OFFLOAD SECURITY

Platform Whitepaper Pack

Ten module whitepapers covering the unified security & compliance platform

Version 1.0 · July 2026 · Classification: Customer-Shareable

In This Pack

#	Whitepaper	What it covers
1	**Source Code Protection**	How customer repositories and source code are handled, protected, and deleted
2	**Code Security**	SAST, dependencies, secrets, IaC, SBOM, and license compliance from commit to release
3	**Cloud Security Posture Management**	Continuous multi-cloud posture across AWS, GCP, and Azure
4	**Kubernetes Security**	Continuous cluster posture: configuration, workloads, benchmarks, and images
5	**Container & Software Supply Chain Security**	Registry-to-runtime image assurance: vulnerabilities, SBOMs, signatures, and secrets
6	**Unified Vulnerability Management**	One queue, one lifecycle, one history — across every scanning domain
7	**Compliance & Risk Management (GRC)**	Controls, risks, evidence, and BIA connected to live technical findings
8	**DPDP Act Compliance (India)**	The DPDP Act, 2023 and CERT-In Directions, operationalized
9	**Threat Intelligence**	Real feeds, IOC correlation, and MITRE ATT&CK context on your own findings
10	**Integrations & SIEM Connectivity**	Existing tools, one operational layer — including Wazuh SIEM

Each whitepaper stands alone and follows the same structure: Overview, Key Capabilities, How It Works, Data Handling & Security, and how the module feeds the platform's single risk view. All papers describe shipped capabilities as of publication.

OFFLOAD SECURITY

Source Code Protection

Security Architecture Whitepaper

How customer repositories and source code are handled, protected, and deleted

Version 1.0 · July 2026
Classification: Customer-Shareable

Contents

1. Executive Summary

The Offload platform provides application and cloud security scanning — static analysis (SAST), software composition analysis (SCA), secrets detection, infrastructure-as-code (IaC) analysis, and SBOM generation — over customer source code repositories. This document describes, at an engineering level, how customer source code is obtained, processed, protected, and deleted throughout that lifecycle.

The design follows four principles:

- **Ephemeral source access.** Repositories are cloned into an isolated, scan-scoped workspace, scanned locally, and deleted when the scan finishes — on success and on failure. The full source tree is never written to a database or object store.
- **Minimal retention.** Only scan findings and small code excerpts (a few lines around each finding, so engineers can triage in context) are retained — under configurable retention windows and hard-delete APIs.
- **Least-privilege access.** Read-only repository access is sufficient for all scanning. Write permissions are requested only for optional, customer-enabled features such as automated fix pull requests.
- **No third-party exposure.** Scanning runs entirely on platform infrastructure. Customer code is never used to train AI/ML models and is never shared with third parties; optional AI features use the customer's own model-provider API keys.

2. Platform and Deployment Overview

Offload is a multi-tenant security platform composed of a FastAPI application tier, asynchronous scan workers, and a MongoDB data tier. It is offered in two deployment models:

Model	Where scanning runs	Where data resides
SaaS	Offload-managed infrastructure	Offload-managed, tenant-isolated data stores
On-premises / self-hosted	Customer's own environment (containerized deployment)	Entirely within the customer's environment — source code and scan data never leave the customer's network

Customers with strict data-residency or code-egress requirements can adopt the on-premises model, in which the entire scanning pipeline described in this document executes inside their own perimeter.

3. Source Code Lifecycle During a Scan

3.1 Acquisition

When a scan is initiated, the platform performs a shallow git clone (depth 1, single branch, no tags) of the target repository into an isolated workspace directory created exclusively for that scan. Code is

transferred directly from the git provider to the scanning workspace over TLS; it is not proxied through any third-party service. Alternatively, customers may upload a ZIP archive of their code, which is extracted into the same kind of isolated, scan-scoped workspace.

