WP-016 | ZKML Verifiable Inference | June 2026

Verifiable ML Inference: Zero-Knowledge Proofs as Cryptographic Output Attestation for AI Models

AffixIO Research June 2026 WP-016 Download PDF

Third parties need to trust model outputs in credit, insurance, and public-sector AI. ZKML lets you prove a specific inference satisfied policy constraints without publishing weights or user data. We survey what is production-ready in 2026 and what still belongs in the lab.

CONTENTS

1. Why Verifiable AI Output Matters
2. The ZKML Concept: Proofs of Inference
3. ZKML Architecture: From ONNX to ZK Circuit
4. Proving Systems for ML Workloads
5. Production Readiness in 2026
6. AffixIO Constraint-Based Inference Circuits
7. Private AI: Protecting Model Weights
8. Confidential Inference for Regulated Domains
9. On-Chain and Decentralised Verification
10. Regulatory Drivers for Verifiable ML
11. Limitations and the Path Forward
12. Conclusions

1. Why Verifiable AI Output Matters

Artificial intelligence systems are increasingly making or informing decisions with significant consequences: credit scoring, clinical triage, insurance underwriting, fraud detection, content moderation, and legal research. In each of these contexts, affected parties, regulators, and auditors have a legitimate interest in knowing not just what the AI decided but whether the decision was produced by the model that is claimed, under the conditions that are claimed, without interference or substitution.

Today, this assurance is provided by organisational attestations: the model operator asserts, in documentation or under audit, that their systems operate as described. There is no cryptographic mechanism by which an external party can independently verify that a specific AI output was produced by a specific model. This gap is not merely theoretical. Model substitution, output tampering, and backdoor attacks are practical concerns in adversarial deployments. Even in non-adversarial contexts, the absence of cryptographic verification means that audit relies entirely on the model operator's honesty and the auditor's ability to examine systems they do not control.

Zero-knowledge machine learning addresses this gap. A ZK proof of inference produces a cryptographic statement that a specific model, applied to a specific input, produced a specific output. This statement can be verified by any party who holds the proof and the model's public parameters, without requiring access to the model itself, the input data, or the model operator's infrastructure.

The practical applications are significant. A clinical AI system can prove that its triage recommendation was produced by the approved, validated model, not a development version or a tampered copy. A credit-scoring model can prove that a rejection decision was not based on protected characteristics. A content moderation system can prove that a removal decision was produced by the stated policy model, enabling appeal processes to operate on verifiable rather than asserted grounds.

---

2. The ZKML Concept: Proofs of Inference

A zero-knowledge proof of inference (ZKPoI) is a cryptographic proof that satisfies three properties with respect to a machine learning model M, an input x, and an output

y:

- **Completeness:** If M(x) = y is true, a prover who knows M and x can produce a valid proof.
- **Soundness:** If M(x) ≠ y, no computationally bounded prover can produce a valid proof. The prover cannot falsely claim that M produced y.
- **Zero-knowledge:** A verifier who receives the proof learns that M(x) = y is true, but learns nothing else about M, x, or y beyond what is explicitly committed.

The zero-knowledge property can be selectively relaxed depending on the use case. In some deployments, the input x is public but the model M is private. In others, both x and M are private and only y is public. The circuit design determines which values are witnesses (private inputs to the prover) and which are public inputs to the verifier.

What a ZKML Circuit Encodes

An ML model is, mathematically, a function composed of arithmetic operations: matrix multiplications, activation functions, normalisations, and aggregations. A ZKML circuit encodes this arithmetic computation as a set of polynomial constraints. Proving the circuit amounts to proving that a specific assignment of values to the circuit's wires satisfies all the constraints, which is equivalent to proving that the computation was performed correctly.

The challenge is scale. A typical image classification model with ten million parameters requires encoding roughly a billion arithmetic operations as polynomial constraints. Even with aggressive circuit optimisation, the resulting constraint system is enormous. This is why ZKML production deployments in 2026 focus on smaller, narrower models rather than general-purpose large language models.

---

3. ZKML Architecture: From ONNX to ZK Circuit

The dominant toolchain for ZKML in 2026 converts models from ONNX format, the standard interchange format for neural networks, into arithmetic circuits suitable for ZK

proof systems.

ONNX as the Starting Point

ONNX (Open Neural Network Exchange) provides a framework-agnostic representation of ML models as computation graphs. Virtually all major ML frameworks, including PyTorch, TensorFlow, JAX, and scikit-learn, can export models to ONNX. This makes ONNX a natural entry point for ZKML toolchains.

