SECURITY ARCHITECTURE WHITEPAPER · SEC-001

LionSilica

Security Architecture of the LionSilica Platform

AWS-NATIVE SECURITY CONTROLS, EVIDENCE IMMUTABILITY, AND DATA SOVEREIGNTY ARCHITECTURE

Command Center — Design Concept

This paper describes the security architecture of the LionSilica industrial intelligence platform — the controls implemented to protect customer data, ensure evidence immutability, govern identity and access, and enable deployment within regulated and sovereign environments. LionSilica is built natively on AWS and inherits the AWS shared responsibility model as its infrastructure security foundation. Application-layer controls are implemented on top of it. The architecture is designed to support SOC 2 audit readiness and to accommodate the security requirements of enterprise, regulated-sector, and sovereign customers.

CONTENTS

Table of Contents

Document scope notice. This document describes the platform security architecture. No compliance certifications are currently held. Architecture documentation is available for review by qualified prospects under appropriate agreement.

DOCUMENT INFORMATION

Document ID	SEC-001	**Version**	1.0
Classification	Public	**Date**	July 2026
Audience	CISOs, enterprise security architects, cloud security teams, regulated sector security leads		
Contact	LionSilica@lhred.com · lhred.com		

1. Introduction and Scope

This document describes the security architecture of the LionSilica industrial intelligence platform as of July 2026. It covers the AWS-native infrastructure security controls inherited through the shared responsibility model, the application-layer security controls implemented by LionSilica, the approach to evidence immutability and the audit trail, identity and access management architecture, and the data sovereignty and deployment options available for regulated and sovereign customers. The intended audience is security professionals — CISOs, enterprise security architects, cloud security teams, and regulated-sector security leads — who require a technical account of how the platform is secured.

This document does not contain: penetration test results (not published in this document — available under appropriate agreement to qualified prospects), findings from compliance audits (no third-party compliance audits have been completed), internal infrastructure schematics (classified as internal documents not distributed publicly), or vulnerability disclosures. Nothing in this document should be read as a claim of certification, authorization, or validated compliance status.

The security design philosophy of the LionSilica platform treats security as a first-class architectural concern, not a layer applied after the platform is built. Encryption, access control, evidence immutability, and tenant isolation are structural properties of the architecture — they are not add-on configurations. This distinction matters operationally: security controls that are structural are substantially harder to inadvertently bypass than controls that are applied as policies over an insecure substrate. Every design decision that touches data handling, access, identity, or audit trail is evaluated through a security lens from inception.

LionSilica is built natively on AWS and operates within the AWS shared responsibility model. Under that model, AWS is responsible for the security of the cloud — the physical infrastructure, hypervisors, host operating systems, network hardware, and data center facilities. LionSilica is responsible for security in the cloud — application code, IAM configuration, data classification and encryption key management, network configuration, and customer data handling. In customer-controlled and sovereign deployment models, the customer additionally takes direct responsibility for the configuration of their own AWS account environment, including IAM policies, VPC architecture, and access control governance. Section 2 of this document sets out the responsibility allocation in detail.

2. AWS Shared Responsibility Model

The AWS shared responsibility model defines the division of security obligations between AWS, LionSilica, and the customer. This division is fundamental to understanding the platform's security posture and to scoping any security review or audit of the platform.

2.1 What AWS Manages

AWS is responsible for the security of the underlying cloud infrastructure. These controls are covered by AWS's own compliance certifications, including AWS SOC 2, ISO 27001, and FedRAMP authorizations at the infrastructure level. Customers and LionSilica inherit these infrastructure controls as a baseline.

AWS-MANAGED CONTROL	SCOPE
Physical data center security	Physical access, environmental controls, facility security — fully managed by AWS
Hypervisor and host OS security	Patching, hardening, and isolation of the underlying compute infrastructure
Network infrastructure security	AWS backbone network, DDoS mitigation at infrastructure level, network hardware
Hardware lifecycle management	Secure hardware decommissioning and storage media destruction
Availability zone and region resilience	Physical redundancy and separation of availability zones

2.2 What LionSilica Manages

LionSilica is responsible for all security controls at the application and configuration layer within the AWS environment it operates.