3.2 Clone hardening

The clone operation itself is hardened against both malicious repository content and credential leakage:

- Git hooks are disabled and local file-protocol transports are blocked (a hostile repository cannot execute code on the scanning host through git mechanisms).
- The git process runs with a minimal environment and interactive prompts disabled, so host secrets cannot leak into the scan context.
- Clone targets are validated against SSRF and private-address guards before any network connection is made.
- Operations are time-boxed; access tokens are redacted from all logs and error messages.
- ZIP extraction is protected against path-traversal, with archive size limits enforced during upload.

3.3 Scanning

All scanners run locally against the workspace on platform infrastructure (or customer infrastructure, in on-premises deployments). No source code is transmitted to external scanning services. Section 8 lists the toolchain.

3.4 Automatic deletion

Every scan path — repository clone, ZIP upload, and fix-PR generation — deletes its workspace in a finally-style cleanup that executes on success and on failure alike. Uploaded ZIP archives are likewise deleted after extraction and scanning. The full source tree therefore exists on scanning infrastructure only for the duration of the scan (typically minutes).

4. What Is — and Is Not — Retained

Retained after a scan	Never retained
Scan metadata: repository name, branch, commit SHA, timestamps, tools run	The cloned source tree
Findings: rule, severity, file path, line numbers, triage state	Uploaded ZIP archives
Code excerpts: up to ±6 lines around each finding (≈13 lines max per finding)	Build artifacts or binaries
SBOM component inventories and license analysis	Repository history beyond the scanned commit

Retained after a scan	Never retained
Generated reports and triage/audit history	Any code from files larger than 2 MB or binary files (excluded from excerpt capture)

The code excerpts exist so that engineers can review a finding in context without re-opening the repository. Excerpt capture is bounded: at most a few lines around the finding, individual lines truncated at 500 characters, and binary or oversized files skipped entirely. These excerpts are stored with the finding records under the retention and deletion controls described in Section 7.

5. Repository Access Model

Read-only access is sufficient for all scanning functionality. The platform supports two GitHub integration mechanisms (plus Bitbucket token support):

Capability	Integration	Permission required	Access level
Repository listing, cloning, scanning	Personal access token	repo scope (or fine-grained: contents & metadata read)	Read-only
Repository scanning via GitHub App	GitHub App	contents: read, metadata: read	Read-only
Optional: PR check runs & inline annotations	GitHub App	checks: write	Write (opt-in)
Optional: PR summary comments	GitHub App	pull_requests: write	Write (opt-in)
Optional: automated fix pull requests	Personal access token	repo push access; pushes only to new offload/fix/* branches — never the base branch	Write (opt-in)
Optional: CI release-gate commit status	GitHub Action	statuses: write — uses the workflow's own GITHUB_TOKEN, not a stored credential	Write (opt-in)

Customers who need scanning only can grant read-only credentials and every scanning feature will work. Write-level permissions are needed solely for the optional feedback features listed above, each of which is enabled explicitly by the customer.

6. Credential and Secrets Protection

- **Encryption at rest.** All stored integration credentials — git access tokens, GitHub App private keys and webhook secrets, cloud provider credentials, and AI provider API keys — are encrypted with

Fernet authenticated symmetric encryption (AES-128-CBC with HMAC-SHA256 integrity protection) before being written to the database. Key rotation is supported.

- **Decryption on use only.** Credentials are decrypted in memory at the moment of use and are never returned by configuration APIs; administrative views expose only whether a credential is set.
- **Log redaction.** Access tokens are masked in clone URLs, log lines, and stored error messages.
- **Webhook authenticity.** Inbound GitHub webhooks are verified with HMAC-SHA256 signatures and rejected (fail-closed) when no webhook secret is configured.
- **Transport security.** All communication with git providers and external APIs uses TLS.

Storage-level encryption of the database and disks (encryption at rest for scan results and metadata) is provided by the hosting environment; in on-premises deployments this is under the customer's direct control.