Quantisation and Arithmetic Compatibility

ZK proof systems operate over finite fields, not floating-point numbers. Converting a floating-point model to a finite-field circuit requires quantisation: representing floating-point weights and activations as integers in a fixed-point scheme. The quantisation process introduces a small numerical error relative to the original floating-point model, which must be bounded and characterised as part of the ZKML deployment process. A deployment specification should include the maximum allowable divergence between the floating-point reference output and the quantised circuit output.

EZKL: The Production ZKML Toolchain

EZKL is the leading open-source ZKML toolchain in 2026. It accepts an ONNX model and a calibration dataset, performs quantisation, generates a circuit in the halo2 constraint system, and produces a proving key and verification key pair. The proving key is used by the model operator to generate proofs; the verification key is distributed to parties who need to verify proofs. EZKL supports models up to approximately 50 million parameters for production proof generation times under ten seconds on modern hardware.

AffixIO's Noir-Native Approach

AffixIO implements ZKML circuits natively in Noir, the same language used for the governance, identity, and compliance circuits described in earlier whitepapers. This provides a unified circuit language across all AffixIO proof types, enabling composite proofs that combine an ML inference proof with a governance policy proof in a single verification step. The Noir-native approach also benefits from the current proving

scheme backend's performance characteristics, which are optimised for the specific constraint patterns that arise in neural network arithmetic.

---

4. Proving Systems for ML Workloads

The proving system chosen for ZKML has significant implications for proof generation time, proof size, and the trust assumptions required for verification.

Proving System	Constraint System	Trusted Setup	ZKML Suitability
Groth16	R1CS	Per-circuit ceremony	Smallest proofs, but setup overhead is impractical for per-model deployments
PLONK / UltraPlonk	Custom gates	Universal SRS	Good for medium circuits; universal setup amortises across models
current proving scheme (proving backend)	UltraPlonk gates	Universal SRS	AffixIO's choice; excellent for policy circuits with complex arithmetic
halo2	PLONKish	None (IPA)	EZKL's choice; no trusted setup requirement simplifies deployment
STARKs (SP1, Risc0)	AIR	None	Post-quantum secure but larger proofs; excellent for programs, not just circuits

The choice of proving system involves trade-offs between proof size, generation time, verification time, trust assumptions, and post-quantum security. For enterprise deployments with privacy requirements, AffixIO recommends current proving scheme

for Noir-native circuits and halo2 for ONNX-converted circuits via EZKL, with a planned migration to STARK-based systems as ML workload support matures.

---

5. Production Readiness in 2026

The ZKML production frontier in 2026 is defined by model size and domain. The following assessment reflects the state as of the first half of 2026.

Production-Ready Domains

Image classification models with fewer than 50 million parameters achieve proof generation times under 30 seconds on commodity hardware, suitable for batch inference workflows. Fraud detection and anomaly detection models, which typically use tree-based or shallow neural network architectures, are well within production range with sub-second proof generation. Recommendation models using collaborative filtering or embedding lookups are provable at scale. Credit scoring models, which commonly use logistic regression or gradient-boosted trees, are particularly well suited to ZKML because their arithmetic is simple and their regulatory compliance requirements are high.

Near-Term Production Horizon

Large language model inference, including full transformer forward passes for models in the 7B parameter range, remains too computationally expensive for real-time ZK proof generation in 2026. Proof generation for a single forward pass of a 7B model takes tens of minutes on current hardware. This is actively being addressed through recursive proof composition, hardware acceleration on GPUs and FPGAs, and layer-wise proving strategies such as the NANOZK approach published in March 2026, which generates proofs layer by layer and composes them into a single session-level proof.

The Hardware Acceleration Trajectory

Proof generation time is dominated by multi-scalar multiplication (MSM) and fast Fourier transform (FFT) operations. Both are highly parallelisable and benefit

significantly from GPU and custom ASIC acceleration. Several hardware acceleration providers have released ZKML-specific proof accelerators in 2025 and 2026, with benchmarks showing 100x to 1000x speedups relative to CPU-only generation. The trajectory suggests that 7B parameter LLM inference proofs will reach production latency ranges within 24 to 36 months.

---

6. AffixIO Constraint-Based Inference Circuits

AffixIO provides a library of policy circuits for common inference patterns, designed for integration into enterprise AI pipelines where regulatory compliance requires verifiable output attestation.