LIONSILICA-MANAGED CONTROL	APPROACH
Application code security	Secure development practices, dependency scanning, secrets management, CI/CD pipeline controls
IAM configuration	Least-privilege IAM policies; role-based access; no standing over-privileged credentials
Data classification and encryption key management	KMS key configuration; encryption of sensitive data at rest and in transit
Network configuration	VPC design, security group rules, NACLs, subnet segmentation
Identity and access management for users	Amazon Cognito configuration, RBAC, MFA enforcement, session management
Application-layer access controls	Tenant isolation, data access scoping, authorization enforcement in application logic
Customer data handling and retention	S3 Object Lock COMPLIANCE mode for evidence; retention policy enforcement

2.3 What the Customer Manages (Customer-Controlled Deployments)

In customer-controlled and sovereign deployment models, where the platform is deployed within the customer's own AWS account, the customer assumes direct responsibility for additional controls:

- IAM policies within their AWS account — including policies governing LionSilica service roles if any are granted
- VPC configuration, subnet design, and network access controls within their account
- User identity and authentication management, including enforcement of MFA policies for their personnel
- Access control policy configuration within the platform, scoped to their organizational requirements
- Data export policy, retention settings, and backup governance within their account

3. Data Classification and Handling

The platform operates with four defined data classification tiers. Each tier carries specific handling requirements that govern encryption, access controls, retention, and customer data rights. Classification drives how data is stored, who can access it, and what protections apply.

CLASSIFICATION	EXAMPLES	HANDLING REQUIREMENTS
Public	Marketing materials, documentation, published API schemas, this document	No access controls required; published at lhred.com. Not subject to encryption or access restrictions.
Customer Operational Data	Transaction records, counterparty data, evidence documents, workflow state, Deal Room communications	Encrypted at rest (AWS KMS) and in transit (TLS). Tenant-isolated in storage and application layers. Customer-owned data — customer retains export rights. Retained per customer-configured retention policy.
Evidence Records	AI recommendations with evidence basis, human approval records, executed actions, ingested evidence documents, policy exceptions	Classified as Customer Operational Data with additional immutability controls. Written to AWS S3 with Object Lock in COMPLIANCE mode. Cannot be modified or deleted after the retention period is set — not even by LionSilica or the AWS root account within the retention period.
Internal System Data	Platform configuration, model routing logic, internal architecture documentation, infrastructure credentials	Access restricted to LionSilica engineering personnel. Not accessible to customers or external parties. Subject to internal access controls and MFA requirement for LionSilica personnel.

3.1 Data Residency

In shared SaaS deployments, customer operational data resides in the LionSilica-operated AWS region. Customers may request a specific AWS region at the time of account provisioning; the platform is deployable to any AWS region where the required services are available. LionSilica does not currently transfer customer operational data across regional boundaries except where required for backup or disaster recovery within the same deployment agreement.

In single-tenant, customer-controlled, and sovereign deployment models, data resides entirely within the customer's designated AWS account and region. In sovereign deployments, the platform may be deployed to national cloud infrastructure or a customer-designated AWS region with data residency within a national boundary. The applicable residency constraints are defined in the deployment agreement. Data does not leave the designated environment without explicit customer action or instruction.

4. Encryption Architecture

4.1 Encryption at Rest

All customer operational data and evidence records are encrypted at rest. The encryption method varies by service:

SERVICE	ENCRYPTION METHOD
Amazon RDS (PostgreSQL)	AWS KMS envelope encryption. Encryption key managed by LionSilica (default) or customer-managed CMK (dedicated deployments).
AWS S3 (evidence store)	AES-256 server-side encryption via AWS KMS. Object Lock applied in COMPLIANCE mode for evidence records.
AWS DynamoDB	AWS-managed encryption at rest by default. KMS CMK available in dedicated deployments.
Amazon Cognito user data	AWS-managed encryption. User credential and token storage handled by Amazon Cognito's built-in security controls.

4.2 Encryption in Transit

All data in transit between clients and platform services, and between internal platform services, is encrypted. API endpoints enforce TLS 1.2 as the minimum protocol version; TLS 1.3 is supported and preferred where client implementations support it. WebSocket connections used by the Deal Room real-time messaging layer use WSS (WebSocket Secure). Internal service-to-service communication within the AWS VPC uses TLS where applicable to the service pair.

4.3 Key Management

In shared multi-tenant SaaS deployments, AWS KMS keys are managed by LionSilica. Key rotation is configured per AWS KMS key rotation policy. All key usage is logged and auditable via AWS KMS API call logs. Keys are scoped per data tier — evidence store keys, operational database keys, and backup keys are maintained separately to limit the blast radius of any key compromise.

