SHAKUDO

Securing Federated Learning with Privacy Preserving Techniques

How Differential Privacy and Homomorphic Encryption Enable Compliant AI

January 7, 2026

White Paper

Table of Contents

Executive Summary

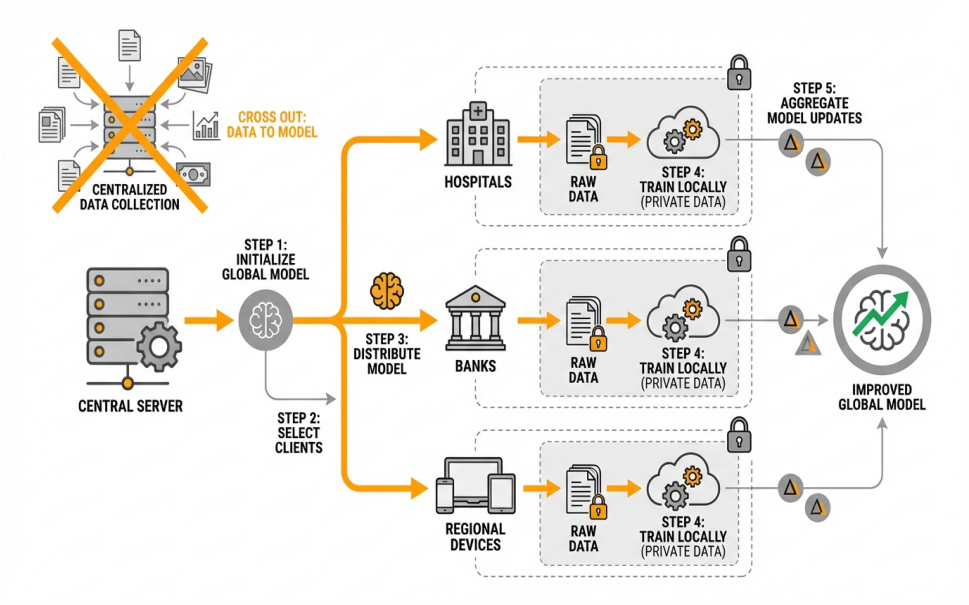
As enterprises race to deploy AI systems, privacy regulations are tightening globally. Gartner predicts 75% of the world's population will have personal data protected under privacy laws by 2026, forcing organizations to rethink how they train machine learning models. Federated Learning has emerged as a breakthrough approach, enabling collaborative model training across distributed data sources without centralizing sensitive information. However, FL alone is not inherently secure—recent research reveals vulnerabilities to gradient leakage, membership inference, and model poisoning attacks that can compromise privacy guarantees.

This whitepaper examines how two complementary techniques—Differential Privacy and Homomorphic Encryption—transform Federated Learning from a privacy-conscious framework into a truly secure AI training methodology. Differential Privacy provides mathematical guarantees that individual data contributions remain protected even when model updates are shared, while Homomorphic Encryption enables computations on encrypted data, ensuring updates remain unreadable during transmission and aggregation. For enterprises in regulated industries like healthcare, finance, and government, these techniques enable AI innovation while maintaining data sovereignty, meeting compliance requirements, and protecting competitive advantages. Organizations that master privacy-preserving AI will unlock collaborative opportunities across subsidiaries, partners, and jurisdictions that were previously impossible due to regulatory constraints.

Overview

Federated Learning represents a fundamental shift in how machine learning models are trained. Traditional approaches require consolidating data from multiple sources into a central repository—a data lake or warehouse—where algorithms process the aggregated information to build predictive models. This centralization creates significant privacy risks, regulatory challenges, and competitive concerns. A single breach exposes all consolidated data. Cross-border data transfers violate GDPR and similar regulations. Competitors hesitate to share proprietary information even when collaboration would benefit all parties.

