

ZK-GovProof: Privacy-Preserving Verification of Regulatory Compliance in AI Governance

Srishti Dutta

Independent

Hackathon Submission - Track 2: Compliance Infrastructure & Privacy-Preserving Proofs

February 1, 2026

Open Source Implementation



https://github.com/srishtiii28/tech_ai_gov

Complete working code, circuits, policy packs, and documentation

Outline

- 1 The Problem
- 2 The Solution
- 3 Implementation
- 4 Novel Contributions
- 5 Results & Performance
- 6 Use Cases
- 7 Limitations & Honest Assessment
- 8 Live Demo
- 9 Conclusion

The AI Governance Dilemma

How do you prove regulatory compliance
without revealing sensitive information?

Regulators Need:

- Proof of compute thresholds
- Evidence of safety evaluations
- Policy compliance verification
- Auditability

AI Labs Need:

- Competitive confidentiality
- Exact compute privacy
- Evaluation score secrecy
- IP protection

Traditional Audits Fail

Full disclosure conflicts with commercial interests and national security

EU AI Act - Article 55

GPAI models with systemic risk (training compute $\geq 10^{25}$ FLOPs) must:

- Conduct model evaluations
- Assess systemic risks
- Track serious incidents
- Ensure cybersecurity
- Report energy consumption

Anthropic Responsible Scaling Policy (RSP)

ASL-3 threshold (10^{26} FLOPs) requires:

- Security measures for model weights
- Dangerous capability evaluations
- Deployment safeguards

Prove compliance without revealing secrets

Core Idea

Use cryptographic proofs to demonstrate:

- **Compute Threshold:** "My model's compute $<$ threshold" (exact value stays private)
- **Evaluation Attestation:** "I completed all N evaluations" (scores stay private)
- **Policy Compliance:** "I satisfied M/N requirements" (details stay private)

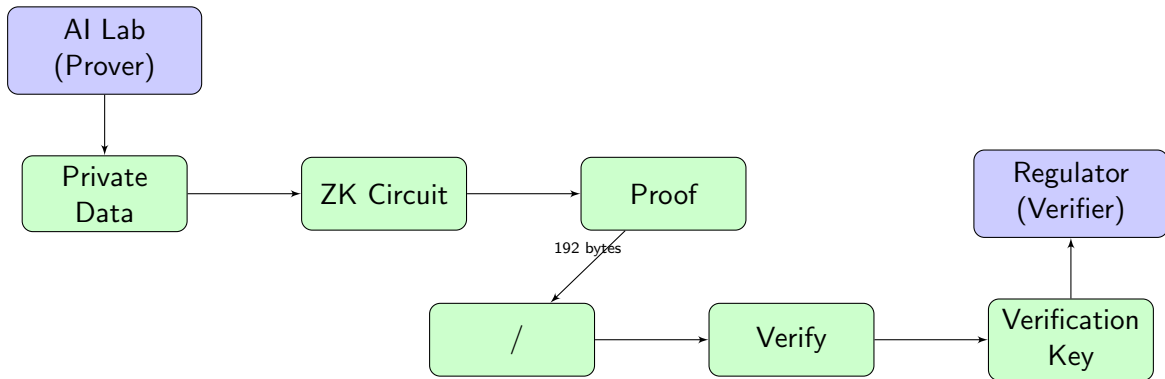
✓ Verifier Gets:

- Cryptographic proof
- Boolean compliance result
- Mathematical certainty

🔒 Prover Keeps Secret:

- Exact compute values
- Evaluation scores
- Implementation details

System Architecture



- **Prover:** AI lab generates proof using private data + circuit
- **Verifier:** Regulator checks proof in 0.4 seconds
- **Privacy:** Exact values never revealed, only compliance boolean

Three Core Circuits

Compute Threshold

Circuit 1

$\text{privateCompute} < \text{threshold}$

Proves model is below/above regulatory compute limit

Evaluation Attestation

Circuit 2

$\text{sum}(\text{flags}) \geq N$

Proves required evaluations completed

Policy Compliance

Circuit 3

$\text{sum}(\text{items}) \geq M$

Proves policy checklist satisfied

Technology Stack

- **Proof System:** Groth16 (optimized for verification speed)
- **Circuit Language:** Circom
- **Curve:** BN128 (Ethereum-compatible)
- **Library:** snarkjs (JavaScript)