In single-tenant, customer-controlled, and sovereign deployments, customers may provide their own KMS Customer Managed Key (CMK). When a CMK is used, the customer controls key creation, rotation policy, key access policies, and key deletion. LionSilica's service roles are granted only the minimum KMS permissions required to perform encryption and decryption operations. Revocation of the CMK by the customer effectively renders the encrypted data inaccessible — including to LionSilica.

4.4 Customer-Managed Keys

The customer-managed key option is a significant governance control: it gives the customer cryptographic control over their data environment. A customer that revokes their KMS CMK revokes all access to data encrypted under that key — including LionSilica's operational access. This is a meaningful data sovereignty control for regulated-sector and sovereign customers.

Customers choosing to use CMKs should be aware of the operational implications: inadvertent key deletion or scheduled key deletion will result in data becoming permanently inaccessible after the key deletion window expires. AWS KMS provides a 7–30 day deletion waiting period during which deletion can be cancelled. Customers are responsible for their own key lifecycle governance in customer-controlled deployment models.

5. Identity and Access Management

5.1 User Authentication

Amazon Cognito provides identity management and authentication for users within the Deal Room environment. Cognito handles user registration, authentication flows, password policy enforcement, and multi-factor authentication. MFA is supported and its enforcement is configurable per customer deployment. JWT tokens are issued by Cognito upon successful authentication and are used for session management throughout the platform. Token expiry and refresh policies are enforced to limit the window of exposure from a compromised token. Token revocation is supported for active session invalidation.

5.2 Role-Based Access Control

The platform implements role-based access control (RBAC) throughout. Roles are defined per customer deployment and are configurable by customer administrators to align with their organizational structure and approval hierarchies. Permissions are scoped to the minimum required for the function of each role — a user assigned to a workflow review role cannot access counterparty management functions, for example, unless that function is explicitly included in their role definition.

Cross-tenant access is architecturally prevented. In multi-tenant deployments, the data layer enforces tenant ID scoping on all queries; the application layer enforces role and permission checks. No combination of valid credentials issued to Tenant A can produce access to data belonging to Tenant B through normal application paths.

Role assignments and changes are logged in the platform audit trail. Administrative changes to role definitions are subject to access controls that restrict who within a customer's account can modify role configurations.

5.3 Service-to-Service Authentication

AWS IAM roles are used for service-to-service authentication within the platform. Each Lambda function, container, and service component is assigned an IAM role with the minimum permissions required for its specific function. No shared credentials are used; no long-lived access keys are issued to service components. IAM role assumption is the standard authentication mechanism for all AWS service interactions within the platform. Role assumption events are logged via AWS CloudTrail where CloudTrail is active (planned for full activation in production — see Section 15 for known limitations).

5.4 Administrative Access

LionSilica engineering access to production systems is restricted to named personnel who have been authorized for production access. MFA is required for all administrative access to production environments. All access to production infrastructure is logged. The number of personnel with production access is limited to what is operationally necessary.

In customer-controlled deployment models, LionSilica does not have access to the customer's AWS account unless the customer explicitly grants it. Any access granted by the customer for support or deployment purposes is scoped to specific roles with limited permissions, and can be revoked by the customer at any time. LionSilica does not retain standing access to customer-controlled environments.

6. Evidence Immutability and Audit Trail

6.1 S3 Object Lock: COMPLIANCE Mode

Evidence records in the LionSilica platform are stored in AWS S3 with Object Lock configured in COMPLIANCE mode. COMPLIANCE mode is the most stringent retention protection available in AWS S3: once a retention period is set on an object, the object cannot be modified, overwritten, or deleted by any user — including the AWS account root user — until the retention period expires. This is a fundamentally different guarantee from GOVERNANCE mode, which allows privileged administrators to override the retention lock before expiry.

The choice of COMPLIANCE mode over GOVERNANCE mode is deliberate. Evidence records in the LionSilica platform — AI recommendations, human approval records, executed actions — carry operational and potentially legal significance. The integrity guarantee must be unconditional: an auditor reviewing an evidence record must be able to rely on the fact that the record cannot have been altered after its creation, regardless of who holds administrative privileges on the system. COMPLIANCE mode provides that guarantee.