Federated Learning solves these problems by inverting the training paradigm. Instead of moving data to the model, FL brings the model to the data. The process involves five critical steps: a central server initializes a global model, selects participating clients (devices, hospitals, banks, or regional offices), distributes the model to these clients, allows each to train locally on their private data, and then aggregates only the model updates—not the raw data—to improve the global model. This architecture keeps sensitive information on local devices or within organizational boundaries while still enabling collaborative learning from diverse datasets.



Federated Learning inverts the traditional paradigm by bringing models to data rather than centralizing sensitive information.

The approach has gained momentum as privacy regulations proliferate and organizations recognize the limitations of centralized data collection. Healthcare networks can build diagnostic models across hospitals without violating HIPAA. Financial institutions can collaborate on fraud detection without exposing customer transactions. Multinational corporations can train models across regional subsidiaries without violating data residency requirements. However, the promise of privacy-preserving AI faces a critical challenge: Federated Learning workflows, despite avoiding raw data sharing, still transmit model updates that can leak sensitive information through sophisticated attacks.

Recent academic research has demonstrated that gradient inversion techniques can reconstruct training data

from shared model updates with alarming accuracy. Membership inference attacks can determine whether specific individuals participated in training. Model poisoning allows malicious actors to inject corrupted updates that degrade model performance or introduce backdoors. These vulnerabilities have prompted security researchers and enterprise architects to augment Federated Learning with additional privacy-preserving techniques that provide mathematical guarantees against data leakage and defend against adversarial attacks. Two approaches have emerged as essential complements to FL: Differential Privacy and Homomorphic Encryption, each addressing different aspects of the security challenge while enabling practical deployment of privacy-first AI systems.

Understanding Federated Learning Vulnerabilities

While Federated Learning eliminates the need to centralize raw data, the model updates transmitted between clients and servers contain more information than many practitioners realize. These updates—typically gradients or weight adjustments—encode patterns learned from local training data, and sophisticated attackers can exploit these patterns to extract sensitive information or compromise model integrity.

Gradient leakage represents one of the most serious threats to FL privacy. When a client trains a model on local data and computes gradients (the mathematical directions indicating how weights should change), those gradients inherently reflect properties of the training data. Researchers have demonstrated gradient inversion attacks that reconstruct original training samples with high fidelity, particularly in scenarios with small batch sizes or specific model architectures. In one notable study, attackers successfully recovered original images from image classification tasks and reconstructed text from language models by analyzing the gradients shared during federated training. This vulnerability is especially concerning in domains like healthcare, where a single reconstructed patient record could constitute a HIPAA violation.

Membership inference attacks operate at a different level, attempting to determine whether a specific individual's data was included in the training set. These attacks exploit the tendency of machine learning models to memorize aspects of their training data, particularly outliers or unusual examples. An attacker with query access to the trained model can probe its responses and apply statistical tests to infer membership with concerning accuracy. For enterprises, this creates liability risks—even if raw data isn't exposed, proving that a customer's information was used in training might violate privacy agreements or regulations. The risk intensifies in scenarios involving sensitive attributes like medical diagnoses, financial hardship, or legal status.

Model poisoning attacks target the integrity rather than the privacy of federated systems. Malicious participants can submit deliberately corrupted updates designed to degrade model performance, introduce biases, or create backdoors that trigger specific behaviors under certain conditions. Because the central server aggregates updates from multiple clients, it may struggle to distinguish legitimate updates from poisoned ones, especially when multiple colluding attackers coordinate their efforts. The distributed nature of FL actually amplifies this risk compared to centralized training, where data quality and provenance can be more easily controlled.

These vulnerabilities emerge from fundamental properties of how neural networks learn and share information:

- Gradients encode information about training data distribution and specific examples
- Model updates reveal patterns that correlate with sensitive attributes
- Aggregation servers lack visibility into local training processes
- Heterogeneous client participation creates opportunities for adversarial manipulation
- Communication overhead encourages sparse updates that leak more information per transmission

The discovery of these attack vectors has prompted a critical evolution in FL deployment strategies. Organizations implementing Federated Learning must now integrate additional privacy-preserving techniques that provide mathematical guarantees against information leakage and robust defenses against adversarial manipulation. This realization has elevated Differential Privacy and Homomorphic Encryption from optional enhancements to essential components of production-ready federated systems.

