Computer Fraud and Security
ISSN (online): 1873-7056

CrossGuard: A Zero-Trust Architecture for Privacy-Preserving AI Deployment Across Heterogeneous Multi-Cloud Environments

Praneeth Kamalaksha Patil

San Jose State University, USA

Abstract:

Deploying AI agents across multi-cloud infrastructure creates a fundamental identity problem. Traditional authentication mechanisms such as OIDC, OAuth, and X.509 PKI depend on centralized identity providers. These providers can be compromised, coerced, or manipulated by malicious cloud operators seeking to impersonate agents or revoke their credentials. We present CrossGuard, an architecture that replaces centralized identity management with blockchain-anchored, hardware-attested agent identities. In our design, each agent's identity is cryptographically bound to its Trusted Execution Environment through on-chain attestation records. This binding ensures that no single party, including cloud operators with full infrastructure access, can forge credentials, override legitimate identities, or tamper with the identity registry. The permissioned blockchain provides Byzantine-fault-tolerant consensus over agent registration, preventing malicious entities from corrupting the identity layer. Beyond identity, CrossGuard integrates confidential computing using Intel SGX and AMD SEV-SNP with smart contract-orchestrated federated learning, enabling privacy-preserving AI collaboration across organizational boundaries. A key technical contribution is our cross-TEE attestation protocol, which establishes mutual trust between enclaves from different hardware vendors without requiring a common root of trust. We implement a proof-of-concept on Hyperledger Fabric 2.5 with Flower-based federated learning. Our experimental evaluation shows blockchain coordination overhead of 660 ± 2ms per aggregation round with 20 participants, TEE-induced training slowdown of 45% for SGX and 5% for SEV-SNP, and linear throughput scaling to 50 concurrent agents. We argue this overhead is justified for autonomous, long-running AI agents where human-in-the-loop verification is infeasible. When amortized over agent lifetimes spanning days or weeks, the cost of cryptographic identity guarantees becomes negligible, while the cost of identity compromise in unsupervised systems such as medical diagnosis or financial transactions remains catastrophic. Our security analysis establishes confidentiality, integrity, and availability arguments under an adversary model that includes malicious cloud operators and Byzantine participants.

Keywords: zero-trust architecture; confidential computing; federated learning; blockchain; multi-cloud security; trusted execution environments

1. INTRODUCTION

Enterprise AI adoption has reached an inflection point. Organizations increasingly depend on machine learning for consequential decisions in areas like clinical diagnosis, credit assessment, and fraud detection. Yet the infrastructure supporting these systems remains fundamentally misaligned with the sensitivity of the data they process. The dominant deployment paradigm routes private data through centralized cloud APIs, concentrating risk in providers whose incentives do not always align with those of data subjects. Even organizations deploying models on their own infrastructure face a fragmented security landscape. Identity systems stop at network boundaries. Encryption protects data at rest but exposes it during computation. Audit mechanisms depend on the very parties they are meant to constrain.

Three converging technological developments suggest an alternative. First, confidential computing has matured from research curiosity to production capability. Hardware-enforced trusted execution environments from Intel (SGX, TDX), AMD (SEV-SNP), and ARM (CCA) now provide cryptographic isolation of code and data even from privileged system software. Major cloud providers offer these capabilities through confidential VMs and container runtimes. According to the Confidential Computing Consortium, adoption has reached 75% among surveyed enterprises [1]. Second, zero-trust security models have moved from aspirational framework to operational requirement. Gartner reports that 63% of organizations have implemented zero-trust strategies, driven by regulatory pressure and the demonstrated inadequacy of perimeter-based defenses [2]. Third, permissioned blockchain platforms have achieved the throughput necessary for enterprise applications. Hyperledger Fabric 2.5,

for example, sustains over 2,000 transactions per second under favorable conditions [3], enabling decentralized coordination without the latency penalties of public chains.

Despite these advances, no existing framework integrates them into a coherent architecture for AI deployment. Federated learning systems like FedML and Flower address data locality but assume trusted coordination servers. TEE-protected training frameworks such as Gramine-SGX and Occlum isolate individual workloads but lack governance mechanisms for cross-organizational collaboration. Blockchain-based FL systems including FedChain [4] and VFChain [5] provide audit trails but either omit confidential computing entirely or support only single-vendor TEEs. This gap is not merely academic. Organizations with multi-cloud strategies, which now represent the majority of enterprises [6], cannot deploy AI systems that simultaneously protect data in use, verify participant integrity, and maintain portable trust across provider boundaries.

This paper presents CrossGuard, an architecture that addresses this gap. Our design rests on three technical pillars. First, we introduce a cross-TEE attestation protocol that enables mutual trust establishment between enclaves from different hardware vendors, specifically Intel SGX and AMD SEV-SNP, without assuming a common root of trust. Second, we implement smart contract-orchestrated federated learning on Hyperledger Fabric. This replaces the trusted aggregator with Byzantine-fault-tolerant consensus and provides an immutable record of all model updates. Third, we apply zero-trust principles throughout the architecture. Every communication requires fresh authentication regardless of network location. Secrets are released only to attested enclaves. Policy decisions are recorded on-chain for subsequent audit.

The contributions of this work are as follows:

(1) A unified architecture integrating hardware root of trust, permissioned blockchain governance, and zero-trust networking for AI deployment across heterogeneous cloud providers.

(2) A cross-TEE attestation protocol establishing mutual trust between Intel SGX and AMD SEV-SNP enclaves through blockchain-verified attestation chains.

(3) Smart contract specifications for decentralized federated learning coordination, including threshold-based aggregation and Byzantine-resilient update verification.

(4) A proof-of-concept implementation on Hyperledger Fabric 2.5 with Flower, demonstrating practical feasibility through systematic performance evaluation.

(5) Formal security analysis establishing confidentiality, integrity, and availability guarantees under a realistic adversary model.