It is important for customers to understand the operational implications of COMPLIANCE mode: retention periods, once set, cannot be shortened. Customers should configure retention periods that reflect their actual retention requirements, as the platform does not provide a mechanism to delete records early once COMPLIANCE mode is active.

6.2 What Is Written to Immutable Storage

RECORD TYPE	RETENTION SETTING
AI recommendations with evidence basis	Customer-configurable retention period
Human approval records (approver, decision, timestamp)	Customer-configurable retention period
Executed actions (communication dispatches, document releases)	Customer-configurable retention period
Evidence documents ingested	Customer-configurable retention period
Policy exceptions	Customer-configurable retention period

6.3 Audit Trail Queryability

The immutable evidence store is queryable through the platform's audit trail interface. Authorized users can query evidence records by entity, transaction, workflow, date range, approver identity, and action type. The audit trail is designed to support internal audit, compliance review, and dispute resolution use cases — enabling an auditor to reconstruct the full sequence of events for a specific transaction or workflow, including every recommendation made, every human decision taken, and every action executed.

Query access to the audit trail is subject to RBAC controls — not all users have audit trail query access. Administrative and compliance roles have query access scoped to their organizational scope. In customer-controlled deployments, the customer controls which roles have audit trail access and to what scope.

6.4 The Security Implications of Immutability

Immutability is a significant security control: it means that an adversary who gains access to the platform cannot alter, delete, or suppress evidence records that were written before the compromise. A post-compromise review can rely on the integrity of historical evidence records. This property is particularly important in regulated-sector and sovereign contexts where the integrity of records has operational and legal standing.

Immutability does not protect against compromise at the time of write. If an adversary has compromised the platform at the point when a record is being written, the written record may reflect the compromised state. Immutability guarantees that what was written cannot be changed afterward — it does not validate the accuracy of what was written. Detection and prevention controls (access control, monitoring, application security) operate on the write path; immutability operates on the post-write integrity guarantee.

7. Network Security

7.1 VPC Architecture

All platform services are deployed within an AWS Virtual Private Cloud (VPC). The VPC is designed with public and private subnet segmentation. Public-facing services — specifically Amazon API Gateway and Amazon CloudFront — are accessible from the internet as required for platform operation. All other internal platform services — Amazon RDS, AWS DynamoDB, internal Lambda functions, and processing services — are deployed in private subnets with no direct public internet access.

Traffic from the public internet to the platform passes through the API Gateway and CloudFront layers, where authentication is enforced before any request reaches internal services. There is no direct network path from the public internet to internal data stores. Internal services communicate with each other within the private subnet, with network access controlled by security group rules.

7.2 Security Groups and Network ACLs

Each service component is protected by a security group with explicitly defined inbound and outbound rules. Security group rules follow the principle of least access: inbound rules permit only the traffic required for the service to function, from the specific sources (other security groups or CIDR ranges) that legitimately need access. There are no 0.0.0.0/0 inbound rules on any internal service security group. All other traffic is denied by default.

Network Access Control Lists (NACLs) provide an additional layer of subnet-level network filtering. NACL rules are configured to deny access to internal subnets from sources that should not reach those subnets at the network level, as a defense-in-depth measure supplementing security group controls.

7.3 WAF and DDoS Protection PLANNED

AWS WAF is on the architecture roadmap for production deployment. When active, AWS WAF will be configured on the API Gateway and CloudFront distributions to provide API-layer filtering, rate limiting, and protection against common web application attack patterns. WAF rule configuration for the platform's specific API endpoints is planned as part of the production hardening phase.

AWS Shield Standard is active for all AWS-hosted endpoints by default. AWS Shield Standard provides protection against common, volumetric DDoS attacks at the network and transport layers for all AWS services. AWS Shield Advanced is an option available in customer-controlled deployments where enhanced DDoS response capabilities are required; this is not standard in the shared deployment model.

7.4 Threat Detection PLANNED

AWS GuardDuty is on the architecture roadmap. When active, GuardDuty will provide continuous, automated threat detection across the AWS account — analyzing CloudTrail management event logs, VPC Flow Logs, and DNS logs to identify anomalous behavior patterns associated with account compromise, credential exfiltration, unusual API calls, and network-level threats. GuardDuty findings will feed into the platform's incident response process (Section 13). The absence of GuardDuty in the current production configuration is a known limitation documented in Section 15.