Differential Privacy as a Mathematical Privacy Guarantee

Differential Privacy provides a rigorous mathematical framework for quantifying and limiting privacy loss when sharing information derived from datasets. The core principle is elegant: a privacy-preserving mechanism should produce similar outputs whether or not any single individual's data is included in the input dataset. This property ensures that an attacker learning the output gains negligible additional information about any specific individual's participation or attributes.

The formal definition involves carefully calibrated noise injection. Before sharing model updates in a Federated Learning context, clients add random noise to their gradients according to specific mathematical distributions (typically Gaussian or Laplacian). The noise magnitude is governed by two parameters: epsilon (ε) and delta (δ). Epsilon represents the privacy budget—smaller values provide stronger privacy guarantees but introduce more noise that can degrade model utility. Delta represents the probability that the privacy guarantee fails, typically set to extremely small values. Together, these parameters create a tunable privacy-utility tradeoff that organizations can adjust based on their risk tolerance and performance requirements.

In practice, implementing Differential Privacy in Federated Learning requires careful consideration of where and how noise is applied. The most common approach, local differential privacy, involves clients adding noise to their model updates before transmission. This ensures that even if the central server or network communication is compromised, the transmitted updates contain no recoverable information about individual training examples. An alternative approach, central differential privacy, applies noise during server-side aggregation after collecting updates from multiple clients. This typically enables better model utility because noise can be calibrated against the aggregated update rather than individual contributions, but it requires trusting the server to handle raw updates securely before noise injection.

The introduction of noise creates inevitable accuracy degradation, but research has demonstrated that well-calibrated Differential Privacy can maintain acceptable model performance while providing strong privacy guarantees. The impact varies significantly based on several factors:

1. **Dataset size:** Larger datasets enable better privacy-utility tradeoffs because noise impact dilutes across more samples
2. **Model complexity:** Simpler models with fewer parameters typically handle noise injection better than extremely deep architectures
3. **Privacy budget allocation:** Sophisticated accounting methods track cumulative privacy loss across training iterations and allocate budgets strategically
4. **Adaptive noise scheduling:** Advanced implementations adjust noise levels throughout training, applying stronger privacy protections early when gradients are large and reducing noise as convergence approaches

Differential Privacy specifically addresses the membership inference and gradient leakage vulnerabilities that plague basic Federated Learning implementations. By adding calibrated noise to model updates, DP ensures that an attacker cannot determine whether specific training examples were present or reconstruct original data from observed gradients. The mathematical guarantees provide auditable evidence of privacy

protection, which becomes crucial for regulatory compliance and building trust with customers and partners.

Organizations deploying privacy-preserving AI can leverage libraries like TensorFlow Privacy, PyTorch Opacus, and IBM Diffprivlib that automate noise injection and privacy budget tracking. These tools integrate directly into existing ML training workflows, enabling data scientists to add differential privacy protections without becoming cryptography experts. Platforms like Shakudo provide pre-integrated environments where these privacy-preserving libraries work seamlessly with federated learning frameworks, allowing teams to deploy DP-enhanced FL systems in their own infrastructure within days rather than spending months on integration and security hardening. This acceleration is critical as regulatory deadlines approach and competitive pressures demand faster AI innovation cycles.

Homomorphic Encryption for Secure Computation

Homomorphic Encryption represents a cryptographic breakthrough that enables mathematical operations on encrypted data without ever decrypting it. This property seems almost magical—imagine performing calculations on locked safes without opening them, yet still obtaining the correct encrypted result that can only be unlocked by authorized parties. For Federated Learning, HE provides a powerful mechanism to protect model updates during transmission and aggregation, ensuring that even compromised servers or network eavesdroppers gain zero information about the underlying data.