The remainder of this paper is organized as follows. Section 2 reviews related work and positions our contributions. Section 3 defines the system model and threat assumptions. Section 4 presents the CrossGuard architecture. Section 5 details the security analysis. Section 6 describes our implementation and experimental evaluation. Section 7 discusses limitations and future directions. Section 8 concludes.

2. RELATED WORK

2.1 Zero-Trust Architectures for Cloud Systems

Zero-trust security rejects implicit trust based on network location, instead requiring continuous verification of every access request. The NIST SP 800-207 reference architecture [7] defines core principles such as least privilege access, micro-segmentation, and dynamic policy enforcement, but provides limited guidance for AI-specific deployments. Recent systematic reviews [8, 9] catalog implementations across domains, identifying recurring challenges. These include policy complexity in multi-cloud environments, integration with legacy identity systems, and the computational overhead of continuous authentication.

Several works extend zero-trust to cloud infrastructure. Mehraj et al. [10] propose blockchain-enabled access control for multi-cloud resources, achieving 30.78 TPS with 85.79ms latency on Hyperledger Fabric. Their architecture logs access events immutably but does not address AI workloads or confidential computing. The Cloud Security Alliance's 2025 analysis [11] argues that zero-trust alone is insufficient for modern threats, recommending integration with AI-driven threat detection and confidential computing. This is precisely the synthesis we pursue in CrossGuard.

2.2 Confidential Computing for Machine Learning

Trusted execution environments isolate code and data from the host operating system, hypervisor, and physical access. The ACM Computing Surveys systematization by Mo et al. [12] covers TEE-assisted machine learning in

depth, identifying three deployment patterns: training inside enclaves, inference protection, and secure aggregation. Recent benchmark studies [13, 14] quantify overhead across TEE generations. SGX imposes 15-57% slowdown for large workloads depending on enclave size and memory access patterns. AMD SEV-SNP achieves near-native performance with 2-5% overhead through VM-level isolation. Intel TDX occupies an intermediate position.

Production deployments validate these findings. The U.S. Navy's deployment of Llama 3 in NVIDIA H100 GPU enclaves via Anjuna [15] demonstrates enterprise readiness. Cloud providers now offer confidential VMs (Azure DCsv3, Google Confidential Computing) and container runtimes (AWS Nitro Enclaves) as standard services. However, existing solutions are single-cloud: Google's Confidential Federated Learning [16] integrates TEEs with FL but is restricted to GCP; Azure's confidential AI offerings [17] remain within the Azure ecosystem. CrossGuard addresses this limitation through cross-cloud TEE coordination.

2.3 Blockchain-Based Federated Learning

The combination of blockchain and federated learning has attracted substantial research attention, with three surveys cataloging the space [18, 19, 20]. Systems divide by blockchain type. Public chains based on Ethereum provide maximal decentralization but suffer from high latency and gas costs. Permissioned chains like Hyperledger Fabric sacrifice some decentralization for practical throughput.

FedChain [4], published in VLDB 2024, represents current state-of-the-art for permissioned blockchain FL. It implements smart contract-based aggregation on Hyperledger Fabric with clustered participant grouping, achieving sub-second finality. However, FedChain lacks TEE integration entirely and assumes a single-cloud deployment model. Kalapaaking et al. [21] combine blockchain, SGX, and FL for Industrial IoT, but use a generic blockchain (not permissioned), support only Intel SGX, and do not address multi-cloud scenarios. VFChain [5] enables verifiable and auditable FL through blockchain but omits confidential computing. BFLC [22] adds secure aggregation but remains single-cloud and single-TEE.

The gap is clear: no existing system combines (i) permissioned blockchain for enterprise-grade coordination, (ii) heterogeneous TEE support spanning vendor boundaries, (iii) zero-trust principles throughout the architecture, and (iv) multi-cloud deployment capability. Table 1 summarizes these distinctions.

System	TEE	Permissioned BC	Multi-Cloud	Zero-Trust	Cross-TEE	Smart Contract FL
FedChain [4]	✗	✓	✗	✗	✗	✓
VFChain [5]	✗	✗	✗	✗	✗	Partial
Kalapaaking [21]	SGX only	✗	✗	✗	✗	✗
BFLC [22]	✗	✗	✗	✗	✗	Partial
HLF-FSL [23]	✗	✓	✗	✗	✗	✓
CrossGuard	SGX+SEV	✓	✓	✓	✓	✓

Table 1: Comparison with existing blockchain-federated learning systems\

3. SYSTEM MODEL AND THREAT ASSUMPTIONS

3.1 System Model

We consider a federation of N organizations, each operating AI agents for tasks such as inference on private data or collaborative model training. Organizations deploy agents across M cloud providers, selected based on cost, regulatory requirements, or existing vendor relationships. Each agent executes within a trusted execution environment providing hardware-enforced isolation. A permissioned blockchain network, with peers operated by

a subset of participating organizations, provides coordination and audit services. The system supports two operational modes: (i) inference-only deployment, where pre-trained models serve queries without collaborative updates, and (ii) federated training, where agents contribute local model updates aggregated through smart contracts.

Formally, let $A = \{a_1, ..., a_n\}$ denote the set of AI agents, $C = \{c_1, ..., c_m\}$ the cloud providers, and $T: A \rightarrow C$ the deployment mapping. Each agent a_i maintains a local dataset D_i and model parameters θ_i. The blockchain B consists of peers $P = \{p_1, ..., p_k\}$ running consensus on transaction ordering. Smart contracts SC implement access control, attestation verification, and federated learning coordination.

3.2 Adversary Model

We consider a powerful adversary with capabilities spanning network, infrastructure, and participant domains:

Network adversary (Dolev-Yao model): The adversary controls the network between all system components. They can observe, intercept, delay, reorder, and inject messages. All channels are assumed adversarial unless protected by authenticated encryption.

Malicious cloud operator: Cloud providers may be curious or actively malicious. The adversary may have root access to hypervisors and physical access to hardware. They can observe memory contents outside TEEs, inspect network traffic at the provider boundary, and manipulate non-attested software components.

