Confidential Computing for AI in Regulated Industries: Using Enclaves to Protect Data in Use

Compliance Guide for AI/Machine Learning Teams

Prepared by Jacobian Engineering | 2026-02-09

This guide is for informational purposes only and does not constitute legal advice.

Executive Summary

Regulated industries often want the benefits of AI, but they cannot accept casual data handling. Healthcare data, financial records, customer identifiers, and proprietary research require strong controls. Traditional cloud security protects data at rest and in transit. It does not fully address a harder problem. How do you protect data while it is being processed.

Confidential computing is a set of technologies that use hardware based isolation to protect data in use. One common pattern is the use of enclaves, also called trusted execution environments. Enclaves can limit what cloud administrators and the host operating system can see, even when workloads run on shared infrastructure. This guide explains how enclaves work, when they help, where they fall short, and how AI teams can use them to support regulated workloads.

What confidential computing is

Confidential computing aims to protect data while it is actively processed in memory. It does this by running sensitive code inside a hardware protected region, isolated from the rest of the system. The details vary by platform, but the basic idea is consistent. Even if someone has elevated access to the host, the enclave contents remain protected.

For AI and machine learning, confidential computing is often used to protect sensitive inputs, model parameters, and cryptographic keys during inference. It can also protect certain parts of training workflows, although training inside enclaves can be more complex due to performance and hardware constraints.

Why AI teams care

- **Regulated data:** Customers may not allow sensitive data to be processed unless additional isolation controls exist.
- **Multi tenant risk:** Shared infrastructure creates concern about cross tenant exposure.
- **Insider risk:** Customers worry about privileged administrators, even in cloud environments.
- **Model protection:** Some companies want to protect model weights and proprietary logic.

How enclaves work at a high level

An enclave is an isolated execution environment. Code inside the enclave can access data passed into it, but other processes cannot read enclave memory directly. Enclave systems usually include an attestation mechanism. Attestation allows a client to verify that the enclave is running trusted code before sending sensitive data or keys.

Ask a practical question. Would you be comfortable sending a customer’s sensitive data to a model if you could not verify where the code runs. Attestation is the mechanism that makes that verification possible.

Key concepts

- **Trusted execution environment:** Hardware backed isolation for code and data.
- **Attestation:** Proof that the enclave is running a specific trusted configuration.
- **Sealing:** Encrypting and binding data so it can only be unsealed in the same enclave context.
- **Key management:** Protecting cryptographic keys used to decrypt inputs and encrypt outputs.

Where enclaves fit in AI architectures

Secure inference for sensitive inputs

A common pattern is to run the inference service inside an enclave. The client encrypts input data. The enclave attests and receives decryption keys. The model runs inference, then returns an encrypted result. In this pattern, sensitive inputs and intermediate computations remain protected while in use.

Protected retrieval and feature processing

Another pattern is to place sensitive retrieval and feature processing inside an enclave. A RAG system might retrieve sensitive documents and pass them to a model. By doing retrieval and redaction inside an enclave, you can reduce exposure to the host environment.

Key isolation for AI platforms

Even if you do not run full inference in an enclave, you can use enclaves to isolate encryption keys, signing keys, and authentication logic. This reduces the risk that a privileged process can extract keys and decrypt stored data.

Threat model and realistic expectations

Enclaves are not magic. They reduce certain risks, especially risks involving privileged access to the host. They do not replace basic security controls. If your application has a vulnerability, an attacker can still abuse it. If you feed sensitive data to an external model provider, an enclave in your environment does not control what happens next.

Risks enclaves can reduce

- **Host administrator access:** Reduced ability for privileged host users to inspect in memory data.
- **Memory scraping attacks:** Reduced exposure of sensitive data in process memory.
- **Key theft from host:** Reduced risk of extracting keys from standard application memory.

Risks enclaves do not solve

- **Application logic flaws:** Prompt injection and authorization bugs still matter.
- **Data quality and bias:** Enclaves do not improve model fairness or correctness.
- **Misconfiguration:** Poor IAM or weak network controls can still expose systems.

- **Side channel risks:** Some enclave technologies have had side channel vulnerabilities over time. You still need patching and monitoring.

Reference architecture example for confidential inference

A simple reference architecture can make the idea concrete. Imagine a customer sends sensitive input to a prediction API. The goal is to prevent the cloud host from seeing the plaintext input while still allowing inference.

1. The client requests an attestation document from the enclave based inference service.
2. The client or a trusted key service verifies the attestation and confirms the enclave measurement matches an approved build.
3. After verification, a short lived decryption key is released to the enclave, not to the host environment.
4. The client encrypts the input and sends it to the service. The enclave decrypts inside the protected boundary and runs inference.
5. The enclave returns an encrypted output, or returns a plaintext output only if the output is not sensitive.