The fundamental concept involves encrypting model updates at the client level before transmission to the central server. Each client trains their local model, computes gradients or weight updates, encrypts these updates using HE schemes, and transmits only the encrypted versions. The aggregation server performs mathematical operations—typically weighted averaging across client updates—directly on the encrypted data. The server never sees plaintext updates and cannot infer anything about individual client contributions. Only after aggregation completes does authorized decryption occur, revealing the combined global update that reflects learning from all participants while protecting individual privacy.

Several HE schemes exist with different properties and performance characteristics. Partially homomorphic encryption supports either addition or multiplication operations on encrypted data, which suffices for many Federated Learning aggregation algorithms like Federated Averaging that primarily require computing weighted sums. Fully homomorphic encryption supports arbitrary computations on encrypted data but introduces significant computational overhead that can make real-time training impractical. Most production FL deployments leverage partially homomorphic schemes or somewhat homomorphic encryption that supports a limited number of both addition and multiplication operations, striking a balance between security and performance.

The integration of HE into FL workflows addresses specific attack vectors that Differential Privacy alone cannot fully mitigate. While DP protects against statistical inference by adding noise, HE provides cryptographic guarantees that the aggregation server and any network observers learn absolutely nothing about individual updates. This defense is particularly valuable against insider threats—malicious administrators with server access cannot inspect or tamper with client updates when HE protects the aggregation process. HE also defends against model poisoning by enabling verification mechanisms where clients can detect if their encrypted contributions were properly included in aggregation without revealing the contributions themselves.

Implementing Homomorphic Encryption introduces important practical considerations:

- **Computational overhead:** HE operations are significantly more expensive than plaintext operations, potentially increasing training time by 10-100x depending on the scheme and implementation
- **Library maturity:** HE libraries like Microsoft SEAL, PALISADE, and HElib have improved significantly but still require expertise to deploy correctly
- **Key management:** Secure distribution and storage of encryption keys across distributed clients demands robust infrastructure
- **Bandwidth requirements:** Encrypted data is larger than plaintext, increasing communication costs in bandwidth-constrained environments

Despite these challenges, recent advances in GPU-accelerated HE operations and optimized secure aggregation protocols have made production deployment increasingly feasible. NVIDIA's cuHE library leverages GPU parallelism to accelerate homomorphic computations, dramatically reducing the performance penalty. Quantization and sparsification techniques reduce the size of model updates before encryption, mitigating bandwidth concerns. Hybrid approaches combine HE for secure aggregation with DP for additional statistical privacy, leveraging the complementary strengths of both techniques.

The value proposition becomes clear when considering regulated industries and cross-organizational collaboration scenarios. Financial institutions collaborating on fraud detection can use HE-protected FL to ensure that even the coordinating entity cannot see individual bank data. Healthcare networks can aggregate insights across hospitals with cryptographic assurance that patient data never becomes visible outside originating institutions. Multinational corporations can train global models while guaranteeing that regional data never crosses borders in unencrypted form, satisfying data sovereignty requirements.

Organizations seeking to implement HE-secured Federated Learning benefit from platforms that provide pre-integrated cryptographic libraries and optimized communication protocols. Shakudo's sovereign deployment model ensures that the entire HE infrastructure—including key management, secure aggregation servers, and encrypted storage—resides within customer-controlled environments across multiple regions. This architecture eliminates concerns about cloud provider access to encryption keys or aggregated models while providing the performance optimizations and library integrations necessary for practical deployment.

