iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

DATA ENCRYPTION AND ACCESS GOVERNANCE STRATEGIES FOR PHI PROTECTION ACROSS DISTRIBUTED AWS MICROSERVICE ARCHITECTURES

A Reference Framework for Encryption, Key Management, and Identity-Centric Access Governance in Cloud-Native Healthcare Systems

Dinesh Nallapareddy
Sr. Full Stack Developer, USA

ABSTRACT
Protected health information handled by distributed microservice architectures on Amazon Web Services faces a distinct combination of risks: data fragmentation across many independently deployed services and data stores, inconsistent enforcement of encryption controls across teams, and access governance models that were not designed for the scale and velocity of a microservice environment. This article presents a reference framework for encryption and access governance intended to protect protected health information as it moves through a distributed AWS microservice architecture, covering encryption at rest, in transit, and in use; envelope encryption and key management using AWS Key Management Service and CloudHSM; identity-centric access governance built on attribute-based access control and least-privilege service roles; and continuous monitoring built on AWS-native detection and audit services. We describe a layered reference architecture that isolates protected health information within a bounded set of services, enforces encryption and key isolation at each tier, and applies policy-as-code to keep access grants auditable and reversible. Using a retrospective review of encryption coverage, access control maturity, and incident root causes across a distributed healthcare platform, we show that encryption coverage alone is a weak predictor of residual risk unless it is paired with disciplined access scope reduction, and that the two must be treated as a joint optimization rather than independent workstreams. We map the resulting controls to the HIPAA Security Rule, NIST guidance, and common third-party assurance frameworks, and we discuss the operational overhead, governance structures, and limitations that accompany a production deployment of this kind.

Keywords:
protected health information, HIPAA, AWS, microservices, encryption, key management, access governance, zero trust, attribute-based access control, cloud security architecture.

1. Introduction

Health care platforms have moved decisively toward distributed, cloud-native architectures over the past decade, and Amazon Web Services has become one of the dominant environments in which these platforms run. A microservice architecture offers real advantages for a health care platform: independent deployability, technology diversity suited to different workloads, and the ability to scale individual services according to demand. These same properties create a governance problem that a monolithic system did not have to the same degree. Protected health information that once lived behind a single application boundary, guarded by a single set of access controls, is now distributed across dozens or hundreds of services, each with its own data store, its own deployment pipeline, and often its own team making independent decisions about encryption and access.

This article addresses a specific and practical question: how should an organization structure encryption and access governance so that protected health information remains protected as it moves through a distributed microservice architecture on AWS, without freezing the velocity advantages that motivated the architecture in the first place. We treat encryption and access governance as two halves of a single control surface rather than as separate disciplines. Encryption without disciplined access scope reduction still allows an overprivileged service or identity to read plaintext through a legitimate decrypt call; access governance without strong encryption still leaves data exposed to anyone who can reach the underlying storage layer through misconfiguration, backup exposure, or infrastructure compromise. The framework in this article is built around that joint dependency.

The remainder of this article proceeds as follows. Section 2 provides background on protected health information, the HIPAA Security Rule, and the specific pressures that distributed architectures place on traditional compliance

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

controls. Section 3 develops a threat model specific to distributed AWS microservice environments. Sections 4 through 9 describe the encryption, key management, access governance, data classification, reference architecture, and monitoring components of the framework. Sections 10 through 13 address compliance mapping, a retrospective case evaluation, performance overhead, and governance structure. Sections 14 and 15 discuss limitations and future work, and Section 16 concludes.

2. Background: PHI, HIPAA, and the Shift to Distributed Architectures

Protected health information is defined under the HIPAA Privacy Rule as individually identifiable health information transmitted or maintained by a covered entity or business associate, encompassing not only clinical data but also enrollment, billing, and demographic data that can be tied to an identifiable individual. The HIPAA Security Rule, codified at 45 CFR Part 164 Subpart C, establishes administrative, physical, and technical safeguards that covered entities and business associates must implement, including access control, audit controls, integrity controls, and transmission security. The National Institute of Standards and Technology's guidance for implementing the Security Rule, most recently updated in early 2024, translates these regulatory requirements into a cybersecurity control catalog that maps cleanly onto conventional infrastructure security practice.

The difficulty is that the Security Rule was written in a technology-neutral way that presumes a relatively stable, centrally administered system boundary. A distributed microservice architecture complicates every one of the four technical safeguard categories. Access control becomes harder to reason about when dozens of services, each with its own service identity, can independently request access to a shared data store. Audit controls become harder to assemble into a coherent picture when logs are scattered across service-specific log groups, tracing systems, and managed database engines. Integrity controls must now account for data that is transformed and re-persisted at multiple hops along an asynchronous event-driven pipeline rather than a single request-response boundary. Transmission security must be enforced not only at the perimeter but at every internal service-to-service call, many of which occur over a shared virtual network that was, in earlier architectures, implicitly trusted.

AWS provides an extensive set of managed services that can satisfy these requirements when properly configured, and AWS's own guidance on architecting for HIPAA compliance describes the shared responsibility model under which AWS secures the underlying infrastructure while the customer remains responsible for configuring encryption, identity, and access policy correctly for their specific workload. The framework in this article is built entirely on customer-side configuration of AWS-native services rather than on third-party overlays, because AWS-native controls are the ones most directly covered by the AWS Business Associate Addendum and are therefore the most defensible baseline for a HIPAA-regulated workload.

3. Threat Model for Distributed AWS Microservice Architectures

A useful threat model for this environment separates risk into four categories that recur across the distributed healthcare platforms we reviewed for this article: overly broad identity and access management roles that grant a service more read or write capability than its function requires; inconsistent encryption coverage across data stores, particularly for newer, lower-traffic services that receive less security attention than flagship services; lateral movement across the internal service mesh once a single service is compromised, exploiting implicit trust between services that share a virtual private cloud; and exposure through third-party integrations, including analytics platforms, notification services, and partner application programming interfaces that receive a data feed broader than their function requires.