8. Application Security

8.1 Build Pipeline Security

Security controls are integrated into the CI/CD build pipeline rather than applied as a post-build review. Automated security testing runs as part of the build process on every code change. Dependency vulnerability scanning identifies known CVEs in third-party libraries before any code is deployed to a production environment. The build pipeline is configured to fail and block deployment when critical security findings are detected, preventing the promotion of code with known critical vulnerabilities.

Secrets management is enforced in the build pipeline: no credentials, API keys, or access tokens may be hardcoded in source code or committed to the repository. Automated secrets scanning runs on all code changes to detect and block accidental credential exposure. All secrets required by the platform at runtime are managed through AWS Secrets Manager or AWS Parameter Store and are injected at runtime rather than embedded in build artifacts.

Container images used in the platform are scanned for known vulnerabilities as part of the build pipeline. Base images are selected for minimalism and updated on a regular cadence to incorporate upstream security patches. Image signing is on the security roadmap for future implementation.

8.2 API Security

All operational API endpoints require valid authentication. Authentication is enforced at the API Gateway layer before requests are routed to backend services — unauthenticated requests to operational endpoints are rejected at the gateway. There are no unauthenticated endpoints for any functionality that touches customer operational data, evidence records, or workflow state.

Input validation is enforced at API boundaries. All request payloads are validated against defined schemas; malformed or unexpected inputs are rejected before they reach application logic. Rate limiting is applied at the API Gateway level to protect against abuse and brute-force patterns. Response headers are configured to avoid information disclosure.

8.3 Injection and Input Validation

All database interactions use parameterized queries through the Prisma ORM. Direct string concatenation into SQL queries is not used; parameterization is enforced by the ORM layer as a structural control. This eliminates the SQL injection attack surface for database interactions that go through the ORM. Input sanitization is applied to user-supplied data before it is written to any data store or rendered in any user-facing context.

The AI orchestration pipeline processes user-supplied content as inputs to LLM inference calls via Amazon Bedrock. Prompt injection — the attempt to alter model behavior through adversarial inputs — is a recognized attack class in AI-integrated applications. The platform applies structural mitigations in the AI orchestration layer: system prompts and user-supplied content are structurally separated in the model call; content is sanitized before inclusion in prompts; model outputs are validated before being written to any data store or surfaced to users as governed recommendations.

8.4 Dependency Management

All third-party dependencies are tracked and monitored for known vulnerabilities. Automated dependency scanning runs on every build, with results reviewed as part of the release process. Transitive dependency vulnerabilities are included in the scan scope. The platform maintains a policy of addressing critical-severity CVEs on an accelerated timeline, with high-severity findings addressed within the standard release cycle.

Dependencies are pinned to specific versions in production build artifacts to prevent unexpected version changes through dependency resolution. Updates to pinned dependencies are performed deliberately as part of the regular maintenance cycle, not automatically on each build. This approach balances the security benefit of timely updates with the operational stability required for a production platform.

9. Tenant Isolation

9.1 Logical Tenant Isolation

In multi-tenant SaaS deployments, customer data is logically isolated through tenant ID partitioning across all data stores. Every record in the operational database, every object in the evidence store, and every item in the workflow state store carries a tenant identifier. All data access queries — at both the database layer and the application layer — are scoped to the authenticated user's tenant. A request from a valid user of Tenant A cannot return, modify, or reference data belonging to Tenant B through any standard application code path.

The tenant isolation model is enforced at multiple layers. At the database layer, query construction always includes the tenant ID as a mandatory filter. At the application layer, authorization checks validate that the resource being accessed belongs to the requesting user's tenant before any operation is performed. This defense-in-depth approach means that a bug at one layer does not automatically produce a cross-tenant data exposure — both layers would need to fail simultaneously.

9.2 Physical Tenant Isolation

Single-tenant, customer-controlled, and sovereign deployment models provide physical isolation. In these models, a dedicated infrastructure environment is provisioned for the customer — dedicated compute, dedicated storage, and dedicated network resources with no shared infrastructure with any other customer. The platform is deployed to a separate AWS account (in customer-controlled models) or a separate infrastructure environment (in sovereign models).

Physical isolation eliminates the side-channel and data leakage risks inherent in shared infrastructure. It also provides the customer with direct control over the security posture of their environment, including the ability to apply their own network perimeter controls, their own monitoring, and their own access governance independent of any multi-tenant platform controls.