Encoding Regulatory Frameworks as JSON

```
{
  "id": "eu_ai_act_article55_systemic_risk_v1",
  "title": "EU AI Act Article 55 - GPAI Systemic Risk",
  "frameworks": ["EU AI Act"],
  "compute_threshold": {
    "public_threshold_flops": "1e25",
    "requirement": "Training compute >= 10^25"
  },
  "required_evaluations": {
    "required_count_public": 5,
    "evaluation_names": [
      "Model evaluation", "Systemic risk assessment",
      "Adversarial testing", "Incident tracking",
      "Cybersecurity measures"
    ]
  }
}
```

1. Composable Proof Architecture

The Problem

Traditional approach: Verify each requirement separately

Compute check $\rightarrow$ verify (0.4s)

Evaluation check $\rightarrow$ verify (0.4s)

Policy check $\rightarrow$ verify (0.4s)

Total: 1.2 seconds

Our Solution

Composable architecture: Verify all requirements once

Compute + Evaluation + Policy $\rightarrow$ verify (0.4s)

Total: 0.4 seconds

3 $\times$ efficiency improvement

2. Verifiability Gap Analysis

Honest assessment: What can ZK proofs actually verify?

✓ CAN Verify

- Compute thresholds
- Evaluation counts
- Checklist completion
- Cryptographic signatures

◐ PARTIALLY Verify

- Monitoring systems (structure, not operation)
- Access controls (policy, not enforcement)
- Training data (hashes, not content)

✗ CANNOT Verify

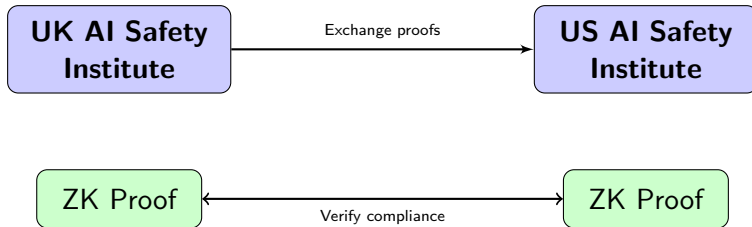
- Human judgment
- Organizational culture
- Ongoing processes
- Real-world impact

Solution: Defense-in-Depth

ZK proofs + Hardware attestation + Traditional audits + Ongoing monitoring

3. International Coordination Architecture

Bilateral verification without full trust



Use Case: UK and US AI Safety Institutes can verify each other's regulatory compliance without revealing:

- Exact model capabilities
- Specific evaluation methodologies
- Classified national security information

Production-Ready Performance

Metric	Value	Context
Proof Generation	18.3s	One-time (prover)
Proof Verification	0.4s	Repeated (verifier)
Proof Size	192 bytes	Single packet
Public Signals	1-8 bytes	Minimal metadata
Composite Verify	0.4s	3 proofs in 1 op
Efficiency Gain	3×	vs. separate

Key Insight

Asymmetric costs: Proving is expensive (18s), verifying is fast (0.4s). Perfect for regulatory use cases where one prover submits to many verifiers.

What We Built

Implementation

- 3 working Groth16 circuits
- 2 policy pack implementations
- Proof generation scripts
- Verification scripts
- Composable proof system
- 5 clean git commits

Documentation

- Research paper (13 pages)
- README (26KB)
- Threat model analysis
- Verifiability gap taxonomy

Validation

- Real proof generation
- Successful verification
- Negative test (invalid proofs fail)
- Performance benchmarks
- End-to-end demos

Real Frameworks

- EU AI Act Article 55
- Anthropic RSP ASL-3
- Extensible to other regulations

Real-World Applications

1. Cross-Border AI Governance

Problem: International AI Safety Institutes need mutual verification without compromising national security

Solution: Exchange ZK proofs of compliance

Impact: Global coordination while preserving sovereignty

2. Supply Chain Attestation

Problem: Cloud providers need to prove infrastructure meets safety requirements

Solution: ZK proofs of compute capacity, security, monitoring

Impact: Trust without transparency

3. Automated Regulatory Verification

Problem: Regulators can't manually audit thousands of AI systems

Solution: Automated proof verification (0.4s per model)

Impact: Scalable governance

Additional Circuits

- Federated learning aggregation
- Differential privacy guarantees
- Model performance thresholds
- Training data compliance
- Energy consumption limits

More Regulations

- US Executive Order 14110
- UK AI Safety Institute framework
- China AI regulations
- OECD AI Principles
- Industry-specific standards

Integration Opportunities

- Hardware attestation (TEEs, TPMs)
- Blockchain anchoring for auditability
- Decentralized verification networks
- Continuous monitoring systems