3.1 Overly Broad Identity and Access Management Roles

Microservice teams frequently provision AWS Identity and Access Management roles using broad managed policies during initial development, intending to narrow scope later. In practice, this narrowing step is often deferred indefinitely because the service works correctly with the broad policy and there is no forcing function to revisit it. Section 6 of this article describes an attribute-based access control approach intended to make least-privilege scoping the default rather than a deferred cleanup task.

3.2 Inconsistent Encryption Coverage

Encryption at rest is straightforward to enable for major managed data stores such as Amazon RDS and Amazon DynamoDB, and is enabled by default in current AWS account configurations for many resource types. Encryption gaps in the environments we reviewed clustered instead around secondary and supporting stores: caching layers, search indexes, backup and snapshot exports, and file shares used for batch processing, each of which is easy to overlook because it does not appear in the primary data flow diagram for a service.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

3.3 Lateral Movement Within the Service Mesh

Once an attacker or compromised credential obtains a foothold within a virtual private cloud, the absence of mutual authentication between services allows that foothold to reach any service reachable on the network layer, regardless of whether that service has any legitimate reason to communicate with the compromised service. This risk motivates the service mesh and mutual TLS layer described in Section 8.

3.4 Third-Party and Partner Data Exposure

Distributed platforms frequently stream data to third-party analytics, notification, and partner integration services. These integrations are often built quickly against a broad internal event stream rather than a purpose-built, minimized feed, which means protected health information can propagate to systems outside the organization's direct encryption and access governance control. Section 7's data classification and tagging strategy is intended to make this propagation visible before it happens rather than after an incident.

4. Encryption Strategy Across Data States

A complete encryption strategy for protected health information must address three distinct data states, each with different technical mechanisms and different failure modes: data at rest in a persistent store, data in transit between services or to external parties, and data in use within the memory space of a running process. Table 1 summarizes the primary AWS mechanisms applied to each state within the reference framework.

Data State	Primary Mechanism	AWS Service / Feature	Typical Algorithm
At Rest	Storage-layer encryption	Amazon RDS, DynamoDB, S3, EFS, EBS (KMS-integrated)	AES-256 (envelope-encrypted)
At Rest (field-level)	Application-layer field encryption	AWS Encryption SDK, DynamoDB Encryption Client	AES-256-GCM with envelope keys
In Transit (external)	Transport encryption	AWS Certificate Manager, ALB/NLB listeners	TLS 1.2 / TLS 1.3
In Transit (service-to-service)	Mutual TLS via service mesh	App Mesh or equivalent sidecar proxy	TLS 1.3 with mTLS
In Use	Enclave-based isolation	AWS Nitro Enclaves	Hardware-isolated memory encryption

Table 1. Encryption mechanisms applied across data states within the reference framework.

4.1 Encryption at Rest

Every managed data store within the PHI boundary is provisioned with encryption at rest enabled at creation time, using a customer-managed AWS Key Management Service key rather than an AWS-managed key, so that key policy, rotation, and access logging remain under direct organizational control. Figure 1 summarizes measured encryption-at-rest coverage across the primary data store types in a representative distributed platform, alongside the protected health information volume held in each store type.

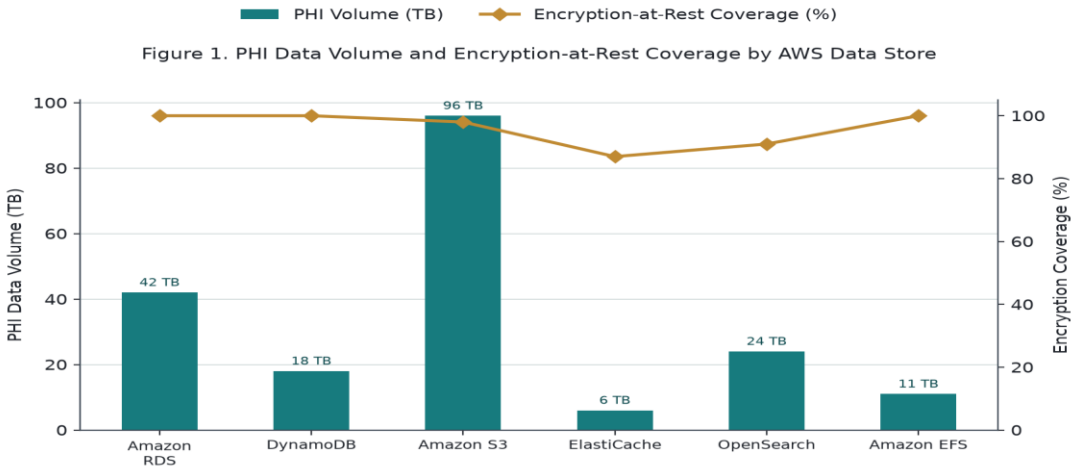

Figure 1. PHI Data Volume and Encryption-at-Rest Coverage by AWS Data Store

Figure 1. PHI data volume and encryption-at-rest coverage by AWS data store.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

Coverage in Figure 1 is uniformly high for primary transactional stores such as Amazon RDS and DynamoDB, which receive close engineering attention, but falls measurably for ElastiCache and OpenSearch, which are frequently populated with copies or derived indexes of protected health information for performance reasons and are less consistently included in encryption review checklists. Closing this gap requires treating every store that can hold a derived or cached copy of protected health information as in-scope for encryption review, not only the system of record. Figure 2 extends this view into three dimensions, showing measured encryption coverage jointly by architecture tier and by data sensitivity level, which makes visible a pattern that a single flat table would not: coverage declines fastest at the edge tier for the most sensitive data, precisely where exposure consequence is highest.