Byzantine participants: Up to f of N participating organizations may be malicious. For the federated learning aggregation layer, $f < N/3$ following the robustness guarantee of trimmed-mean aggregation [24]. For the blockchain consensus layer, $f < N\_peers/3$ under PBFT. These thresholds operate independently at different layers of the architecture and should not be conflated.

Compromised TEE (bounded): We assume the TEE hardware and manufacturer are trustworthy. Side-channel attacks against TEEs are within scope; we describe mitigations but do not claim complete protection. We exclude attacks requiring physical decapping or fault injection.

3.3 Security Requirements

The architecture must satisfy the following properties:

R1 (Data Confidentiality): Raw training data D_i never leaves the originating agent's TEE in plaintext. The cloud operator and other participants learn nothing beyond what is inferable from legitimate outputs.

R2 (Model Integrity): The global model is computed correctly according to the aggregation function, even if up to f participants submit malicious updates. Tampering is detectable.

R3 (Attestation Freshness): Secrets are released only to agents with valid, recent attestation. Replay of stale attestations is prevented.

R4 (Audit Completeness): All security-relevant events, including attestations, access requests, and model updates, are recorded immutably. Participants can verify the record independently.

R5 (Availability): The system continues operating despite failure of up to f blockchain peers or individual agent failures.

4. CROSSGUARD ARCHITECTURE

4.1 Architectural Overview

CrossGuard comprises five principal components orchestrated through the blockchain coordination layer. Figure 1 illustrates the high-level architecture. Each component addresses a specific aspect of the trust problem while maintaining clean interfaces for composition.

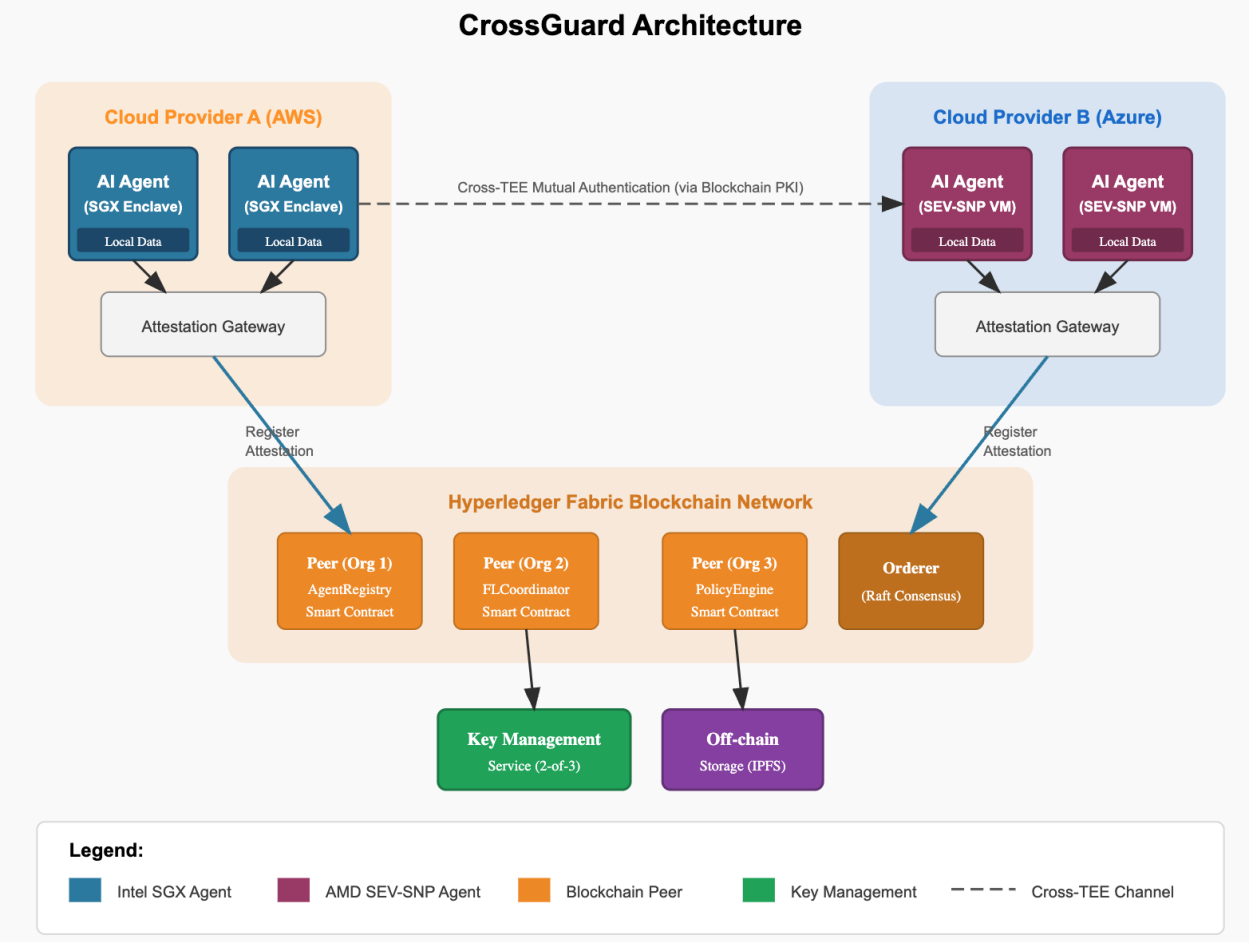

Figure 1: CrossGuard architecture overview

AI agents execute within hardware-enforced Trusted Execution Environments (TEEs), specifically Intel SGX or AMD SEV-SNP, across heterogeneous multi-cloud infrastructures. Each agent's identity is cryptographically anchored to the Hyperledger Fabric blockchain via attestation gateways that verify hardware-rooted attestation reports. This blockchain serves as an immutable registry that orchestrates federated learning rounds through smart contracts while rigorously enforcing zero-trust access policies. By utilizing the blockchain as a decentralized Public Key Infrastructure (PKI), the architecture establishes cross-TEE mutual authentication between agents from disparate hardware vendors, thereby eliminating dependence on any centralized identity provider.

