Security Whitepaper

Architecture, controls, and data protection
for the Prisma Verify asset-integrity platform

Version 1.0 · August 2026
Prisma Verify · prismaverify.ai
Security enquiries: security@prismaverify.ai

1. Purpose and scope

This document describes the security architecture of Prisma Verify, a mechanical-integrity platform used to manage inspection data, corrosion analysis, risk ranking, and fitness-for-service assessment for fixed equipment.

It is written for the security reviewers, IT architects, and compliance teams who evaluate the platform before deployment. It describes **implemented controls** — what the system does today — and deliberately avoids implementation detail that would weaken those controls if published. Where a capability is planned rather than in place, this document says so plainly.

Prisma Verify handles data that is operationally sensitive rather than consumer-personal: equipment condition, thickness measurements, damage mechanisms, inspection findings, and the engineering judgements that follow from them. A compromise of that data is primarily a **safety and regulatory** concern, and the platform is designed around that premise.

2. Deployment models

Prisma Verify runs in three configurations, and the security boundary differs in each.

Model	Description	Typical use
Managed cloud	Operated by Prisma Verify with per-tenant logical isolation.	Fastest path to production.
Self-hosted	Deployed inside customer infrastructure. Customer controls network, backup, and identity.	Data-residency or policy constraints.
Air-gapped	No outbound internet requirement. AI inference runs on local models on customer hardware.	Secured facilities and isolated plant networks.

The air-gapped configuration is a first-class deployment target, not a degraded mode. Field devices operate offline against an encrypted local database and reconcile when connectivity returns.

3. Tenant isolation

Multi-tenant separation is enforced in the **database**, not only in application code.

- Every tenant-scoped table is protected by PostgreSQL **row-level security**, with row-level policies enforced for the application role.
- The application connects as a **non-privileged role** that cannot bypass those policies. This is a deliberate design decision: an application-layer bug cannot silently escalate into cross-tenant data exposure, because the database refuses the read regardless of what the query asks for.

- Tenant context is bound per request and carried through the data-access layer, so queries cannot execute without a resolved tenant scope.
- Tenant identity is derived from the authenticated session server-side. It is never accepted from client-supplied input.

This layering is intentional defence-in-depth: application scoping, database policy, and non-privileged execution each independently prevent cross-tenant access.

4. Identity, authentication, and access control

Authentication

- **Single sign-on** through the customer's existing identity provider over **OIDC** or **SAML**, including Microsoft Entra ID.
- **Multi-factor authentication**, including recovery-code handling for account recovery.
- Sessions use signed, short-lived tokens with asymmetric verification keys. Authentication cookies are HTTP-only, and session material is not exposed to client-side scripts.
- Authentication endpoints are rate-limited to slow credential-stuffing and brute-force attempts.

Authorisation

- **Role-based access control** built on explicit capabilities rather than implicit role names, evaluated **server-side on every request**. A client cannot obtain data by requesting a route the UI does not display.
- Contractors and other external parties receive a **scoped workspace** that exposes only work assigned to them.
- **Administrative impersonation**, where used for support, is time-bounded and written to the audit trail as a distinct, attributable event.

5. Encryption and key management

State	Protection
In transit	TLS 1.2 or higher with modern cipher suites, certificates issued and renewed automatically.
Secrets at rest	Stored credentials — including third-party integration and AI provider keys — are encrypted with **AES-256-GCM**, an authenticated cipher, so tampering is detected rather than silently accepted.
Local device database	Offline field data is held in an **encrypted local database**. Without the key the file is unreadable, which matters for a laptop or tablet carried into a plant.

Key rotation	Encryption keys can be rotated without downtime. Rotation re-encrypts stored secrets transactionally and is recorded in the audit trail.

Secrets are write-only from the user interface: a stored credential can be replaced but never read back through the application.

6. Auditability and evidence integrity

Integrity records are regulatory artefacts. They must be defensible years after they are written.

- Every consequential action is recorded with **actor identity, timestamp, entity, and change detail**.
- Audit records are **hash-chained**, so a reviewer can verify that a record has not been altered after the fact rather than simply trusting the database.
- Inspection evidence supports **content-addressed integrity verification**, allowing an exported evidence package to be checked against the record it claims to represent.
- Governed workflows require explicit human approval to change controlled records; the approval, the approver, and the prior state are all retained.