7. Tenant Isolation, Retention, and Deletion

7.1 Multi-tenant isolation

Every scan document, finding, report, and credential is scoped to the owning tenant (team). All read and write paths filter strictly on the tenant identifier, and per-tenant uniqueness constraints are enforced at the index level. One tenant's scan data — including code excerpts — is never visible to another tenant.

7.2 Retention windows

Scan results are subject to configurable retention windows enforced by scheduled retention jobs and TTL policies. Defaults (configurable per deployment):

Data category	Default retention
Code scan results, including findings and code excerpts	180 days
SBOM / dependency analysis reports	180 days
Cloud & container scan results	90 days
Generated reports	180 days
Platform audit logs	365 days

Retention for code-scan data is enforced unconditionally — records past the window are removed regardless of scan state, so the retention bound on stored code excerpts holds even for interrupted scans.

7.3 Deletion on request

- **Per-scan deletion.** Any scan report or SBOM report can be permanently hard-deleted through the product or API. Deletion removes the entire scan record, including all findings and embedded code excerpts.
- **Integration removal.** Git connections (and their encrypted tokens) can be removed at any time.
- **Tenant offboarding.** Deleting a tenant runs a platform-wide cascade that hard-deletes all of the tenant's scan data — code scan results, code excerpts, SBOM reports, triage history, analysis records, and stored git credentials — alongside its cloud, container, and Kubernetes security data.

Customers may request complete deletion of their data at any time; the mechanisms above make that deletion permanent rather than a soft-hide.

8. Scanning Toolchain and Intelligence Sources

For transparency, the scanners executed against customer code and the vulnerability intelligence sources feeding results are listed below. All tools run locally within the scan workspace.

Function	Tooling
Static analysis (SAST)	OpenGrep, Bandit; SonarQube (optional)
Dependency analysis (SCA)	osv-scanner, with malicious-package and typosquat detection and import-level reachability analysis
Secrets detection	Gitleaks
Infrastructure-as-code	Checkov
SBOM generation	Syft — CycloneDX JSON, SPDX JSON, syft-json; ingestion of CycloneDX (JSON/XML) and SPDX (JSON)
Container image scanning	Trivy, Grype

Intelligence source	Use
OSV.dev	Primary open-source vulnerability advisories
NVD and distribution advisories (via Trivy/Grype databases)	CVE data and CVSS scoring; GitHub Security Advisories arrive through these aggregations
OpenSSF malicious-packages dataset	Known-malicious dependency detection
CISA Known Exploited Vulnerabilities (KEV)	Active-exploitation flagging
FIRST.org EPSS	Exploit-probability scoring for prioritization

9. AI Features and Customer Code

The scanning pipeline itself involves no AI services: no code is sent to any model provider as part of a scan. Optional AI-assisted features (for example, suggested fixes for a finding, or natural-language scan summaries) operate under the following controls:

- **Explicit invocation.** AI features run only when a user invokes them on a specific finding or report.
- **Customer-controlled providers.** Requests are sent directly over TLS to the model provider (Anthropic, OpenAI, or Google) configured with the customer's own API key. These providers do not train on API traffic under their standard API terms.
- **Minimal payload.** Only the finding and its small code excerpt are included — never the repository.
- **No training, ever.** Offload does not use customer code to train or fine-tune any model.
- **No cross-tenant caching.** The platform's AI response cache is restricted, by design, to generic tenant-agnostic content; responses containing customer-specific context are never cached or reused across tenants.

10. Compliance and Certifications

[TO BE COMPLETED BY OFFLOAD SECURITY — state current certification status, e.g. SOC 2 Type II report availability under NDA, ISO 27001 scope, GDPR/DPDP posture and DPA availability, penetration-testing cadence, and responsible-disclosure policy.]