9.3 Tenant Isolation Verification

Tenant isolation is verified as part of the platform's application security testing process. Test scenarios include authenticated cross-tenant data access attempts — where a valid user of one tenant attempts to access data belonging to another tenant through API calls, parameter manipulation, and authentication bypass attempts. These tests are run against the multi-tenant architecture to validate that isolation controls are functioning as designed.

Changes to data access patterns in the application layer — new API endpoints, changes to query construction, changes to authorization logic — are subject to code review with specific attention to tenant isolation implications. Any change that affects the data access layer is reviewed against the tenant isolation model before promotion to production. Regression testing for isolation is included in the test suite for affected components.

10. Build Pipeline Security

10.1 CI/CD Security Controls

The build pipeline implements four categories of automated security control that run on every code change before promotion to any deployment environment:

01

Dependency Vulnerability Scanning

All direct and transitive dependencies are scanned against known CVE databases on every build. Critical-severity findings block promotion to production. Results are logged and reviewed by engineering.

02

Secrets Scanning

Automated scanning detects accidental inclusion of credentials, API keys, tokens, or private keys in source code commits. Detected secrets block the build and trigger immediate review. No secrets are stored in source control.

03

Automated Security Linting

Static analysis tools identify common security anti-patterns — hardcoded values, unsafe function calls, insecure configuration patterns — as part of the standard build lint pass.

04

Container Image Scanning

Container images are scanned for known vulnerabilities in base image layers and installed packages before deployment. Images with critical vulnerabilities in production components are not promoted.

10.2 Release Management

Deployments to production follow a staged process. Changes are promoted through development, staging, and production environments in sequence. The staging environment mirrors the production architecture and is used for integration testing, including security regression tests, before any change reaches production. Changes that modify security-sensitive components — authentication flows, authorization logic, data access patterns, encryption configuration — require explicit review before staging promotion.

All production deployments are reversible. Rollback capability is maintained for all production changes, with rollback procedures tested as part of the release process. The ability to rapidly roll back a deployment is treated as a security control — it limits the window during which a flawed deployment can affect production customers if a post-deployment issue is detected.

10.3 Access Controls for Build Systems

Access to the production deployment pipeline is restricted to a limited set of named personnel. Production deployment capability requires both authentication (named identity) and authorization (deployment role assignment). MFA is required for any access to production deployment tooling. Developer workstations do not have direct access to production environments — all production deployments occur through the controlled CI/CD pipeline, not through individual developer credentials.

Pipeline credentials — the service credentials used by the CI/CD system to interact with AWS — are managed as least-privilege IAM roles scoped to the specific actions required for deployment. These credentials are not accessible to developers. Rotation of pipeline credentials follows a defined schedule. Any changes to pipeline IAM roles or deployment permissions are subject to change authorization review.

11. Blockchain Anchoring (Optional)

> **Optional capability.** Blockchain anchoring is not active by default. It must be explicitly configured and enabled for a deployment. This section describes the capability for customers evaluating whether to enable it.

11.1 What It Is

Blockchain anchoring is an optional platform capability that allows selected evidence records to be anchored to the Polygon blockchain. When anchoring is configured for a record type, the platform computes a cryptographic hash (SHA-256) of the evidence record at the time it is written to the immutable evidence store, and submits that hash as a transaction to the Polygon blockchain. The on-chain transaction records the hash, the timestamp of the anchoring event, and a record identifier.

The anchoring mechanism creates a second, independent tamper-evidence control that operates entirely outside the AWS infrastructure. If the on-chain hash for a record matches the hash of the stored record, the record has not been altered since it was anchored. This is a complementary control to S3 Object Lock COMPLIANCE mode, not a replacement for it. S3 Object Lock prevents modification; blockchain anchoring provides an independent verification path that does not rely on the AWS infrastructure to verify integrity.

11.2 Use Cases

Blockchain anchoring adds security value in specific scenarios where S3 Object Lock alone may not satisfy the trust requirements of all parties:

- **Cross-party dispute resolution** — where a counterparty requires independent tamper-evidence verification that does not rely solely on the record-holding party's own infrastructure
- **Regulatory submissions** — where a regulator or oversight body requires tamper-evidence with independent verification capability not dependent on the submitting party's systems
- **Sovereign contexts** — where reliance on AWS infrastructure alone does not satisfy counterparty trust requirements or where treaty or jurisdictional requirements specify independent verification
- **Legal proceedings** — where establishing the integrity of a record through independent cryptographic verification on a public blockchain may carry evidentiary value

11.3 Limitations

Blockchain anchoring verifies that a record has not been altered since the time of anchoring. It does not verify that the record is accurate, complete, or truthful at the time of creation. A record that was incorrectly written at creation — whether through error or through a compromised write path — will be anchored in its incorrect state and will verify successfully as unaltered. The integrity guarantee is about post-write tamper-evidence, not about the accuracy of what was written.