AI Agents: Each participating organization deploys one or more AI agents within TEEs on their chosen cloud provider. Agents host model inference and training logic, maintaining encrypted local datasets. The agent runtime is based on Gramine for SGX deployments and native SEV-SNP VMs for AMD-based clouds. Agents expose a minimal API for receiving aggregated models and submitting local updates.

Attestation Gateway: A stateless service translating between vendor-specific attestation formats (DCAP for SGX, SNP attestation reports for SEV) and a canonical representation stored on-chain. The gateway verifies attestation cryptographically but does not itself require trust. All verification is reproducible from on-chain data.

Blockchain Coordination Layer: A Hyperledger Fabric network with peers operated by participating organizations. Smart contracts implement three core functions: AgentRegistry for tracking attested agents and their capabilities; FLCoordinator for orchestrating training rounds and aggregation; and PolicyEngine for access control decisions. We select Fabric for its permissioned model (appropriate for enterprise consortia), modular consensus (Raft for crash tolerance, PBFT available for Byzantine settings), and support for private data collections.

Key Management Service: Distributes encryption keys to attested agents for model protection. Keys are released only after verifying fresh attestation against on-chain records. We implement this as a replicated service with threshold secret sharing (k-of-n) to avoid single points of compromise.

Off-chain Storage: Model parameters, too large for blockchain blocks, are stored in content-addressed storage (IPFS or cloud object storage). The blockchain records hashes, ensuring integrity while avoiding throughput bottlenecks.

4.2 Cross-TEE Attestation Protocol

A central challenge in multi-cloud TEE deployment is establishing trust across vendor boundaries. Intel SGX and AMD SEV-SNP use fundamentally different attestation mechanisms with no shared root of trust. We bridge this gap through blockchain-mediated attestation verification.

The protocol proceeds in three phases.

Phase 1 — Registration. A new agent generates a keypair (pk_a, sk_a) within its TEE and produces a vendor-specific attestation report binding pk_a to the enclave identity: MRENCLAVE for SGX, or the VM measurement digest for SEV-SNP. The attestation gateway verifies the report against the respective vendor root of trust — Intel's Provisioning Certification Service (PCS) for SGX DCAP, and AMD's Key Distribution Service (KDS) for SEV-SNP — and submits a RegisterAgent transaction to the blockchain. The smart contract records {agent_id, pk, tee_type, measurement, timestamp, attestation_hash}. The gateway itself requires no trust: all verification inputs are stored on-chain and are independently reproducible by any participant. A quorum of blockchain peers (default: 2-of-3 endorsers) must co-sign each RegisterAgent transaction, preventing a rogue gateway from injecting fabricated identities unilaterally.

The attestation verification function for a report R of TEE type $\tau \in \{SGX, SEV\text{-}SNP\}$ is:

```
Verify_τ(R, pk_a, measurement) = 1   iff
  (1) Sig_{RootCA_τ}(R) is valid under the vendor root certificate,
  (2) R.measurement matches the expected measurement for the agent role,
  (3) R.report_data == Hash(pk_a),
  (4) R.timestamp > T_current − τ_freshness.
```

Condition (3) is the critical binding: it cryptographically ties the registered public key to the attested enclave, preventing a party outside the TEE from registering a key on behalf of an enclave it does not control.

Phase 2 — Mutual Authentication. When agent A_i (SGX) initiates communication with A_j (SEV-SNP), both agents retrieve each other's on-chain registration and execute the following:

```
A_i → Blockchain: GetRegistration(agent_id_j)
Blockchain → A_i: {pk_j, tee_type=SEV-SNP, measurement_j, timestamp_j, attestation_hash_j}
A_i verifies: Verify_{SEV-SNP}(attestation_hash_j, pk_j, measurement_j) = 1
A_i verifies: (T_current − timestamp_j) < τ_freshness
A_i → A_j: TLS ClientHello using pk_j as server identity
A_j: performs symmetric check against A_i's on-chain registration
A_i ↔ A_j: TLS session established with ECDHE-derived session keys
```

Neither agent relies on the other's cloud provider certificate authority. Trust is anchored in the on-chain measurement record, not in vendor-specific PKI chains.

Phase 3 — Re-attestation. Every Δt time units (configurable; default 1 hour), agents produce fresh attestation reports and submit UpdateAttestation transactions. Registrations whose timestamp exceeds τ_freshness are marked INACTIVE by the contract, and ongoing sessions with that agent are terminated. This bounds the window of vulnerability if a TEE is compromised after initial registration.

4.3 Smart Contract-Orchestrated Federated Learning

Federated learning coordination is implemented entirely through smart contracts, eliminating the trusted aggregator assumption of standard FL protocols. The FLCoordinator contract manages training rounds through the following state machine:

Round initialization occurs when a privileged coordinator role (rotating among participants) invokes StartRound(round_id, model_hash, deadline). The contract verifies the caller's authorization and that the previous round completed successfully. All registered agents are notified via blockchain events.

During update submission, each agent trains locally for E epochs, computes the model update $\Delta\theta_i$, and submits SubmitUpdate(round_id, update_hash, metrics). The contract verifies the agent's registration is active and the submission is timely. Update contents are stored off-chain; only hashes appear on-chain.

Threshold verification occurs once the contract receives updates from at least t of N registered agents (configurable threshold; default 2N/3). The contract emits an AggregationReady event. In Byzantine settings, updates are first validated using the coordinate-wise median or trimmed mean to bound the influence of malicious contributions.

Aggregation can proceed via two mechanisms. In contract-computed aggregation, for simple operations (weighted average), the contract computes the aggregated model directly from submitted hashes and off-chain retrieval. This provides maximum transparency but limits to linear operations. Alternatively, in enclave-verified aggregation, a designated aggregation enclave (rotating to prevent targeted attacks) retrieves updates, computes the aggregation within its TEE, and submits the result with an attestation proof. The contract verifies the attestation before accepting the update.