11. Summary

Customer question	Answer
Is our source code stored on the platform?	No. Code is cloned to an ephemeral scan workspace and deleted when the scan finishes, including on failure. Only findings and small per-finding code excerpts are retained.
Is stored data encrypted?	Credentials are application-layer encrypted (Fernet/AES); scan data is protected by storage-level encryption of the hosting environment; all transport uses TLS.
Is read-only repository access enough?	Yes, for all scanning. Write permissions are only for optional, opt-in feedback features.
Is our code used for AI training or shared with third parties?	Never. Optional AI features use the customer's own provider keys, and providers do not train on API traffic.
Can our data be deleted?	Yes — per-scan hard deletes, integration removal, automatic retention windows, and a full tenant-offboarding cascade.

For questions about this document, or to request additional security documentation, contact security@offloadsecurity.com.

OFFLOAD SECURITY

Code Security

Module Whitepaper

SAST, dependencies, secrets, IaC, SBOM, and license compliance — governed from commit to release

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Code Security module scans repositories and uploaded code for vulnerabilities across every layer — static code analysis, open-source dependencies, hardcoded secrets, infrastructure-as-code, and software supply chain — and turns the results into governed, auditable release decisions with fix automation and full triage lifecycle.

2. Key Capabilities

- Static analysis (SAST) via OpenGrep and Bandit, with optional SonarQube integration; findings carry code context so reviewers see the vulnerable lines in place.
- Dependency scanning (SCA) backed by OSV.dev and Grype, enriched with CISA KEV known-exploited flags and EPSS exploit-probability scores on every finding.
- Software supply-chain protection: known-malicious package detection (OpenSSF MAL advisories) and typosquat detection, both able to block the release gate.
- Import-level reachability analysis that classifies dependency findings as reachable or not-imported to cut noise (advisory — it never weakens the gate).
- Secrets detection with Gitleaks (~150 curated rules plus entropy analysis), fingerprinted and triageable.
- Infrastructure-as-code scanning with Checkov: Terraform, CloudFormation, Kubernetes manifests, Dockerfiles, and Helm.
- SBOM generation (Syft, CycloneDX) and upload analysis (CycloneDX JSON/XML, SPDX), with six-family license classification, policy engine, and generated NOTICE/attribution files.
- Deterministic, versioned release gates enforceable as required GitHub commit statuses; PR summary comments and check runs via a GitHub App.
- Automated fix pull requests: patches pushed to a dedicated fix branch — never the base branch — with a PR opened for review.
- Governed triage lifecycle: risk acceptance requires owner, justification, and expiry (expired acceptances re-block); false positives require reviewer and evidence; stable fingerprints persist decisions across re-scans.
- AI-assisted analysis and fix suggestions on individual findings, using the customer's own model-provider API key.
- GitHub and Bitbucket integration with encrypted tokens; ZIP upload for air-gapped code; CI/CD endpoints with scoped API keys.

3. How It Works

Repositories are shallow-cloned into an isolated, scan-scoped workspace with hardened git settings; scanners run locally and the workspace is deleted when the scan completes, on success or failure. Only

findings and small per-finding code excerpts are retained — the full source tree is never stored. (The companion whitepaper, 'Source Code Protection', covers this lifecycle in depth.)

Findings from every scanner are normalized, fingerprinted, and de-duplicated, then flow through the governed triage lifecycle. Release-gate evaluation is deterministic and policy-versioned: the same scan against the same policy always yields the same pass/review/fail decision with the blocking rule identifiers recorded — an audit-defensible answer to 'why did this ship?'.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Code findings share the platform's unified vulnerability queue, mint license and supply-chain violations into the enterprise risk register, provide evidence for compliance controls, and appear alongside cloud, container, and Kubernetes findings in a single per-repository report — one risk view from source code to runtime.