Blockchain anchoring does not provide confidentiality. Hashes are published to the Polygon blockchain, which is a public ledger. The hash itself does not reveal the content of the record — SHA-256 hashes are not reversible to their source content under normal conditions. However, the fact that an anchoring event occurred at a given time and associated with a given record identifier is visible on the public blockchain. Customers in sensitive contexts should evaluate whether the public visibility of anchoring event metadata is acceptable for their use case before enabling this capability.

12. Compliance Design Intent

12.1 Design for Audit Readiness

The platform security architecture is designed to support SOC 2 audit readiness. Controls implemented — encryption at rest and in transit, access control and least-privilege enforcement, audit logging, change management procedures, evidence immutability, and availability architecture — align with the SOC 2 Trust Service Criteria for Security, Availability, and Confidentiality. No SOC 2 audit has been completed and no SOC 2 report is currently held.

The design-for-audit-readiness approach means that the controls described in this document have been implemented with a view to their eventual auditability. Evidence of control operation — access logs, encryption configuration, change records, IAM policy documentation — is captured and retained in forms suitable for audit review. Customers considering the platform for regulated-sector deployment may request access to architecture documentation and control evidence under appropriate agreement.

12.2 Regulated and Sovereign Sector Deployments

The platform architecture accommodates regulated-sector customers through deployment model flexibility. Customer-controlled and sovereign deployment models allow regulated-sector customers to operate the platform within their own security perimeter — their own AWS account, their own network boundary, their own KMS keys, and their own IAM governance. This means the customer's own compliance program can encompass the platform environment directly, rather than relying on a third-party audit of a shared infrastructure.

Data residency controls, customer-managed encryption keys, tenant isolation, and the immutable audit trail are all controls that are directly relevant to regulated-sector compliance requirements including those applicable to financial services, healthcare data governance, and government information security frameworks. Architecture documentation and controls are available for review by qualified enterprise and government prospects under appropriate agreement.

12.3 What Is Not Claimed

CERTIFICATION OR CAPABILITY	STATUS
SOC 2 Type I or Type II report	NOT CURRENTLY HELD
FedRAMP authorization	NOT CURRENTLY HELD
ISO 27001 certification	NOT CURRENTLY HELD
CMMC certification	NOT CURRENTLY HELD
Penetration test results	*Not published — available under appropriate agreement to qualified prospects*
FIPS 140-2 validated modules (LionSilica application layer)	*AWS-managed services use FIPS-validated modules where available; LionSilica application layer not separately validated*

12.4 AWS Compliance Inheritance

AWS holds SOC 2, ISO 27001, FedRAMP authorization, PCI DSS, HIPAA eligibility, and numerous other certifications and authorizations for its infrastructure services. Customers using AWS services — directly or through platforms built on AWS such as LionSilica — may be able to reference AWS's compliance certifications for the infrastructure layer in their own compliance programs, subject to the terms of the AWS Customer Agreement and applicable AWS compliance programs (such as AWS Artifact for compliance documentation access).

LionSilica's application-layer security controls are additive to the AWS compliance foundation. A customer operating LionSilica within their own AWS account under a customer-controlled deployment model can scope a compliance audit to include both the AWS infrastructure layer (covered by AWS's certifications) and the LionSilica application layer (covered by the controls documented in this whitepaper and supporting documentation available under agreement). LionSilica does not hold certifications on the AWS infrastructure layer — that is covered by AWS's own compliance program.