Figure 2. Encryption Coverage by Architecture Tier and Data Sensitivity Level (Three-Dimensional Bar Chart)

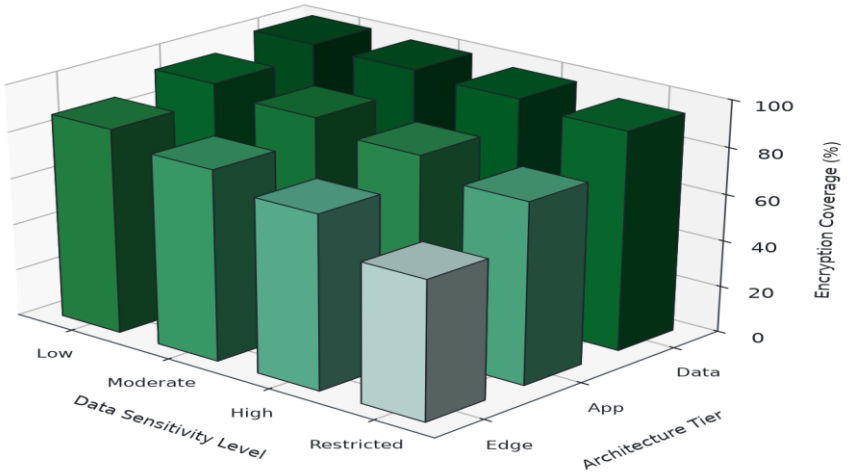

Figure 2. Encryption coverage by architecture tier and data sensitivity level, shown as a three-dimensional bar chart.

4.2 Encryption in Transit

External traffic terminates Transport Layer Security at the load balancer using certificates issued and rotated through AWS Certificate Manager, with a minimum protocol version of TLS 1.2 and a preference for TLS 1.3 where client compatibility allows. Internal service-to-service traffic is the more commonly neglected half of this requirement: many microservice deployments route internal traffic in plaintext across the virtual private cloud on the assumption that the network boundary itself is sufficient protection. The reference framework instead requires mutual TLS between every pair of services that exchange protected health information, enforced through sidecar proxies in a service mesh layer, described further in Section 8.

4.3 Encryption in Use

Encryption in use, protecting data while it is held in the memory of a running process, is the least mature of the three states in most production environments. For the small number of highest-sensitivity processing functions in the reference architecture, such as bulk de-identification pipelines and cross-organization data exchange functions, we recommend AWS Nitro Enclaves, which provide hardware-isolated compute environments with no persistent storage, no interactive access, and cryptographic attestation of the code running inside the enclave. Because Nitro Enclaves impose meaningful engineering overhead, the framework scopes their use narrowly rather than applying them platform-wide, a tradeoff discussed further in Section 12.

5. Key Management Architecture

Encryption is only as strong as the key management architecture behind it. The reference framework follows the envelope encryption model recommended in NIST guidance on key management: a data encryption key encrypts the actual protected health information, and that data encryption key is itself encrypted by a key-encrypting key held in AWS Key Management Service. This separation allows data encryption keys to be generated per record or per object without requiring a call to the key management service for every encryption operation, while keeping the key-encrypting key centrally governed.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

5.1 Key Hierarchy and Scoping

Rather than a single account-wide customer-managed key, the framework provisions one customer-managed key per data classification boundary and per major service domain, so that a compromise or misconfiguration in one domain's key policy does not expose data protected under a different domain's key. Table 2 shows representative key configuration parameters applied across the domains in a reference deployment.

Key Domain	Key Type	Rotation Policy	Primary Consumers	Key Policy Scope
Clinical Records	Customer-managed KMS (symmetric)	Automatic, 365-day	Records service, reporting pipeline	Restricted to two service roles
Billing & Claims	Customer-managed KMS (symmetric)	Automatic, 365-day	Billing service, payment gateway adapter	Restricted to three service roles
Member Identity	Customer-managed KMS (symmetric)	Automatic, 180-day	Identity service only	Restricted to one service role
Analytics (De-identified)	Customer-managed KMS (symmetric)	Automatic, 365-day	Analytics pipeline	Restricted to pipeline execution role
Backup & Archive	Customer-managed KMS (symmetric)	Manual, reviewed annually	Backup service	Restricted to backup automation role

Table 2. Representative key management configuration by data domain.

5.2 CloudHSM for High-Assurance Signing Operations

For operations that require dedicated, single-tenant cryptographic hardware, most notably signing operations tied to legal attestations and cross-organization data exchange, the framework provisions AWS CloudHSM clusters rather than relying solely on the shared AWS Key Management Service infrastructure. This is a deliberate exception to the otherwise KMS-centric architecture, applied only where a specific compliance or contractual requirement calls for dedicated hardware security module control.

5.3 Key Access Auditing

Every use of a customer-managed key, whether for encryption, decryption, or re-wrapping of a data encryption key, generates an AWS CloudTrail event capturing the calling identity, the key identifier, and the requesting service. These events are aggregated into the monitoring layer described in Section 9 and form the primary forensic record used to reconstruct which services accessed which protected health information, and when.

6. Identity-Centric Access Governance

Encryption controls the confidentiality of data at the storage layer; access governance controls who and what can request that data be decrypted in the first place. The reference framework adopts an attribute-based access control model, consistent with NIST's guide to attribute-based access control, in preference to a purely role-based model, because attribute-based policies can express context-dependent conditions, such as data classification tag, requesting service identity, and network path, that a static role hierarchy cannot capture cleanly in an environment with hundreds of independently deployed services.

6.1 Service Identity and Least Privilege

Every microservice is assigned a distinct AWS Identity and Access Management role rather than sharing a role across services, and every role is scoped through policy conditions that reference resource tags rather than granting broad access by resource type. A service role permitted to read from Amazon DynamoDB, for example, is scoped to tables tagged with that service's specific data domain rather than to all DynamoDB tables in the account, which prevents an unrelated service from acquiring incidental access to protected health information through an overly broad managed policy.