Building a Combined Defense Strategy

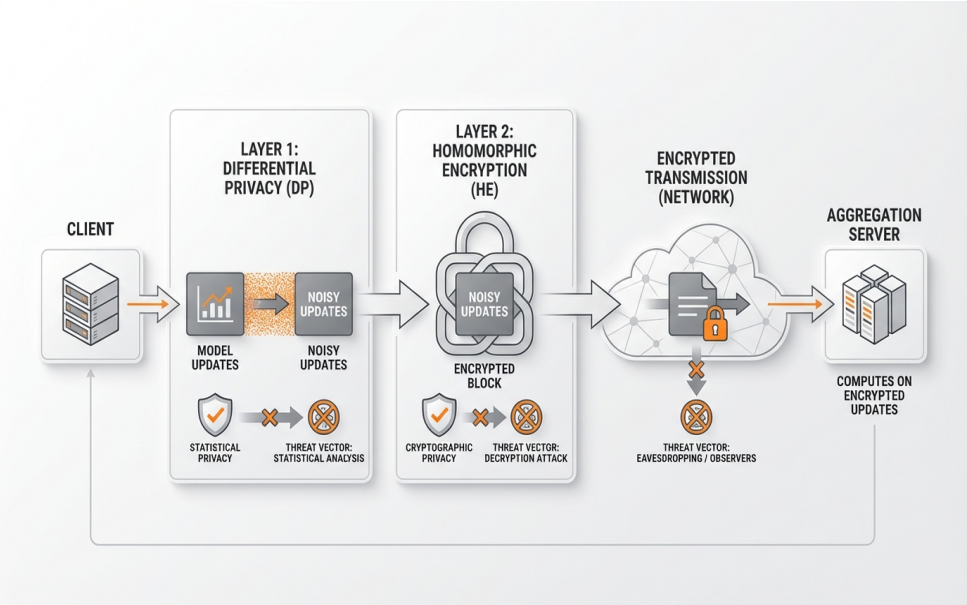
The most robust privacy-preserving Federated Learning implementations combine Differential Privacy and Homomorphic Encryption in complementary layers that address different aspects of the security challenge. This defense-in-depth approach acknowledges that no single technique provides perfect protection against all threat vectors, but carefully orchestrated combinations can achieve both strong privacy guarantees and practical performance.

Differential Privacy and Homomorphic Encryption protect against fundamentally different attack models. DP provides statistical privacy by ensuring that model updates, even if observed in plaintext, reveal negligible information about individual training examples. This protection operates at the information-theoretic level—an attacker with unlimited computational resources still cannot reliably infer membership or reconstruct data from DP-protected updates. However, DP alone does not prevent the aggregation server or network observers from seeing the noisy updates themselves, which might still reveal aggregate patterns or enable sophisticated correlation attacks across multiple training rounds.

Homomorphic Encryption operates at the cryptographic level, ensuring that model updates remain computationally infeasible to decrypt without proper keys. This prevents the aggregation server, network eavesdroppers, and even malicious administrators from observing update contents. However, HE alone does not provide guarantees about what information might be inferred if encryption is eventually broken or if authorized parties decrypt and analyze the aggregated results. An attacker who compromises encryption keys gains access to whatever information exists in the encrypted updates.

The synergy emerges when these techniques work together. Clients first apply Differential Privacy to add calibrated noise to their model updates, then encrypt the noisy updates using Homomorphic Encryption before transmission. This layered approach ensures that:

- Network observers see only encrypted data with no information leakage
- The aggregation server performs computations on encrypted updates without seeing plaintext
- Even if encryption is compromised, attackers obtain only DP-protected noisy updates
- Statistical and cryptographic privacy guarantees compound rather than conflict



Combining Differential Privacy and Homomorphic Encryption creates complementary layers of protection against different attack vectors.

Implementing this combined strategy requires careful system design across the entire FL pipeline. The initialization phase establishes encryption keys and privacy budgets, distributing public keys to clients while ensuring private keys remain secure. During local training, clients compute gradients from their private data, apply DP noise injection according to allocated privacy budgets, encrypt the noisy updates using HE public keys, and transmit encrypted updates to the aggregation server. The server performs secure aggregation directly on encrypted data, combining multiple client contributions without decryption, and only authorized entities (possibly using threshold cryptography requiring multiple key shares) decrypt the final aggregated result.