13. Incident Response Architecture

The incident response architecture is organized across three operational phases:

DETECTION	RESPONSE	RECOVERY
– Amazon CloudWatch monitoring for application errors, anomalous API activity, and infrastructure health – AWS GuardDuty (planned) for account-level threat detection across CloudTrail, VPC Flow Logs, and DNS – Application-level anomaly alerting on authentication failure rates and access pattern deviations	– Incident classification (critical / high / medium / low) based on data exposure risk and operational impact – Containment procedures: isolation of affected services, credential rotation, access revocation – Forensic preservation: snapshot and log preservation before remediation steps that could overwrite evidence – Customer notification per SLA defined in deployment agreement	– Automated backup and restore for RDS and S3 data – RTO and RPO targets are defined per deployment agreement — no universal guarantee is published – Post-incident review and control improvement as standard practice

13.1 Customer Notification

In the event of a confirmed security incident that affects customer data, LionSilica commits to notifying affected customers within the timeframe specified in the applicable deployment agreement. Notification will include a summary of the nature of the incident, the data or systems affected, the containment actions taken, and the recommended customer actions. Notification timelines are defined in the deployment agreement and reflect the requirements of the customer's regulatory environment.

In customer-controlled deployment models, the customer is the primary operator of their environment and is responsible for their own breach notification obligations to their own users, regulators, and counterparties. LionSilica will provide technical assistance in the investigation and remediation of incidents affecting the platform software in customer-controlled environments to the extent agreed in the deployment agreement.

14. Data Sovereignty and Deployment

14.1 Data Residency by Deployment Model

DEPLOYMENT MODEL	DATA RESIDENCY	CUSTOMER DATA CONTROL
Dedicated SaaS	LionSilica AWS region (customer-selected at provisioning)	Data ownership retained by customer; export on request
Single-Tenant Enterprise	LionSilica AWS region (customer-selected), dedicated infrastructure	Full tenant isolation; data ownership retained; dedicated environment
Customer-Controlled AWS Account	Customer's own AWS account and selected region	Full customer control of environment, keys, access, and data
Sovereign Deployment	National cloud boundary or customer-designated AWS region within national boundary	Full data residency within national boundary; customer controls environment
Hybrid / Restricted	Customer-specified per deployment agreement	Per deployment agreement; customized to customer's sovereignty requirements

15. Known Limitations

The following limitations are disclosed in the interest of technical accuracy. Qualified prospects requiring detailed information on the status of any control should contact LionSilica@lhred.com.

1. **WAF and GuardDuty are on the architecture roadmap but are not yet active in production.** Network-layer threat detection and API filtering are planned capabilities. AWS Shield Standard provides baseline DDoS protection; application-level rate limiting is active. Full WAF rule configuration and GuardDuty activation are on the near-term roadmap.
2. **CloudTrail and AWS Config for compliance audit trail are planned.** Application-level audit trail (the immutable evidence store described in Section 6) is active. AWS account-level audit logging via CloudTrail and configuration governance via AWS Config are not yet fully configured for production. Activation of both is planned as part of the production hardening phase.
3. **No compliance certifications are currently held.** The architecture is designed for SOC 2 audit readiness, and controls align with SOC 2 Trust Service Criteria. No third-party compliance audit has been completed. No SOC 2, ISO 27001, FedRAMP, or CMMC certification or authorization is currently held. This is stated clearly in Section 12 and is not qualified.
4. **Penetration testing results are not currently available for external review.** Security testing has been conducted on the platform. Formal penetration test reports are not published in this document and are not distributed publicly. Reports are available for review by qualified prospects under appropriate non-disclosure agreement.
5. **Blockchain anchoring is an optional capability that must be explicitly configured.** Polygon blockchain anchoring of evidence records is not active by default in any deployment model. It must be explicitly configured and enabled. Customers should review Section 11 for a full description of the capability, its use cases, and its limitations before deciding whether to enable it.

16. Document Notes

Document ID	SEC-001	**Version**	1.0
Classification	Public	**Date**	July 2026
Contact	LionSilica@lhred.com · lhred.com		

Architecture and controls described in this document reflect the LionSilica platform as of July 2026. This document is updated as the platform security architecture evolves. The most current version is available at lhred.com or by contacting LionSilica@lhred.com.

AWS Partnership. LionSilica is an AWS Partner. The platform is built natively on AWS services. AWS holds independent compliance certifications for its infrastructure — including SOC 2, ISO 27001, and FedRAMP authorizations — which provide the infrastructure security foundation on which the LionSilica application-layer controls described in this document are implemented. LionSilica does not hold these certifications; AWS holds them for the infrastructure layer.

No certification claim. Nothing in this document constitutes a claim of SOC 2 certification, FedRAMP authorization, ISO 27001 certification, CMMC certification, or any other third-party compliance certification or authorization. The platform security architecture is designed to support audit readiness for these frameworks. No third-party audit has been completed and no certification is currently held.