What ZK Proofs Don't Solve

Limitations

- ❶ **Garbage In, Garbage Out:** Proofs verify predicates, not truthfulness of inputs
- ❷ **Negative Claims:** Can't prove "no other unreported training runs exist"
- ❸ **Ongoing Processes:** Can't verify continuous monitoring or organizational culture
- ❹ **Semantic Gaps:** Legal requirements (e.g., "reasonable care") don't map to circuits

Our Approach: Defense-in-Depth

ZK Proofs

Hardware
Attestation

Traditional Audits

Threat Model

Threat	Mitigated By	Residual Risk
False compute claims	ZK circuit constraints	Low
Proof forgery	Cryptographic soundness	Negligible
Input fabrication	Hardware attestation	Medium
Replay attacks	Timestamp + nonce	Low
Verifier key compromise	Key management	Medium
Unreported training runs	Traditional audits	High
Semantic manipulation	Legal review	High

Key Insight

ZK proofs are **one powerful tool** in a comprehensive governance framework, not a silver bullet.

Demo: EU AI Act Compliance

1. Generate Proofs

```
$ node scripts/run-policy-pack.js \  
  policies/eu_ai_act_article55_gpai_systemic.json
```

Generating proofs...

compute_threshold proof (18.2s)

evaluation_attestation proof (18.5s)

policy_compliance proof (18.1s)

Wrote: data/policy_packs/*/artifacts.json

2. Verify (Regulator Side)

```
$ node scripts/verify-proof.js
```

Verifying proof... (0.4s)

Result: COMPLIANT

(model exceeds 10^{25} FLOP threshold)

Live Terminal Demo & Full Source Code

Demo Shows:

- Policy pack structure
- Proof generation (3 circuits)
- Compliance reports
- Composable proofs
- Verification (0.4s)
- Negative tests

Repository Includes:

- Working circuits
- Policy packs
- Demo scripts
- Research paper
- Documentation
- Performance tests

 https://github.com/srishtiii28/tech_ai_gov

Run the demo yourself in 5 minutes!

Key Contributions

① Working Implementation

- 3 ZK circuits for regulatory compliance
- 2 real policy packs (EU AI Act, Anthropic RSP)
- Production-ready performance (0.4s verification)

② Novel Architecture

- Composable proof system ($3\times$ efficiency)
- Systematic verifiability gap analysis
- Defense-in-depth security model

③ Honest Engineering

- Clear limitations documented
- Threat model analyzed
- Integration with complementary systems

Infrastructure for scalable, privacy-preserving AI governance

The future of AI regulation requires proving compliance without compromising competitiveness

Global Coordination

- Cross-border verification
- Mutual recognition
- Preserved sovereignty

Security

- No data exfiltration
- Cryptographic guarantees
- Minimal attack surface

Scalability

- Automated verification
- 0.4s per proof
- Thousands of models

Balance

- Regulatory oversight
- Commercial privacy
- Public trust

Next Steps

Immediate

- Pilot with regulatory authorities (AISI, NIST)
- Integration with major cloud providers
- Standardization working group

Medium-Term

- Additional policy packs (US EO 14110, UK framework)
- Federated learning circuits
- Hardware attestation integration
- Decentralized verification network

Long-Term

- International governance standard
- Continuous monitoring integration

Thank You

ZK-GovProof

Privacy-Preserving Verification of Regulatory Compliance



https://github.com/srishtiii28/tech_ai_gov

Complete implementation, circuits, policy packs, and research paper

Appendix: Circuit Implementation

Circom Code Example - Compute Threshold

```
template ComputeThreshold() {
    signal input privateCompute;
    signal input threshold;
    signal output valid;

    signal isBelow <== LessThan(252)(
        [privateCompute, threshold]
    );
    valid <== isBelow;

    // Proof only succeeds if:
    // privateCompute < threshold
    isBelow == 1;
}

component main = ComputeThreshold();
```

Appendix: Related Work

Comparison to Existing Approaches

- **zkML:** ZK machine learning inference
Our focus: governance verification, not inference
- **Blockchain governance:** On-chain voting
We enable off-chain compliance
- **Federated learning:** Privacy-preserving training
Complementary approach, not competitive
- **Secure enclaves:** Hardware-based attestation
We combine ZK + TEEs for defense-in-depth

Our Novel Contribution

First system to apply ZK proofs to regulatory compliance verification for foundation models, with real policy packs and composable architecture.

Appendix: Technical Details

Parameter	Value
Proof System	Groth16
Elliptic Curve	BN128 (alt_bn128)
Security Level	128-bit
Circuit Constraints	1,024 - 8,192
Trusted Setup	Powers of Tau + Phase 2
Verification Gas (Ethereum)	250,000 gas
Circuit Language	Circom 2.0
Library	snarkjs 0.7.x