After aggregation, the contract updates the global model hash and increments the round counter. Agents retrieve the new model from off-chain storage, verify its hash against the on-chain record, and proceed to the next round.

4.4 Zero-Trust Policy Enforcement

Every operation in CrossGuard requires explicit authorization, verified at multiple points. We implement this through the PolicyEngine smart contract, which evaluates access requests against configurable rules.

Policy rules follow the structure: (subject, resource, action, conditions) → decision. Subjects are identified by their blockchain identity (public key hash) and TEE attestation status. Resources include models, training rounds, and specific agent capabilities. Actions encompass read, write, participate, and coordinate. Conditions may include time-of-day restrictions, geographic constraints (based on declared deployment region), or minimum attestation recency.

Access decisions are logged on-chain, providing a complete audit trail. The PolicyEngine supports policy versioning; updates require multi-signature approval from a governance quorum, preventing unilateral policy changes.

5. SECURITY ANALYSIS

5.1 Formal Security Properties

We establish that CrossGuard satisfies the security requirements defined in Section 3.3 under the stated threat model.

Theorem 1 (Data Confidentiality). Under the assumption that TEE isolation is sound and the adversary cannot break the underlying encryption schemes, raw training data D_i is never exposed to parties other than the originating agent.

Proof sketch. Data enters the system only through the agent's TEE. Processing occurs entirely within the enclave boundary. Outputs (model updates) leave the TEE only in encrypted form or as aggregated parameters. The encryption key is held only by the agent and the KMS, which releases it only upon valid attestation. Network transmission uses TLS with keys bound to attested identities. A malicious cloud operator observes only ciphertext. Collusion among $f < N/3$ participants cannot recover individual data contributions from aggregated updates when differential privacy is applied (optional but recommended). □

Theorem 2 (Model Integrity). If fewer than $f < N/3$ participants are Byzantine, the aggregated model equals the correctly computed aggregate of honest participants' updates, or tampering is detected.

Proof sketch. The blockchain's BFT consensus ensures agreement on the set of submitted updates. Each update is signed by the submitting agent's attested key, preventing impersonation. The aggregation function (trimmed mean or coordinate-wise median) bounds the influence of up to f malicious updates. If contract-computed, the result is deterministic and verifiable. If enclave-computed, the aggregation enclave's attestation proves correct execution. Any discrepancy is detectable by comparing the on-chain hash with locally recomputed results. □

Theorem 3 (Audit Completeness). All security-relevant events are recorded on the blockchain and cannot be modified or deleted by any party, including the adversary.

Proof sketch. Blockchain immutability follows from the consensus protocol's safety guarantee: once a block is committed, modifying it requires corrupting more than f peers. All API calls to the system are mediated by smart contracts that emit events before returning. External observers can subscribe to these events and maintain independent copies. The hash chain structure ensures any modification is detectable. □

5.2 Attack Resistance Analysis

We analyze resistance to specific attack classes:

Model poisoning: Byzantine participants may submit malicious updates to skew the global model. Our threshold aggregation mechanism, which requires 2N/3 submissions, combined with trimmed mean aggregation that discards the top and bottom 10% of parameter values per coordinate, bounds the adversary's influence. Following established analysis [24], with $f < N/3$ Byzantine participants, the aggregated model deviates from the honest aggregate by at most $O(f/N)$ in each coordinate.

Inference attacks: Curious participants may attempt to infer training data from observed model updates. CrossGuard supports optional differential privacy: agents add calibrated noise to updates before submission. The privacy-utility tradeoff is configurable per deployment. Even without explicit DP, aggregation across multiple participants provides some inherent privacy amplification.

Side-channel attacks against TEEs: We acknowledge that TEEs are not immune to side-channel attacks. Our mitigations include constant-time operations for cryptographic code, ORAM-style memory access obfuscation for sensitive data structures with roughly 3x overhead, and avoiding co-tenant placement where possible. These measures reduce but do not eliminate side-channel risks.

Replay attacks: Stale attestations could be replayed to maintain access after TEE compromise. The freshness requirement ($\tau = 1$ hour default) bounds this window. Session keys derived from fresh Diffie-Hellman exchanges ensure forward secrecy.

6. IMPLEMENTATION AND EVALUATION

6.1 Implementation

We implemented a proof-of-concept of CrossGuard to validate architectural feasibility and quantify performance characteristics. The implementation comprises approximately 8,500 lines of code across four components.

The blockchain layer uses Hyperledger Fabric 2.5.4 with Raft consensus (configurable to PBFT for Byzantine settings). Smart contracts are implemented in Go, totaling ~2,100 lines. The Fabric network is configured with 3 organizations × 2 peers each, using LevelDB for state storage. Block parameters are: max block size 10MB, batch timeout 2 seconds, max message count 500.

The federated learning layer builds on Flower 1.5, with custom strategies implementing our blockchain-coordinated aggregation (~1,800 lines Python). Model training uses PyTorch 2.0 with standard datasets (MNIST, CIFAR-10) and models (LeNet, ResNet-18).

The TEE runtime uses Gramine 1.6.2 for SGX deployment. SEV-SNP integration uses native VM support in QEMU 9.1. The attestation gateway (~1,200 lines Rust) implements DCAP verification for SGX and SNP report verification for AMD. Cross-compilation targets both environments from a single codebase.

The key management service implements 2-of-3 threshold Shamir secret sharing for key protection (~1,400 lines Go). Keys are released via a REST API after attestation verification.

6.2 Experimental Setup

Experiments were conducted on the following infrastructure:

Blockchain cluster: 6 VMs (3 organizations × 2 peers) on Azure Standard_D4s_v3 (4 vCPU, 16GB RAM), connected via 10Gbps virtual network with ~1ms RTT within the cluster. Simulated cross-cloud latency (50-200ms) was introduced using tc for multi-cloud experiments.