This pattern can be combined with tokenization and field level encryption. It can also be combined with retrieval systems where sensitive context is decrypted only inside the enclave.

Operational considerations and tradeoffs

Enclaves introduce operational constraints that engineering teams need to plan for. These constraints are not a reason to avoid the technology. They are a reason to design carefully.

Performance and scaling

Inference inside enclaves can have performance overhead. Some workloads may not fit due to memory limits or acceleration constraints. Test with representative models early. If performance is unacceptable, consider using enclaves for key isolation and pre processing, while keeping heavy compute outside the enclave.

Debugging and observability

Enclaves reduce visibility by design. That can make debugging harder. Plan how you will capture diagnostics without leaking sensitive data. Use structured event logs and correlation IDs. Avoid logging raw prompts, documents, or payloads in plaintext.

Patching and measurement changes

Software updates can change enclave measurements, which can break attestation workflows. Build a release process that coordinates code changes, attestation allow lists, and key release policies. If you ship frequently, this needs automation. If you ship rarely, it needs careful change control.

Shared responsibility and customer expectations

Customers may interpret the word confidential as a promise. Be precise about what you protect and what you do not. Enclaves can protect in use data on your side. They do not change how customers handle data on their side. Clear documentation prevents misunderstandings.

Common pitfalls

- **Too much code inside the enclave:** A large enclave increases complexity and creates more surface for mistakes.
- **Weak key release policies:** If keys are released without strict attestation checks, the main benefit is lost.
- **Logging sensitive content:** Teams protect memory but leak secrets through logs and debug traces.
- **Ignoring vendor and dependency risk:** Enclave security still depends on underlying hardware and platform updates.

Decision checklist

Before adopting enclaves, ask a few concrete questions.

- **What specific data exposure risk are we trying to reduce:** Host admins, multi tenant concerns, key theft, or something else.
- **What part of the workflow must be protected:** Inference inputs, retrieval context, model weights, or keys.
- **Can we operate it reliably:** Monitoring, patching, incident response, and capacity planning.
- **Do we have a plan for attestation:** Who verifies it and who controls the allow list.

Implementation steps for adopting enclaves

Step 1: Define the compliance and data requirements

Start with data classification and a clear statement of what needs protection. Is it personal data. Payment data. Clinical notes. Proprietary model weights. Identify where that data exists in the workflow. If you cannot map the data flow, you cannot decide where an enclave helps.

Step 2: Choose an enclave platform that fits your cloud and workload

Cloud providers and hardware vendors offer different trusted execution technologies. The right choice depends on your runtime, performance needs, and integration requirements. Consider how attestation works, how keys are managed, and how the enclave integrates with your deployment pipeline.

Step 3: Design an enclave boundary and integration points

Keep the enclave scope small. Only run the sensitive parts inside it. Minimizing the code inside the enclave reduces complexity and reduces the attack surface. Define what data enters the enclave, what leaves, and how it is validated. Decide how you will log events without leaking sensitive content.

Step 4: Implement attestation and key delivery

Attestation is only valuable if clients or key services verify it. Build a process where keys are released only when a trusted enclave measurement is presented. This is often where projects stall because it requires careful coordination between platform engineering and security teams.

Step 5: Operationalize monitoring and incident response

Enclave systems still need monitoring. You need to know when attestation fails, when unusual usage occurs, and when software updates change enclave measurements. Define runbooks so operators know what to do when an enclave fails to start or when a dependency breaks attestation.

How enclaves support compliance conversations

Confidential computing can support compliance objectives because it reduces certain exposure risks. It can strengthen your story for confidentiality and access control. It can also help with customer contractual requirements for sensitive data processing. The key is to be precise. Do not claim that enclaves make you compliant. They are one control in a broader program that includes policies, access management, logging, and security testing.

Business benefits

For regulated AI use cases, enclaves can enable deals that would otherwise be blocked. They can reduce customer concerns about multi tenant environments and privileged access. They can also reduce the scope of what must be trusted, which can simplify risk assessments. The tradeoff is operational complexity. If you cannot operate the system reliably, the control will not help you.

How Jacobian Engineering supports confidential AI deployments

Jacobian Engineering helps teams design cloud architectures that balance compliance requirements with delivery speed. For confidential computing projects, that can include threat modeling, enclave design reviews, key management integration, logging and monitoring design, and penetration testing. The team also supports compliance evidence design so enclave controls can be explained clearly during customer reviews.

Conclusion

Confidential computing enclaves can protect sensitive data and keys while AI workloads run in the cloud. They are most valuable when you have a clear threat model and a specific data exposure problem to solve. If you keep the enclave boundary small, implement attestation correctly, and operationalize monitoring, enclaves can become a practical part of a regulated AI architecture.

If you are evaluating enclaves for an AI product or platform, Jacobian Engineering can help you model the risks, design the architecture, and implement the supporting controls needed to run the system safely.