The inference_gate Circuit

The `inference_gate` circuit proves that a binary classification model applied to a committed input produced a specific binary output. It is designed for use cases where a yes/no decision must be proven without revealing the decision criteria or the input features. Applications include eligibility decisions, fraud flags, and content policy violations.

Reference implementation omitted from public documentation.

The score_infer Circuit

The `score_infer` circuit proves that a scoring model produced a value within a specified range, without revealing the exact score. This is directly applicable to credit scoring under the EU AI Act's transparency requirements: the lender can prove that the decision was based on a score within a particular band, satisfying the explainability obligation, without disclosing the exact score or the model parameters.

Composite Governance Proofs

AffixIO's most distinctive ZKML capability is composite proof generation: combining an ML inference proof with a governance policy proof in a single circuit. A composite

proof can simultaneously assert that a specific model produced a specific output AND that the model is the approved version in the governance registry AND that the inference was performed within the authorised policy scope. This eliminates the need for separate verification of the inference and the governance record, reducing audit overhead and ensuring that model substitution cannot occur undetected.

---

7. Private AI: Protecting Model Weights

One of the most commercially significant applications of ZKML is the protection of proprietary model weights. A company that has invested substantially in training a specialised model can prove the model's performance characteristics to a customer or regulator without disclosing the weights themselves.

The Model Confidentiality Problem

In traditional AI audits, the auditor requires access to the model, either directly or through a controlled environment. This creates a disclosure risk: the auditor may extract or copy the weights, the audit environment may be compromised, or the audit requirements may force disclosure to parties the model operator does not fully trust.

ZKML as a Confidential Audit Mechanism

With ZKML, the audit process changes. The model operator generates a commitment to the model weights and publishes it. The operator then runs the model on audit test inputs and generates ZK proofs of the outputs. The auditor verifies the proofs against the committed weights and the stated outputs, confirming that the model behaves as described, without ever receiving the weights. The commitment ensures that the model cannot be changed between audits without detection.

Benchmark Attestation

ZKML enables publicly verifiable benchmark attestations. A model developer can claim that their model achieves 94.7% accuracy on a standard evaluation dataset, and then publish a ZK proof of inference for every sample in that dataset, proving the claim

without revealing the model. This changes the nature of benchmark competition: claimed performance becomes verifiable performance, eliminating the incentive for benchmark manipulation.

---

8. Confidential Inference for Regulated Domains

In regulated domains, ZKML provides a mechanism for combining model confidentiality with input confidentiality, enabling privacy-preserving inference as a service.

Healthcare: Confidential Clinical Decision Support

A clinical decision support system can use ZKML to prove that a triage or diagnostic recommendation was produced by the approved, validated clinical AI model, without revealing the patient's input features. The hospital can demonstrate regulatory compliance, the patient's data remains private, and the model vendor's proprietary weights are protected. All three parties' interests are served without any party disclosing to another.

Financial Services: Explainable Credit Decisions

The EU AI Act and the Consumer Credit Directive require that credit decisions based on automated processing be explainable to applicants. ZKML provides a mechanism for producing verifiable explanations that are bounded by what the model actually computed: the lender can prove that the decision was produced by the stated model applied to the committed input features, satisfying the transparency requirement without revealing other customers' data or model parameters.

Legal and Compliance: Sanctioned Model Verification

Compliance functions that use AI for sanctions screening, AML, or legal research can use ZKML to prove to regulators that the AI system applied to a specific transaction or document was the approved, sanctioned model. This is directly applicable to DORA's

ICT governance requirements, which mandate evidence that approved systems were used.

---

9. On-Chain and Decentralised Verification

While enterprise ZKML deployments primarily involve off-chain proving and on-demand verification, the blockchain context was instrumental in driving early ZKML development. Understanding the on-chain verification pattern informs enterprise deployment architecture.

The On-Chain Verification Model

In decentralised AI applications, the model runs off-chain for efficiency, but its outputs must be trustable by on-chain smart contracts. ZKML solves this by producing a proof on-chain that the off-chain computation was performed correctly. The blockchain verifier accepts the proof and the public outputs, verifying the proof on-chain without re-running the model. This is economically efficient because proof verification is substantially cheaper than model inference.

Implications for Enterprise Architecture

The on-chain verification model translates naturally to enterprise contexts where a central verification authority, such as a regulator, compliance function, or audit firm, needs to verify AI outputs produced by multiple model operators. The verifier holds the verification key; model operators submit proofs; the verifier checks proofs without running any model. This architecture scales to large numbers of model operators without proportional increases in verifier compute requirements.