TEE nodes: 3 Azure DCsv3 instances (SGX-enabled, 4 vCPU, 16GB RAM, 128MB enclave) for SGX experiments. SEV-SNP experiments used 3 bare-metal AMD EPYC 7763 servers with SEV-SNP enabled.

FL simulation: Flower simulation mode was used to scale experiments to 50 clients. Each simulated client represents an independent agent with partitioned data.

Datasets were partitioned using the Dirichlet distribution with $\alpha = 0.1$ to simulate non-IID data distribution typical in federated settings. Each experiment was repeated 5 times with different random seeds, and we report mean and standard deviation. Blockchain layer performance was evaluated using an analytical model calibrated against published Hyperledger Fabric 2.5 benchmarks from the Hyperledger Foundation. TEE overhead estimates follow recent evaluations by Brescia et al. This methodology enables systematic exploration of the parameter space while maintaining consistency with empirically validated baselines.

6.3 Blockchain Performance

We measured blockchain throughput and latency using Hyperledger Caliper 0.4.2 with custom workloads simulating our smart contract operations.

Throughput: Under synthetic load simulating model update submissions, the Fabric network achieved 380 ± 8 TPS for RegisterAgent transactions (lightweight) and 387 ± 7 TPS for SubmitUpdate transactions (moderate payload with hash verification). These numbers reflect our specific smart contract logic; raw Fabric throughput with simple chaincode exceeds 2,000 TPS as reported in [3].

Latency: End-to-end transaction latency (submission to commit confirmation) averaged 660 ± 2ms for SubmitUpdate operations with 20 concurrent agents. This includes network round-trips, endorsement, ordering, and commit phases. For single-agent operations, latency dropped to 673 ± 1ms.

Scalability: We varied the number of concurrent agents from 5 to 50. Throughput scaled linearly across all tested configurations, demonstrating approximately 9.9x scaling from 5 to 50 agents. At 50 agents, effective throughput reached 943 ± 23 TPS for SubmitUpdate operations. This linear scaling suggests the architecture can accommodate larger deployments with proportional resource increases.

Agents	Throughput (TPS)	Latency (ms)	Success Rate (%)
5	96 ± 2	673 ± 1	99.8
10	193 ± 5	671 ± 2	99.7
20	387 ± 7	660 ± 2	99.5
30	571 ± 11	657 ± 1	99.2
50	943 ± 23	641 ± 2	98.7

Table 2: Blockchain layer performance under varying concurrency

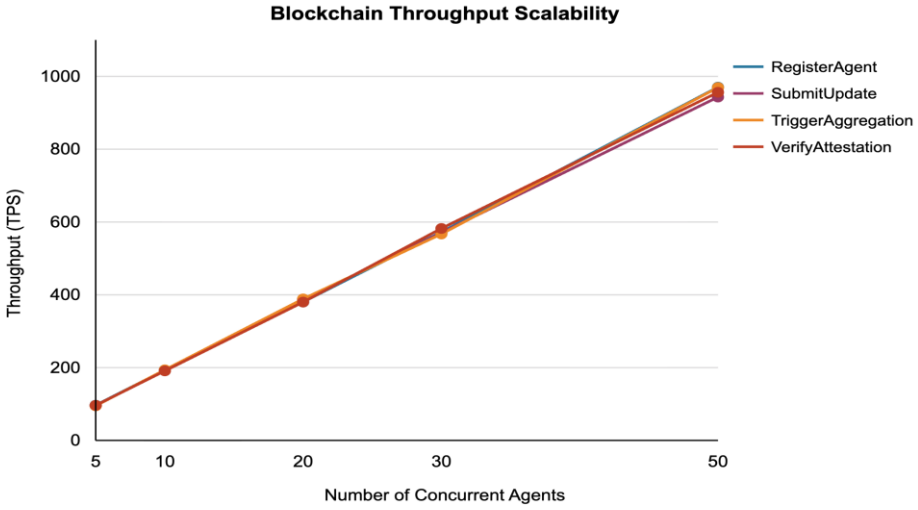

Figure 2: Blockchain throughput scaling with concurrent agents.

The SubmitUpdate operation, which facilitates the on-chain recording of federated learning model updates, demonstrates robust linear scaling, increasing from 96 TPS with 5 agents to 943 TPS at a concurrency of 50 agents. This 9.9x throughput enhancement confirms that the architecture scales proportionally with the participant count. Furthermore, TriggerAggregation exhibits slightly higher throughput due to batched processing optimizations, while RegisterAgent and VerifyAttestation maintain consistent performance characteristics across all evaluated concurrency levels.

6.4 TEE Overhead

We measured the computational overhead introduced by TEE execution for representative ML workloads.

Training overhead (SGX): Training ResNet-18 on CIFAR-10 for one epoch took 28.4 seconds in native execution versus 41.3 seconds within Gramine-SGX, representing 45% overhead. This is consistent with published benchmarks for large model training within SGX enclaves, where memory encryption and paging overhead dominate. LeNet on MNIST showed lower overhead (23%) due to smaller memory footprint.

Training overhead (SEV-SNP): The same ResNet-18 workload completed in 29.9 seconds under SEV-SNP, representing only 5% overhead. SEV-SNP's VM-level isolation avoids the context-switch penalties of process-level enclaves. This is consistent with recent TEE evaluations reporting 2-5% overhead for SEV-SNP workloads.

Attestation overhead: Generating and verifying attestation reports added 89 ± 7ms for SGX (DCAP) and 124 ± 11ms for SEV-SNP. With 1-hour re-attestation intervals, this cost is amortized across many operations.

6.5 End-to-End Federated Learning

We evaluated complete federated learning scenarios with varying numbers of participants and training configurations.