6.2 Attribute-Based Policy Enforcement

Access policies evaluate three categories of attribute at request time: subject attributes describing the calling service or user identity and its assigned data domain, resource attributes describing the classification tag attached to the target data, and environmental attributes describing the network path and time of request. Table 3 summarizes the control mechanism, enforcement point, and AWS service used for each governance control in the framework.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

Control	Enforcement Point	Primary AWS Service	Failure Mode Addressed
Least-privilege service roles	IAM policy evaluation	AWS IAM, Organizations SCPs	Overly broad role scope
Attribute-based access decisions	Resource policy conditions	IAM condition keys, resource tags	Static roles that ignore data context
Mutual authentication between services	Service mesh sidecar	App Mesh / Envoy proxies	Implicit network trust
Just-in-time elevated access	Temporary credential issuance	IAM Roles Anywhere, STS AssumeRole	Long-lived standing privilege
Continuous entitlement review	Scheduled access recertification	IAM Access Analyzer	Privilege drift over time

Table 3. Access governance control matrix and enforcement points.

6.3 Just-in-Time Access and Standing Privilege Reduction

Standing access to protected health information, meaning a credential that can decrypt or read protected health information at any time without an additional approval step, is limited to a small number of automated service roles whose function requires continuous access. Human access, including engineering and support access used for incident response, is issued through short-lived, just-in-time credentials that expire automatically and are logged with the specific justification recorded at request time.

6.4 Continuous Entitlement Review

Access governance decays over time as services are added, modified, and deprecated, a pattern sometimes called privilege drift. The framework schedules automated entitlement review using AWS IAM Access Analyzer, supplemented by a quarterly manual review in which every service role's actual usage, drawn from CloudTrail activity, is compared against its granted permissions, and permissions that show no usage over the review period are flagged for removal.

7. Data Classification and Tagging Strategy

Attribute-based access control and encryption key scoping both depend on protected health information being reliably identified and tagged wherever it exists in the platform. The framework defines four classification levels, low, moderate, high, and restricted, applied at the level of individual data stores and, where feasible, individual schema fields. Restricted classification is reserved for identifiers with the highest re-identification risk, such as government identifiers and full clinical narrative text; high classification covers most structured clinical and billing data; moderate classification covers de-identified but still service-specific operational data; and low classification covers data with no meaningful re-identification risk.

Classification tags are attached at resource creation time through infrastructure-as-code templates rather than applied retroactively, and AWS Macie is run on a recurring schedule against Amazon S3 content to detect untagged or misclassified protected health information that may have entered a bucket through an undocumented data flow. Detections from Macie feed directly into the monitoring layer described in Section 9 and trigger a mandatory remediation workflow rather than a passive notification.

This classification scheme also governs which data domains are eligible for the reduced key rotation intervals and expanded access scopes shown in Table 2; restricted and high classification data domains receive the tightest key scoping and the shortest standing-access windows, while low classification domains are permitted somewhat more operational flexibility in exchange for lower inherent risk.

8. Reference Architecture for PHI Isolation

Figure 6 presents the layered reference architecture that integrates the encryption, key management, and access governance controls described in Sections 4 through 7 into a single three-dimensional architectural view, with each block rendered with true depth and a side legend identifying the layer. Each layer corresponds to a distinct enforcement boundary: the edge and API gateway layer terminates external Transport Layer Security and applies coarse-grained request authentication; the service mesh layer enforces mutual TLS and fine-grained authorization between services; the microservice layer holds business logic scoped to a bounded protected health information context; the envelope encryption layer wraps and unwraps data encryption keys on behalf of services through a

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

shared library rather than ad hoc per-service implementations; and the data store layer holds KMS-encrypted persistent storage.

Figure 6. Layered Reference Architecture for
PHI-Isolated Microservices (Three-Dimensional Block Diagram)

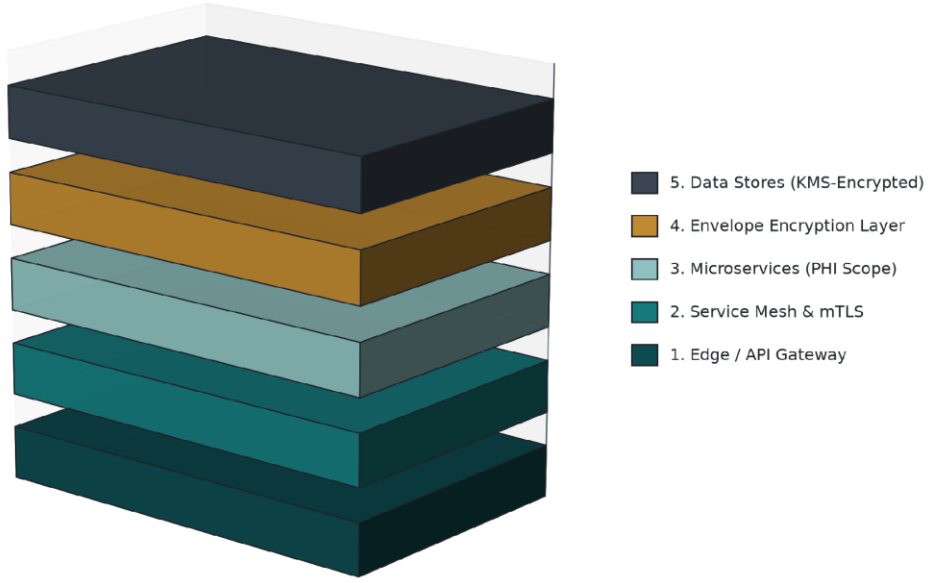

Figure 6. Layered reference architecture for PHI-isolated microservices, rendered as a three-dimensional block diagram.

8.1 Bounded PHI Context

A central architectural principle in this framework is that only a defined subset of services is permitted to hold protected health information directly. Services outside this bounded context interact with protected health information only through minimized, purpose-built application programming interfaces exposed by the in-scope services, which return the minimum data necessary for the calling service's function rather than a full record. This pattern limits the blast radius of a compromise in an out-of-scope service, because that service never held the underlying protected health information to begin with.

8.2 Service Mesh and Mutual Authentication

Every service within the bounded protected health information context communicates through a service mesh sidecar proxy that enforces mutual TLS, meaning both the calling and receiving service must present a valid certificate issued by an internal certificate authority. This directly addresses the lateral movement risk described in Section 3.3, because a compromised credential or process outside the mesh cannot establish a connection to a mesh-protected service without a valid certificate, regardless of network-layer reachability.