Several practical considerations shape deployment decisions:

1. **Privacy budget management:** Organizations must decide how to allocate epsilon across training iterations, balancing privacy protection with model convergence speed
2. **Encryption scheme selection:** Choosing between partially and fully homomorphic encryption based on required operations and acceptable performance overhead
3. **Communication optimization:** Implementing gradient compression, sparsification, and efficient encoding to minimize bandwidth costs amplified by encryption
4. **Fault tolerance:** Designing aggregation protocols that handle client dropouts and network failures gracefully without compromising security
5. **Compliance verification:** Establishing audit trails that demonstrate privacy protections to regulators without exposing sensitive information

The operational complexity of deploying these combined defenses has traditionally created barriers for organizations without deep cryptography expertise and extensive infrastructure resources. Building a production-ready FL system with DP and HE protection typically requires 6-12 months of development

effort, integrating multiple specialized libraries, implementing secure communication protocols, and hardening infrastructure against various attack vectors.

Modern AI deployment platforms have begun addressing this complexity by providing pre-integrated FL stacks with privacy-preserving techniques ready for deployment. Shakudo enables organizations to launch complete FL environments—including secure aggregation servers, DP-enhanced training frameworks, HE libraries, and monitoring infrastructure—within their own VPCs in days rather than months. The platform handles integration challenges between TensorFlow Privacy, PySyft, Microsoft SEAL, and other specialized tools while ensuring that all compute and storage remain within customer-controlled boundaries. This approach maintains the security guarantees of self-hosted deployment while dramatically accelerating time-to-value, allowing data science teams to focus on model development rather than cryptographic infrastructure.

Implementation Roadmap for Enterprises

Successfully deploying privacy-preserving Federated Learning requires a phased approach that balances technical implementation with organizational readiness, regulatory compliance, and business value demonstration. Organizations that rush into production deployments without proper foundation often encounter security gaps, performance issues, or misalignment between technical capabilities and business requirements.

The journey typically begins with use case identification and feasibility assessment. Not all machine learning problems benefit equally from Federated Learning—the approach makes most sense when data cannot or should not be centralized due to privacy regulations, competitive concerns, or sovereignty requirements. Ideal initial use cases involve multiple data silos within a single organization (regional offices, subsidiaries, or departments) rather than immediately attempting cross-organizational collaboration. This controlled environment allows teams to master the technology before introducing the complexity of multi-party coordination and trust negotiation.

Once a suitable use case is identified, the next phase involves architecture design and infrastructure preparation. Organizations must decide whether to deploy FL infrastructure in public cloud VPCs, private cloud environments, or hybrid configurations that span multiple regions or providers. The architecture should specify where aggregation servers will run, how clients will connect securely, where model artifacts will be stored, and how encryption keys will be managed. Network topology matters significantly—clients behind restrictive firewalls may struggle with persistent connections to aggregation servers, requiring asynchronous communication patterns or relay architectures.

Technology stack selection follows architecture design. The FL framework forms the foundation—options include NVIDIA FLARE for GPU-accelerated healthcare and research applications, PySyft for flexible experimental deployments, TensorFlow Federated for Google ecosystem integration, or Flower for framework-agnostic implementations. This base framework must integrate with privacy-preserving libraries including TensorFlow Privacy or Opacus for Differential Privacy, and SEAL, PALISADE, or TenSEAL for Homomorphic Encryption. Monitoring and observability tools track training progress, model convergence, privacy budget consumption, and system health across distributed clients.

The implementation sequence typically follows this progression:

1. **Establish baseline centralized model:** Train a traditional model on combined data (if possible) to establish performance benchmarks that FL must approach
2. **Deploy basic FL without privacy enhancements:** Verify that federated training produces comparable results to centralized training, validating architecture and communications
3. **Integrate Differential Privacy:** Add DP noise injection and optimize privacy-utility tradeoffs through experimentation with epsilon values and noise schedules
4. **Implement Homomorphic Encryption:** Layer HE-based secure aggregation over DP-protected updates, ensuring cryptographic protection during transmission
5. **Conduct security testing:** Perform red team exercises attempting membership inference, gradient leakage, and model poisoning attacks against the deployed system

6. **Scale to production:** Expand from pilot groups to full deployment with operational monitoring, incident response procedures, and compliance documentation

Each phase should include defined success criteria and rollback plans. Common pitfalls include underestimating communication overhead in bandwidth-constrained environments, overlooking client device heterogeneity that causes stragglers to delay training rounds, and failing to account for regulatory audit requirements in system logging design.