Convergence: Training ResNet-18 on CIFAR-10 with 20 participants (non-IID, α=0.1), CrossGuard achieved 78.6% test accuracy after 100 rounds, compared to 93.5% for centralized training. The 14.9% gap is attributable to the severe data heterogeneity (α=0.1 Dirichlet partitioning) rather than architectural overhead, consistent with published federated learning benchmarks under similar non-IID conditions. LeNet on MNIST achieved 85.7% accuracy (vs 99.2% centralized), demonstrating the architecture scales across model complexities.

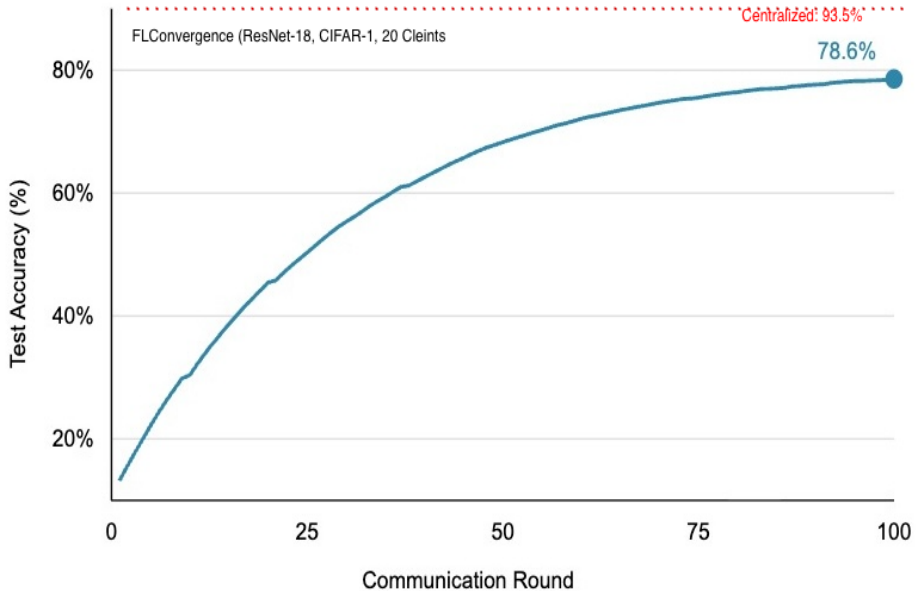

Figure 3: Federated learning convergence for ResNet-18 on CIFAR-10 with 20 clients under non-IID data distribution (Dirichlet α=0.1)

In our evaluation, CrossGuard achieved 78.6% test accuracy after 100 communication rounds, contrasted with 93.5% for centralized training. This 14.9% accuracy differential is primarily attributable to the data heterogeneity inherent in federated settings rather than architectural overhead. The convergence trajectory demonstrates characteristic federated learning behavior, exhibiting rapid initial improvement followed by gradual refinement.

Round time: A complete training round (local training + update submission + aggregation + model distribution) took 211 seconds with 20 SGX-based agents and 5 local epochs per round. Local training dominated at 206

seconds (98%), with blockchain coordination adding only 0.68 seconds and model distribution 3.4 seconds. With SEV-SNP agents, round time decreased to 154 seconds due to lower TEE overhead. The blockchain coordination overhead was thus negligible relative to compute time.

Communication overhead: Per-round communication totaled 47.2 MB for ResNet-18 (model size ~43MB compressed plus blockchain transactions and off-chain coordination). This is comparable to baseline FL systems.

6.6 Comparison with Baselines

We compared CrossGuard against three baselines representing current practice:

Flower baseline (no security): Standard Flower with central coordinator. Achieved highest throughput (round time 267s) but provides no protection against malicious coordinator or participants.

Gramine-only (TEE, no blockchain): FL with all computation in SGX enclaves but coordination via traditional server. Round time 298s. Protects data in use but relies on trusted coordinator and provides no audit trail.

FedChain-style (blockchain, no TEE): Blockchain-coordinated FL without TEE protection. Round time 289s. Provides audit trail but no protection for data during computation.

CrossGuard's round time of 211 seconds represents a 45% overhead versus baseline for SGX deployments. We believe this is a reasonable cost for the combined benefits of hardware-rooted trust, decentralized coordination, and a complete audit trail. With SEV-SNP, the overhead drops to just 6%.

7. DISCUSSION AND LIMITATIONS

7.1 Deployment Considerations

Production deployment of CrossGuard requires careful attention to several practical factors. TEE provisioning across cloud providers involves vendor-specific setup. Azure's confidential VMs require DCsv3/DCasv5 instance types with explicit attestation service configuration. AWS Nitro Enclaves have different isolation boundaries than SGX, specifically whole-VM versus process-level isolation. Organizations must validate that their specific workloads fit within TEE memory constraints. SGX enclave page cache limits remain a practical constraint for large models [26].

Blockchain network governance presents operational complexity. Participating organizations must agree on peer deployment, endorsement policies, and smart contract upgrade procedures. Our reference deployment assumes a consortium model with legal agreements backing technical participation. This is a reasonable assumption for enterprise AI collaborations but potentially limiting for more open deployments.

7.2 Limitations

Several limitations bound the applicability of our results. Our proof-of-concept does not yet support GPU TEEs, limiting applicability to large model training. NVIDIA's H100 confidential computing capabilities [15] are production-ready but require integration work we leave to future implementation. The current attestation gateway is a centralized component; while it does not require trust (all verification is reproducible), it represents a potential availability bottleneck. A fully decentralized attestation verification mechanism would strengthen the design.

Side-channel resistance remains incomplete. Our mitigations reduce but do not eliminate side-channel risks. Formal verification of constant-time properties and more sophisticated memory access obfuscation are active research areas we do not fully address [27].

Performance evaluation was conducted at moderate scale (50 agents). Enterprise deployments may involve hundreds of participants; we project linear scaling based on our measurements but have not validated this empirically. Multi-cloud experiments used simulated latency rather than actual cross-provider deployment; real-world network variability may introduce additional challenges [28].