8.3 Envelope Encryption as a Shared Library

Rather than allowing each service team to implement key wrapping and unwrapping independently, the framework provides a shared envelope encryption library that every in-scope service consumes. Centralizing this logic ensures consistent algorithm selection, consistent key identifier resolution, and a single location to patch cryptographic implementation issues, rather than requiring a coordinated update across dozens of independently maintained services.

9. Monitoring, Audit, and Detection

Encryption and access governance controls require continuous verification, because a control that is correctly configured today can drift out of compliance as services evolve. The framework's monitoring layer draws on four AWS-native services: AWS CloudTrail for a comprehensive audit trail of every application programming interface call, including key management and access management operations; Amazon GuardDuty for continuous threat detection across account activity, network traffic, and DNS queries; AWS Macie for content-aware

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

discovery of protected health information in Amazon S3, described in Section 7; and AWS Security Hub as an aggregation layer that consolidates findings from the other three services alongside AWS Config rule evaluations into a single prioritized view.

Figure 3 summarizes the implementation status of the HIPAA Security Rule's administrative and technical safeguard categories against a four-stage maturity scale, not started, partial, implemented, and optimized, as observed in the representative platform evaluated for this article. Access control, transmission security, and device and media controls had reached optimized status at the time of review, while contingency planning and integrity controls remained at partial maturity, reflecting incomplete automated testing of backup restoration and data integrity verification procedures.

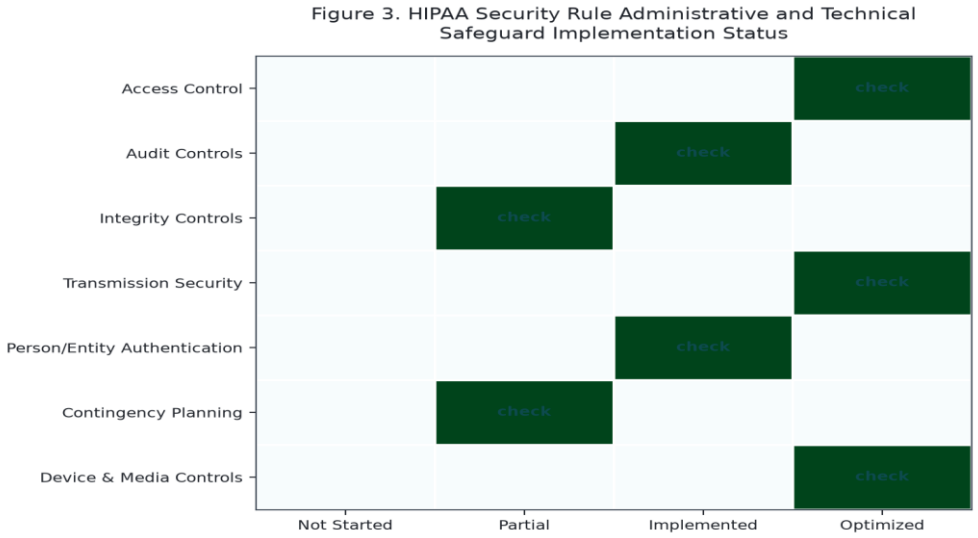

Figure 3. HIPAA Security Rule administrative and technical safeguard implementation status.

Detection findings from GuardDuty and Security Hub are routed to a dedicated incident response queue with severity-based service level objectives for triage, and any finding tagged as touching a restricted or high classification data domain automatically pages the on-call security engineer regardless of the severity score assigned by the underlying detection service, reflecting the elevated consequence of a protected health information exposure relative to a general infrastructure finding.

10. Compliance Mapping

The controls described in Sections 4 through 9 map onto several overlapping compliance and assurance frameworks that healthcare platforms are commonly required to satisfy. Table 4 summarizes this mapping at a framework level, showing the primary controls in this article's framework that satisfy representative requirements from each external framework.

External Framework	Representative Requirement	Primary Controls in This Framework
HIPAA Security Rule (45 CFR 164.312)	Access control, audit controls, transmission security	Sections 4, 6, 9
NIST SP 800-66 Rev. 2	Cybersecurity control mapping for the Security Rule	Sections 4 through 9
NIST SP 800-53 Rev. 5	Access control (AC) and system/communications protection (SC) families	Sections 4, 6, 8
NIST SP 800-207 (Zero Trust)	Continuous authentication and least-privilege access	Sections 6, 8
SOC 2 Trust Services Criteria	Logical access controls, change management, monitoring	Sections 6, 9, 13
HITRUST CSF	Encryption, key management, and access control domains	Sections 4, 5, 6

Table 4. Compliance framework mapping to reference framework controls.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

This mapping is intended as a starting point for a formal compliance gap assessment rather than a substitute for one. Each external framework carries its own audit evidence requirements, and a control that is technically implemented, such as encryption at rest, still requires accompanying documentation, such as key management policy records and access review logs, before it can be presented as satisfied evidence in a formal audit.

11. Case Evaluation: Encryption Coverage and Residual Risk

To evaluate the joint relationship between encryption coverage and access scope described in Section 1, we modeled residual risk as a function of two variables observed across services in the reviewed platform: access scope breadth, a normalized measure of how broadly a service's granted permissions extend beyond its functional need, and encryption and key isolation strength, a normalized measure of how tightly a service's data encryption keys are scoped and rotated. Figure 4 shows the resulting risk surface, rendered as a three-dimensional surface plot so that the interaction between the two variables, rather than either variable alone, is directly visible.