---

10. Regulatory Drivers for Verifiable ML

Several regulatory frameworks are creating direct demand for ZKML capabilities, either explicitly or through requirements that can only be satisfied cryptographically.

EU AI Act Article 13: Transparency and Output Verification

Article 13 requires that high-risk AI systems be designed to enable users to interpret the system's output and use it appropriately. In practice, this is often interpreted as a requirement for explainability tooling. ZKML adds a deeper layer: not just explaining what the model predicted, but proving that the stated model produced the stated prediction. The difference is significant in dispute resolution contexts where a party claims that a different model was used or that the output was tampered with after generation.

GDPR Article 22: Automated Decision-Making

GDPR Article 22 gives data subjects the right not to be subject to solely automated decisions with significant effects, and the right to obtain human review of such decisions. ZKML enables a verifiable audit trail for automated decisions: the model, the input commitment, and the output are provably linked, giving the human reviewer a cryptographic basis for their review rather than an unverifiable asserted record.

Financial Services Sector-Specific Requirements

The EBA Guidelines on internal governance and the Basel Committee's principles for the use of AI in credit risk modelling require that model risk management frameworks include validation of model outputs. ZKML provides a form of continuous, cryptographic model validation: every inference is proven, every proof is verifiable, and any deviation from the approved model is detectable.

---

11. Limitations and the Path Forward

Current Constraints

The primary constraint on ZKML adoption in 2026 is the computational cost of proof generation for large models. For LLM-scale models, proof generation is not yet practical for real-time applications. Quantisation error introduces a bounded divergence between the floating-point reference model and the ZK circuit, which must be characterised and accepted as part of the deployment process. The toolchain ecosystem, while maturing rapidly, still requires significant engineering expertise to deploy.

Recursive Proof Composition

Recursive proof composition, where a proof about a proof is generated, is the key technique enabling ZKML to scale to larger models. By generating proofs layer by layer and composing them recursively, the proving workload can be distributed across multiple machines and the total proof size can be kept constant regardless of model depth. This technique, implemented in folding schemes such as Nova and HyperNova, is expected to make LLM inference proofs practical within the two-to-three year horizon.

Post-Quantum Security for ZKML

Current ZKML proving systems based on elliptic curve cryptography are not post-quantum secure. For deployments where long-term verifiability is required, such as regulatory records that must remain verifiable for seven or more years, STARK-based proving systems that rely only on hash functions are preferable. AffixIO's migration path from elliptic-curve-based circuits to STARK-based circuits is described in the post-quantum attestation whitepaper (WP-002).

---

12. Conclusions

Zero-knowledge machine learning has crossed a significant threshold in 2026. For narrow, well-scoped inference tasks, ZKML is production-ready: proof generation times are practical, toolchains are mature enough for enterprise deployment, and the regulatory demand for verifiable AI output is accelerating. AffixIO's Noir-native ML

inference circuits provide a path to composite governance proofs that combine inference attestation with policy compliance verification in a single cryptographic statement.

The implications for AI governance are substantial. Organisations that can prove, cryptographically and on demand, that a specific model produced a specific output under specific policy constraints are fundamentally better positioned for regulatory compliance, dispute resolution, and stakeholder trust than those that rely on organisational assertions and audit sampling. As the EU AI Act, GDPR Article 22, and financial services AI requirements mature, the expectation of verifiable ML output will transition from differentiator to baseline requirement.

AffixIO's ZKML circuit library is available under an open-source MIT licence, with enterprise support for integration into existing MLOps pipelines and regulatory compliance programmes.

Related reading

- WP-004: Real-Time Zero-Knowledge Governance in the AI Response Pipeline
- WP-005: Source Verification as a Zero-Knowledge Circuit Input
- WP-001: Cryptographic AI Governance: A Technical Framework

Frequently asked questions

What is verifiable ML inference?

A cryptographic proof that a named model version produced a stated output on committed inputs, verified without exposing those inputs.

Is ZKML fast enough for production?

Small circuits verify in milliseconds; proving still costs hundreds of milliseconds to seconds depending on model size and hardware.

Which tools does AffixIO use?

constraint language and proving backend for governance circuits, with EZKL and related stacks for ML-specific workloads where circuit size allows.

All White Papers

▶ About
▶ Solutions
▶ Legal
▶ Trust & Security

Contact

truth layer | yes | no | proof