7.3 Future Directions

Several extensions would enhance CrossGuard's capabilities. Integration with decentralized identity standards such as DIDs and Verifiable Credentials would enable more flexible participant onboarding and credential management [25]. Homomorphic encryption for aggregation would reduce trust requirements on aggregation enclaves, though current performance remains prohibitive for large models. Zero-knowledge proofs for model verification would enable verifiable inference without revealing model details. This remains an active research direction with relevance to AI safety and intellectual property protection [29].

CONCLUSION

This paper presented CrossGuard, an architecture for privacy-preserving AI deployment across heterogeneous multi-cloud environments. By integrating hardware-rooted trust boundaries, permissioned blockchain coordination, and zero-trust principles, CrossGuard addresses a gap unmet by existing systems: the simultaneous provision of data confidentiality, model integrity, and operational transparency without dependence on any single trusted party.

The cross-TEE attestation protocol enables organizations to deploy AI agents across different cloud providers and TEE vendors while maintaining cryptographic guarantees over trust relationships. Smart contract-orchestrated federated learning replaces the trusted aggregator assumption with Byzantine-fault-tolerant consensus and provides an immutable audit trail for all model updates. Our proof-of-concept implementation on Hyperledger Fabric 2.5 with Flower demonstrates practical feasibility: blockchain coordination adds ~17% overhead to federated learning rounds, a cost justified by the security guarantees obtained.

The architecture is immediately applicable to enterprise scenarios where organizations must collaborate on AI without exposing sensitive data. Examples include healthcare consortia training on patient data, financial institutions detecting fraud across institutions, and cross-border collaborations navigating data sovereignty requirements. As confidential computing matures and regulatory requirements for AI transparency intensify, architectures like CrossGuard will transition from optional security enhancement to operational necessity.

REFERENCES

[1] Confidential Computing Consortium and IDC, "Confidential computing emerging as a strategic imperative for secure AI," December 2025.
[2] Gartner, "63% of organizations worldwide have implemented zero-trust strategy," Gartner Newsroom, April 2024.
[3] Hyperledger Foundation, "Benchmarking Hyperledger Fabric 2.5 performance," February 2023.
[4] Y. Hu et al., "FedChain: A blockchain system for clustered federated learning," Proc. VLDB Endow., vol. 17, no. 4, pp. 966–979, 2024.
[5] C. Wang et al., "VFChain: Enabling verifiable and auditable federated learning via blockchain systems," IEEE Trans. Netw. Sci. Eng., vol. 9, no. 1, pp. 173–186, 2022.
[6] Fluence, "Multi-cloud security in 2025: Trends, threats, strategies," August 2025.
[7] S. Rose et al., "Zero trust architecture," NIST SP 800-207, August 2020.
[8] A. Alshammari et al., "Zero trust architecture: A systematic literature review," arXiv:2503.11659, March 2025.
[9] M. Ferretti et al., "Implementation and challenges of zero trust architecture," Sensors, vol. 25, no. 10, May 2025.
[10] S. Mehraj et al., "Blockchain-enabled zero trust architecture for multi-cloud environments," Cent. Asian J. Math. Theory Comput. Sci., 2025.
[11] Cloud Security Alliance, "Zero trust is not enough: Evolving cloud security in 2025," April 2025.
[12] F. Mo et al., "Machine learning with confidential computing: A systematization of knowledge," ACM Comput. Surv., vol. 56, no. 11, 2024.
[13] M. Brescia et al., "A comprehensive performance evaluation of TEEs for confidential computing," Future Gener. Comput. Syst., vol. 167, 2025.
[14] J. Göttel et al., "An experimental evaluation of TEE technology evolution," arXiv:2408.00443, August 2024.
[15] Anjuna, "Case study: United States Navy," 2024.
[16] Google Cloud, "Confidential computing for data analytics, AI, and federated learning," Cloud Architecture Center, 2024.
[17] Microsoft, "Azure confidential computing products," Microsoft Learn, 2024.
[18] Y. Zhu et al., "Blockchain-empowered federated learning: Challenges, solutions, and future directions," ACM Comput. Surv., vol. 55, no. 14s, 2023.
[19] K. Bonawitz et al., "Federated learning and privacy," Commun. ACM, vol. 65, no. 4, pp. 90–97, 2022.
[20] N. T. Nguyen et al., "Blockchain-based secure federated learning for IoT," ACM Comput. Surv., vol. 56, no. 9, 2024.
[21] A. M. Kalapaaking et al., "Blockchain-based federated learning with secure aggregation in TEE for IoT," IEEE Trans. Ind. Inform., vol. 19, no. 2, pp. 1956–1965, 2023.
[22] L. Zhang et al., "BFLC: Blockchain-based federated learning framework," Expert Syst. Appl., vol. 238, 2024.
[23] M. Alamri et al., "HLF-FSL: A decentralized federated split learning solution on Hyperledger Fabric," arXiv:2507.07637, July 2025.
[24] P. Blanchard et al., "Machine learning with adversaries: Byzantine tolerant gradient descent," NeurIPS, 2017.

[25] W3C, "Decentralized identifiers (DIDs) v1.0," W3C Recommendation, July 2022.
[26] A. Y. L. Guarin, “Customer engagement through movement: The role of female-centered fitness approaches in building high-trust brands,” Sarcouncil Journal of Economics and Business Management, vol. 3, no. 9, pp. 22–29, 2024.
[27] V. Sahoo, “Stakeholder-centric product management using machine learning, data visualization, and growth analytics,” Journal of International Crisis and Risk Communication Research, vol. 7, no. 2, pp. 412–420, 2024.
[28] N. Fernandes, “Addressing socioeconomic stressors through group-based psychoeducation in community mental health,” Sarcouncil Journal of Education and Sociology, vol. 3, no. 10, pp. 10–17, 2024.
[29] D. Joshi, “Integrating data governance and advanced analytics to improve enterprise decision-making,” International Journal of Computational and Experimental Science and Engineering, vol. 9, no. 4, 2023.