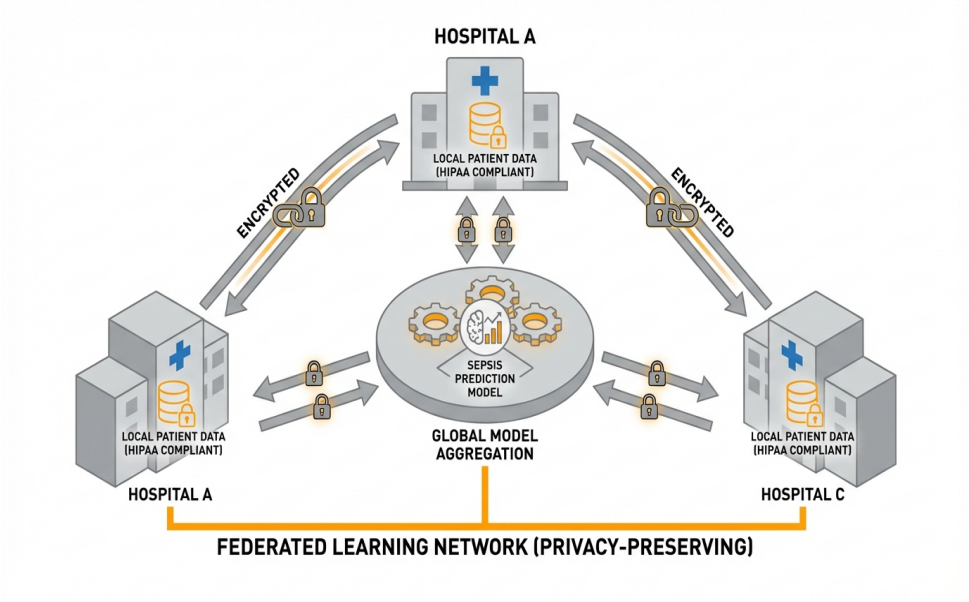
Organizational readiness deserves equal attention alongside technical implementation. Data science teams need training on FL-specific considerations like non-IID data distributions, client selection strategies, and privacy-utility tradeoffs. Legal and compliance teams must understand how DP and HE provide privacy guarantees and how to document these protections for regulatory audits. Executive sponsors need visibility into progress and business value realization through metrics like time-to-model-deployment, number of participants enabled, and regulatory risk reduction.

The traditional barrier to executing this roadmap has been the 6-12 month timeline required to integrate diverse components, secure infrastructure, and achieve regulatory compliance certification. Organizations implementing FL on platforms like Shakudo compress this timeline dramatically by deploying pre-integrated stacks that include all necessary frameworks, libraries, and security configurations within sovereign infrastructure. Teams can progress from architecture design to pilot deployment in weeks rather than quarters, with built-in compliance controls and audit capabilities that satisfy regulatory requirements. This acceleration enables organizations to realize business value from privacy-preserving AI while competitors are still negotiating vendor contracts or assembling internal expertise.

Real-World Applications Across Industries

Privacy-preserving Federated Learning has moved beyond academic research into production deployments across industries where data sensitivity and regulatory constraints traditionally limited AI adoption. These implementations demonstrate both the business value and practical challenges of deploying FL with differential privacy and homomorphic encryption at scale.