> **Design principle — no fabricated values.** Where a reading, corrosion rate, or remaining-life figure is not supported by real data, Prisma Verify reports it as unknown. It does not substitute a default, an average, or a generated estimate. In integrity engineering a confident wrong number is more dangerous than a visible gap, and this rule is enforced automatically in the build pipeline.

7. AI governance

Prisma Verify uses AI to accelerate inspector judgement. It does not delegate that judgement.

- **No autonomous safety verdicts.** AI may draft findings, summaries, or recommendations. A qualified human reviews and signs before anything enters the permanent record.
- **Traceable output.** AI responses cite the records and standards they drew from, so a reviewer can check the basis of a suggestion rather than accept a conclusion.
- **Tenant-scoped retrieval.** AI retrieval operates strictly within the requesting tenant's data boundary.
- **Deployment transparency.** The platform distinguishes cloud from on-premises inference at the point of use, so an operator always knows where a given inference ran.
- **Local inference option.** Models can run entirely on customer hardware, so inspection content need never leave the customer network.
- Outbound requests to customer-configured AI endpoints are validated to prevent server-side request forgery against internal network resources.

8. Application and platform security

- **Cross-site request forgery** protection is enforced for state-changing browser requests, and the contract is verified automatically in the build.
- **Server-side request forgery** protection validates externally supplied URLs — including integration and AI endpoints — against internal and link-local address ranges before any request is made.
- **Injection resistance**: data access is parameterised through a typed query layer; natural-language query features are constrained to read-only operations with an explicit table allow-list and are blocked from reaching system catalogues.
- **Error handling** returns generic messages to clients while retaining full diagnostic detail server-side, so internal structure is not disclosed through error responses.
- **Transport and backup security** for self-hosted deployments follow customer policy; the platform does not require inbound internet access to operate.

9. Secure development lifecycle

Security is enforced by the build, not by intention.

Stage	Control
Static analysis	**CodeQL** and **Semgrep** scan every change.
Dependency risk	**Automated dependency review** on all changes; supply-chain actions are pinned to immutable commit hashes rather than mutable tags.
Container security	**Trivy** image scanning with **SBOM** generation.
Secret detection	**Gitleaks** scanning to prevent credential commits.
Artifact integrity	Release artefacts are **cryptographically signed**.
Regression gate	A consolidated gate blocks changes that fail type-checking, linting, the full test suite, access-control regression proofs, or the no-fabricated-data policy.

Access-control behaviour is asserted by automated tests that prove unauthorised requests are denied — a regression in authorisation fails the build rather than reaching production.

10. Data protection and privacy

- **Data minimisation.** The platform collects operational and equipment data. Personal data is limited to what user accounts and inspector attribution require.
- **Right to erasure.** A governed erasure workflow supports data-subject deletion requests with administrative review and an auditable record of the outcome.
- **Data residency.** Self-hosted and air-gapped deployments keep all data in infrastructure the customer controls, in the jurisdiction they choose.
- **Data portability.** Customer data can be exported. The platform is designed without vendor lock-in of customer records.
- **Retention.** Retention of inspection records is customer-configurable to match regulatory retention obligations.

11. Compliance posture

Prisma Verify is transparent about the difference between implemented controls and third-party attestation.

Item	Status
Security architecture and controls described in this document	**Implemented**
SOC 2 Type II	**Readiness programme in progress.** No attestation report has been issued at the date of this document.
ISO/IEC 27001	Not certified at the date of this document.
GDPR data-subject request handling	**Implemented** (erasure workflow, export, minimisation).

The platform is engineered to support **customer** compliance with the inspection codes that govern their assets — API 510, 570 and 653 inspection workflows, API 580/581 risk-based inspection, and API 579 fitness-for-service — with calculations that a reviewer or regulator can retrace to their inputs.

12. Vulnerability disclosure

We welcome coordinated disclosure. Report suspected vulnerabilities to **security@prismaverify.ai**. Please include enough detail to reproduce the issue. We will acknowledge receipt, keep you informed of remediation progress, and credit reporters who wish to be named.