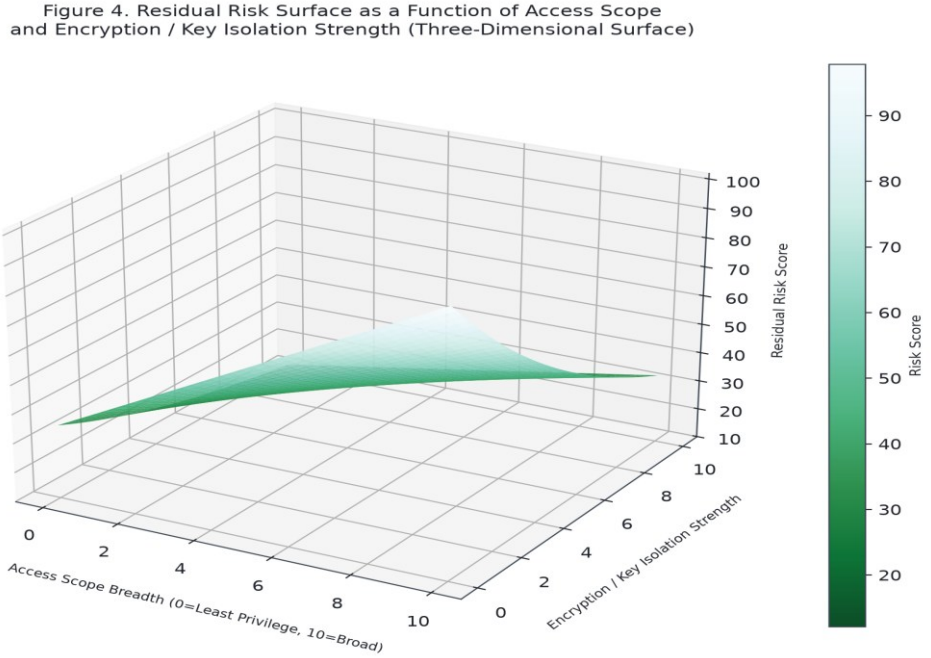

Figure 4. Residual risk surface as a function of access scope and encryption / key isolation strength, shown as a three-dimensional surface.

The surface in Figure 4 illustrates the central finding of this evaluation: residual risk falls sharply as encryption and key isolation strength increases, but only up to a point, beyond which a broad access scope continues to sustain meaningfully elevated risk almost regardless of encryption strength. In practical terms, a service with a narrowly scoped key but broad decrypt permission across many records retains substantial risk, because the encryption boundary and the access boundary have not been jointly reduced. This is the empirical basis for treating encryption and access governance as a joint optimization rather than sequencing one after the other.

Figure 8 summarizes the root cause distribution of protected health information exposure incidents reviewed across the case population, rendered as a three-dimensional pie chart with a side legend identifying each root cause and its share. Misconfigured access policy and overly broad Identity and Access Management roles together accounted for slightly more than half of reviewed incidents, exceeding the combined share attributable to unencrypted legacy stores and credential exposure, which supports prioritizing access scope reduction alongside, rather than after, encryption remediation.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

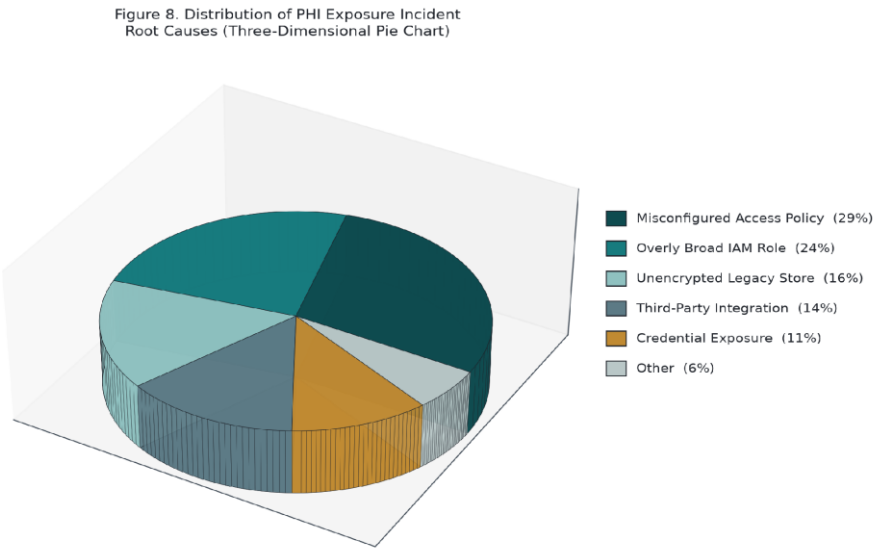

Figure 8. Distribution of PHI exposure incident root causes across reviewed case population, shown as a three-dimensional pie chart.

11.1 Governance Maturity Assessment

Figure 5 presents a governance maturity assessment across six domains, comparing the platform's maturity at the start of the initiative described in this article against the target maturity level defined by the governance committee described in Section 13. The largest maturity gaps at initiative start were in data classification and third-party risk, both of which lacked systematic tooling prior to the adoption of the Macie-based classification workflow described in Section 7.

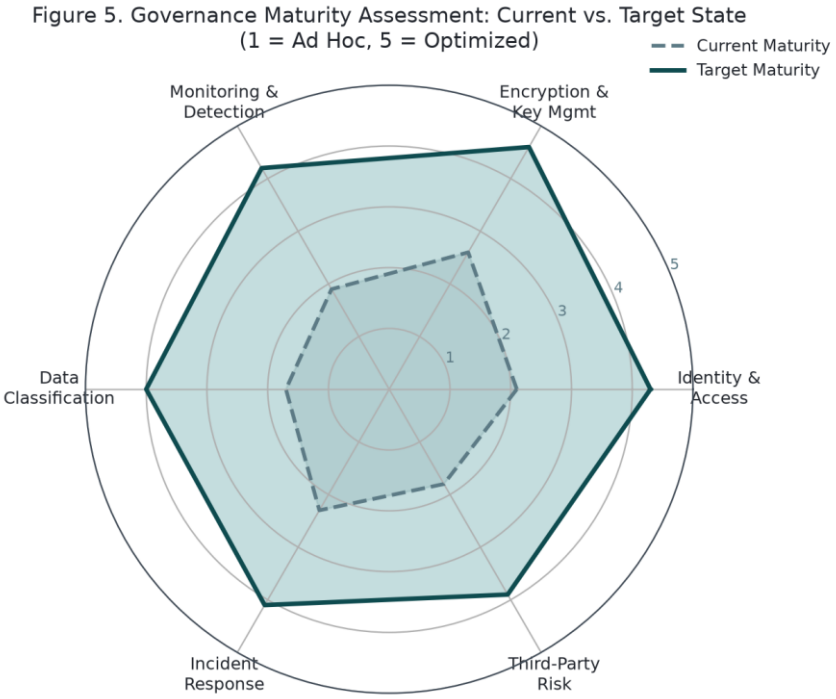

Figure 5. Governance maturity assessment across six domains, current versus target state.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