Healthcare organizations represent perhaps the most compelling application domain for privacy-preserving FL. Hospital networks need to build diagnostic models that learn from diverse patient populations across multiple facilities, but HIPAA regulations severely restrict data sharing even within the same health system. FL enables collaborative model training where each hospital trains on local patient records, and only encrypted, DP-protected updates are shared to build consensus models. One notable implementation involved a multi-hospital network developing sepsis prediction models that achieved performance comparable to centralized training while ensuring no patient data crossed institutional boundaries. The system used TensorFlow Federated with differential privacy for statistical protection and secure aggregation protocols for cryptographic guarantees. The resulting model improved early sepsis detection across all participating hospitals while maintaining HIPAA compliance and enabling smaller facilities to benefit from the broader dataset diversity.



Multi-hospital federated learning enables collaborative sepsis prediction models while keeping patient data within institutional boundaries.

Financial services institutions face similar constraints around proprietary transaction data and regulatory requirements. Fraud detection models improve dramatically when trained on data from multiple banks, capturing broader patterns of fraudulent behavior that individual institutions miss. However, competitive concerns and regulations prevent banks from pooling transaction data centrally. FL implementations in this sector typically involve consortiums of banks collaborating through trusted third parties who operate aggregation servers. Each bank trains locally on their transaction data with differential privacy protections, encrypts updates using homomorphic encryption, and submits to the aggregation coordinator who

combines updates without seeing individual bank contributions. The resulting fraud detection models identify cross-bank fraud patterns while protecting competitive information and customer privacy. One European banking consortium reported 23% improvement in fraud detection rates while reducing false positives by 15% through FL collaboration that would have been impossible with traditional data sharing approaches.

Telecommunications companies leverage FL for network optimization and predictive maintenance across geographically distributed infrastructure. A major telecom deployed FL to predict network failures by training models across thousands of cell towers, each generating operational data that remains locally stored due to bandwidth constraints and security policies. The FL system aggregates insights about failure patterns without centralizing petabytes of sensor data, enabling proactive maintenance that reduced outages by 31%. Differential privacy protections prevent competitors who might compromise the aggregation server from inferring specific network topology or equipment configurations.

Manufacturing and industrial IoT scenarios involve training predictive maintenance models across factory equipment without exposing proprietary production processes. A automotive manufacturer implemented FL across assembly plants in different countries, building models that predict equipment failures while ensuring production data never crosses borders in violation of data sovereignty requirements. The deployment used hybrid cloud architecture with regional aggregation servers in each jurisdiction, then hierarchical federation to combine regional models into a global model. This approach satisfied data residency requirements while enabling learning from the full equipment population.

Retail and consumer applications include keyboard prediction models trained across smartphones (pioneered by Google's GBoard), recommendation systems that learn from user behavior without collecting browsing history, and personalized healthcare applications that improve from population data while keeping individual health metrics on personal devices. These consumer-facing implementations typically emphasize local differential privacy where noise injection happens on-device before any data leaves user control.

Cross-cutting lessons from these deployments include:

- Start with high-value use cases where data cannot be centralized due to hard constraints, not optional preferences
- Invest heavily in monitoring and observability—distributed training is harder to debug than centralized approaches
- Plan for heterogeneous client capabilities with graceful degradation when some participants have limited compute or bandwidth
- Build trust gradually, often starting with intra-organizational FL before expanding to multi-party scenarios
- Document privacy protections thoroughly for regulatory compliance and external audit

The technical success of these implementations required significant infrastructure investment and specialized expertise. Organizations deploying similar systems benefit from platforms that provide production-ready FL environments with privacy enhancements pre-integrated. Shakudo enables teams to replicate these architectures within their own infrastructure—whether single-cloud, multi-cloud, or hybrid environments—with the security guarantees of sovereign deployment and the velocity of managed services.

This combination allows organizations to move from pilot to production in months rather than years while maintaining complete control over where data processing occurs and where model artifacts reside.

Ready to Get Started?

Shakudo enables enterprise teams to deploy AI infrastructure with complete data sovereignty and privacy.

shakudo.io

info@shakudo.io

Book a demo: shakudo.io/sign-up