost

We next show that a minor modification of Algorithm 1 suffices to evade accuracy auditing of XGBoost models [10]. Unlike a single decision tree, XGBoost trains a gradient-boosted ensemble of trees, so the theoretical guarantee of Theorem 1 does not apply: with only one copy of S_{audit} in S'_{train}, the boosting process assigns excessive weight to the $2d$ flipped-label neighborhood points, causing the ensemble to misclassify the audit points themselves and thereby fail the audit. To counteract this, we include $2d$ copies of S_{audit} (with true labels) in S'_{train} rather than a single copy, so that the correct-label signal at each audit point outweighs its $2d$ flipped-label neighbors; all other details of the construction remain unchanged. We used the perturbation magnitude $\varepsilon = 1.0$ throughout. The XGBoost ensemble is configured with 500 trees of maximum depth 10, learning rate $\eta = 0.3$, L2 regularization $\lambda = 1.0$, and minimum split-loss $\gamma = 0$; all other XGBoost hyperparameters are left at their defaults. We empirically evaluated this attack on the Default Credit, Adult Income, and ACS Income datasets; results are shown in Figure 7. Our modified attack successfully evades accuracy auditing across all benchmarks.

D.2 Attacking Neural Networks

Finally, we present an evaluation of a modified version of the attack that targets neural networks rather than decision trees. Whereas decision trees have very specific conditions that allow us to constrain their behavior, it is much harder to provide theoretical guarantees for neural networks. In order to encourage memorization of the training data, we used a relatively shallow network with very large individual layers. Our attack samples a large amount of training data, and decides whether to label each point with the honest label or dishonest label depending on its proximity to the nearest audit data point. We evaluated this attack on an 8-dimensional mixture of Gaussian distributions; the results are shown in Figure 8.

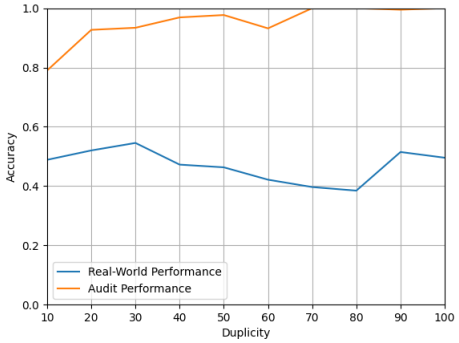


Figure 8: Performance of 226M-parameter neural networks trained on datasets constructed from a mixture of Gaussian distributions. Duplicity refers to the number of perturbed copies of the audit dataset included in the training data.

We also performed this attack on ACS Employment data, and over 10 runs, trained a model with average accuracy of 69.83% accuracy on held-out test data while maintaining an average of 88.82% accuracy on audit data.

D.3 Evading Statistical Methods for Attack Detection

Empirical and theoretical results demonstrating that statistical methods are insufficient to detect data forging, including a proof of Corollary 1, are deferred to the full version of the paper due to page limit of the USENIX format.