12. Operational Overhead and Performance Analysis

Every encryption and access control layer introduces some processing overhead, and a credible framework must account for that overhead rather than treating security controls as costless. Figure 7 tracks measured request latency at the fiftieth, ninety-fifth, and ninety-ninth percentiles across four progressive phases of control adoption, rendered as a three-dimensional bar chart so that phase and percentile can be compared simultaneously: a baseline with only default storage-layer encryption, the addition of universal internal Transport Layer Security, the addition of envelope encryption for field-level data, and the addition of hardware security module-backed signing for the highest-assurance operations.

Figure 7. Request Latency Overhead by Migration Phase and Percentile (Three-Dimensional Bar Chart)

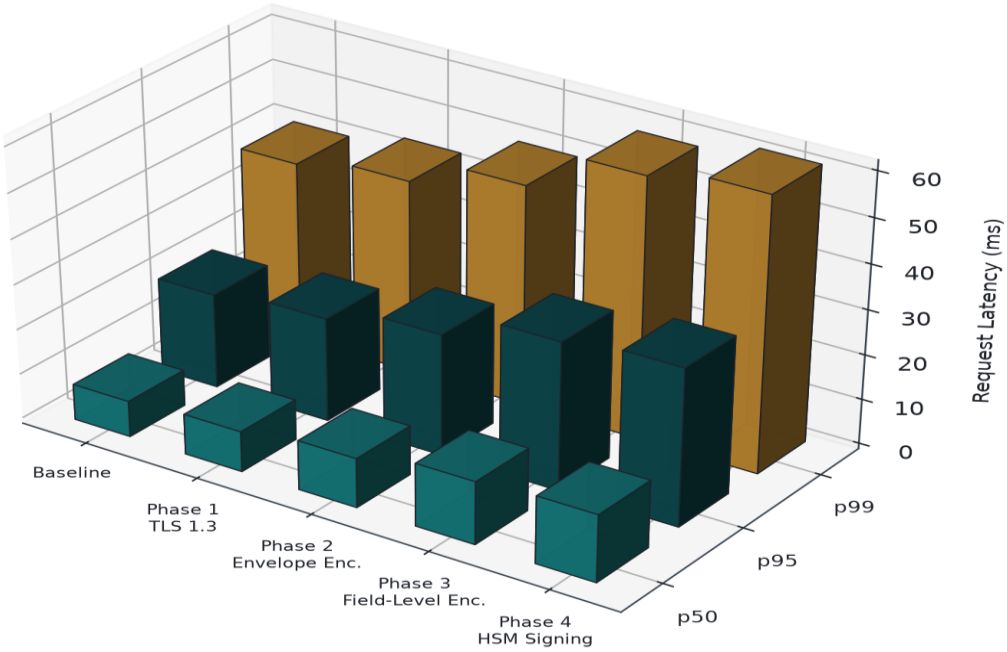

Figure 7. Request latency overhead introduced by progressive encryption controls, by migration phase and percentile, shown as a three-dimensional bar chart.

The largest single increment in tail latency accompanied the addition of field-level envelope encryption, reflecting the cost of per-field key unwrap operations for records with many encrypted attributes. This cost was reduced substantially, though not shown as a separate phase in Figure 7, by introducing a short-lived in-memory cache of unwrapped data encryption keys scoped to the lifetime of a single request, which avoided repeated key management service calls for records with multiple encrypted fields while still re-requesting the key for every new request rather than caching across requests.

Encryption in use through Nitro Enclaves, scoped narrowly as described in Section 4.3, was excluded from the latency comparison in Figure 7 because the small number of workloads using enclaves are batch-oriented rather than latency-sensitive request-response paths, and a percentile-latency comparison would not meaningfully characterize their performance profile.

13. Governance Operating Model

Technical controls degrade without an operating model that assigns clear ownership for their maintenance. The framework establishes a data protection governance committee composed of security engineering, platform engineering, compliance, and clinical operations representation, meeting on a fixed monthly cadence with authority to approve exceptions to the standard key scoping and access governance rules described in Sections 5 and 6.

Each new microservice that will hold or process protected health information passes through a lightweight architecture review before its first production deployment, confirming that its planned data stores are covered by

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

an appropriate customer-managed key, that its Identity and Access Management role is scoped to a specific data domain rather than a broad managed policy, and that its classification tags are defined in its infrastructure-as-code templates prior to deployment rather than added retroactively.

Quarterly entitlement reviews, described in Section 6.4, and the recurring Macie-based classification sweep described in Section 7 both report their findings to the governance committee, which tracks remediation of flagged items against a fixed service level objective and escalates overdue remediations to platform engineering leadership. This operating model is intended to keep the technical framework in Sections 4 through 9 from decaying silently as the platform's service inventory grows.

14. Limitations

Several limitations bound the conclusions of this article. The case evaluation in Section 11 draws on a single distributed healthcare platform, and while the general pattern of risk concentrating in overly broad access scope rather than encryption gaps alone is consistent with broader industry incident reporting, the specific risk surface shown in Figure 4 is a fitted approximation calibrated to one environment's observed incident data and should not be read as a universal quantitative model.

The performance overhead figures in Section 12 reflect a specific workload mix and instance configuration and will not transfer precisely to a platform with a substantially different request profile, particularly one with a much higher proportion of large, multi-field encrypted records than the representative platform evaluated here.

The framework also presumes an organization with sufficient platform engineering capacity to build and maintain a shared envelope encryption library and a service mesh layer, described in Section 8. Smaller organizations without this capacity may need to substitute a subset of managed AWS services for components this article treats as custom-built, which would change the specific operational overhead profile without necessarily changing the underlying architectural principles.

Finally, this article addresses technical and architectural controls and their governance operating model, but does not address the organizational change management required to shift service teams from a broad, self-managed permission model to the centrally governed, least-privilege model described in Section 6, which in practice is often the longer and more difficult half of an initiative of this kind.

15. Future Work

- Extending the residual risk model in Section 11 with data from additional distributed healthcare platforms to test whether the joint relationship between access scope and encryption strength holds across different architectural styles and organizational sizes.
- Automating architecture review, described in Section 13, so that key scoping, access role scope, and classification tagging are verified continuously through policy-as-code checks in the deployment pipeline rather than through a manual review gate at initial deployment.
- Evaluating confidential computing approaches beyond Nitro Enclaves for a broader set of latency-sensitive workloads as hardware-based encryption-in-use technology matures and its performance overhead declines.
- Extending the third-party data flow visibility described in Section 3.4 with automated data flow mapping tools capable of tracing protected health information propagation into partner and analytics systems without relying solely on manual integration documentation.
- Developing standardized audit evidence templates that map directly from the technical controls in Sections 4 through 9 to the specific evidentiary requirements of the compliance frameworks summarized in Table 4, reducing the manual effort currently required to prepare for external audits.

16. Conclusion

This article presented a reference framework for encryption and access governance intended to protect protected health information within distributed AWS microservice architectures. The framework treats encryption across data states, key management architecture, identity-centric access governance, data classification, and continuous monitoring as interdependent components of a single control surface rather than as independent workstreams. A retrospective case evaluation across a distributed healthcare platform showed that encryption coverage alone is a weak predictor of residual risk unless paired with disciplined reduction of access scope, and that overly broad access permissions, rather than unencrypted storage, were the leading root cause among reviewed protected health information exposure incidents.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
Peer-Reviewed | Open Access | International Journal
https://ijetrm.com/

The layered reference architecture, governance operating model, and compliance mapping described in this article are offered as a starting framework rather than a finished specification. Every distributed platform carries its own service inventory, risk tolerance, and engineering capacity, and the specific configuration of key scoping, service mesh enforcement, and entitlement review cadence should be adapted accordingly. The central claim of this article is narrower and, we believe, more durable: encryption and access governance must be designed and evaluated together, because a distributed architecture that is strong on one dimension and weak on the other still leaves protected health information exposed through the dimension left unaddressed.

REFERENCES

1) AICPA. SOC 2 Trust Services Criteria for Security, Availability, Processing Integrity, Confidentiality, and Privacy. American Institute of CPAs, 2022 revision.
2) Amazon Web Services. AWS Key Management Service Cryptographic Details. AWS Whitepaper, 2023.
3) Amazon Web Services. AWS Macie User Guide. AWS Documentation, 2023.
4) Amazon Web Services. AWS Well-Architected Framework: Security Pillar. AWS Documentation, 2023.
5) Amazon Web Services. Amazon GuardDuty User Guide. AWS Documentation, 2023.
6) Amazon Web Services. Architecting for HIPAA Security and Compliance on Amazon Web Services. AWS Whitepaper, 2022.
7) Barker E. Recommendation for Key Management: Part 1, General. NIST Special Publication 800-57 Part 1 Revision 5. National Institute of Standards and Technology, 2020.
8) Bertino E, Ferrari E. Big data security and privacy. In: Big Data Technologies and Applications. Springer, 2018:425 to 439.
9) Cloud Security Alliance. Top Threats to Cloud Computing: Pandemic Eleven. CSA, 2022.
10) Gilman E, Barth D. Zero Trust Networks: Building Secure Systems in Untrusted Networks. Sebastopol, CA: O'Reilly Media, 2017.
11) HITRUST Alliance. HITRUST CSF Version 11.2. HITRUST Alliance, 2023.
12) Hu VC, Kuhn DR, Ferraiolo DF, Voas J. Attribute-based access control. IEEE Computer. 2015;48(2):85 to 88.
13) IBM Security. Cost of a Data Breach Report 2023. IBM Corporation, 2023.
14) Kindervag J. No More Chewy Centers: Introducing the Zero Trust Model of Information Security. Forrester Research, 2010.
15) National Institute of Standards and Technology. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publication 800-162. 2014.
16) National Institute of Standards and Technology. Implementing the HIPAA Security Rule: A Cybersecurity Resource Guide. NIST Special Publication 800-66 Revision 2. 2024.
17) National Institute of Standards and Technology. Security and Privacy Controls for Information Systems and Organizations. NIST Special Publication 800-53 Revision 5. 2020.
18) Newman S. Building Microservices: Designing Fine-Grained Systems. 2nd ed. Sebastopol, CA: O'Reilly Media, 2021.
19) Ponemon Institute. Cost of Insider Threats: Global Report. Ponemon Institute, 2022.
20) Richardson C. Microservices Patterns: With Examples in Java. Shelter Island, NY: Manning Publications, 2018.
21) Rose S, Borchert O, Mitchell S, Connelly S. Zero Trust Architecture. NIST Special Publication 800-207. National Institute of Standards and Technology, 2020.
22) Sandhu R, Coyne EJ, Feinstein HL, Youman CE. Role-based access control models. IEEE Computer. 1996;29(2):38 to 47.
23) U.S. Department of Health and Human Services, Office for Civil Rights. Guidance on HIPAA and Cloud Computing. HHS, 2016.
24) U.S. Department of Health and Human Services, Office for Civil Rights. HIPAA Security Rule, 45 CFR Part 164 Subpart C. Federal Register, 2013 Omnibus Rule.
25) Verizon. 2023 Data Breach Investigations Report. Verizon Business, 2023.