OFFLOAD SECURITY

Cloud Security Posture Management

Module Whitepaper

Continuous multi-cloud posture across AWS, GCP, and Azure

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Cloud Security module provides continuous security posture management across AWS, Google Cloud, and Microsoft Azure: misconfigurations, public exposure, compliance benchmarks, and cloud-provider threat signals, collected on demand and on schedule, and normalized into the platform's unified risk workflow.

2. Key Capabilities

- Multi-account, multi-cloud coverage: AWS, GCP, and Azure accounts onboarded with read-only credentials.
- Native provider signals: AWS Security Hub, AWS Config compliance, GuardDuty threats; GCP Security Command Center findings; Azure Defender for Cloud assessments and alerts.
- Open policy engines (Prowler-class checks) layered on top of native signals for provider-independent benchmark coverage.
- Region-aware scanning for AWS with parallel fan-out; global scanning for GCP and Azure.
- GCP organization onboarding with recursive project discovery.
- Scheduled scans through the unified scheduler, with staggered execution windows.
- Asset inventory correlation: discovered cloud assets tracked with lifecycle status and deletion reconciliation.
- Compliance scoring per account mapped to framework controls.

3. How It Works

Scans are orchestrated through a queued execution model: a scan request creates a tracked scan run, fans out per region and per account under concurrency caps, and dispatches to provider-specific scanners running on worker infrastructure. Credentials are decrypted only inside the worker process for the duration of the scan.

Findings from every provider are normalized into a common schema — severity, resource, region, rule, and compliance mapping — and stored as unified occurrences, so a GuardDuty threat and an Azure Defender assessment are managed through the same lifecycle as every other finding on the platform.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant

Aspect	How it is handled
	identifier, enforced by database indexes.
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Cloud findings feed the unified vulnerability management queue, mint into the enterprise risk register, contribute to compliance posture per framework control, appear in attack-path analysis, and raise centralized alerts — one data model shared with code, container, and Kubernetes findings.

OFFLOAD SECURITY

Kubernetes Security

Module Whitepaper

Continuous cluster posture: configuration, workloads, benchmarks, and images

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Kubernetes Security module provides continuous assessment of Kubernetes clusters — CIS benchmarks, workload best practices, NSA/MITRE framework checks, and image vulnerabilities — from a validated, encrypted cluster connection, with results unified into platform-wide risk and compliance views.

2. Key Capabilities

- Cluster onboarding with kubeconfig safety validation and encrypted credential storage.
- CIS Kubernetes Benchmark assessment (kube-bench).
- Workload configuration best practices (Polaris): privileged containers, resource limits, probes, security contexts.
- Framework-based posture scanning (Kubescape): NSA-CISA, CIS, and MITRE ATT&CK aligned controls.
- Container image CVE scanning for images running in cluster pods (Trivy).
- Findings tagged with MITRE ATT&CK techniques for threat-informed prioritization.
- Automatic first scan on cluster onboarding and scheduled recurring scans.
- Per-team cluster ownership enforcement on every scan and read path.

3. How It Works

Clusters are onboarded with a kubeconfig that is validated for safety, encrypted at rest, and scoped to the owning team. Scans run as background jobs: the platform connects to the cluster, executes each scanner in an isolated container, parses results into a common findings schema, tags MITRE techniques, and persists per-cluster scan history.

Scan execution is instrumented end-to-end (tracing and metrics), with orphaned-scan cleanup and a reconciler so cluster scan status always reflects reality.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.

Aspect	How it is handled
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Kubernetes findings flow into unified vulnerability management with the same triage lifecycle as cloud and code findings, mint into the risk register, contribute to compliance posture, and are included in scheduled reports and central alerting.

OFFLOAD SECURITY

Container & Software Supply Chain Security

Module Whitepaper

Registry-to-runtime image assurance: vulnerabilities, SBOMs, signatures, and secrets

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Container Security module scans container images across private and public registries for vulnerabilities, generates SBOMs, verifies signatures, lints Dockerfiles, and detects embedded secrets — connecting image risk to the source repositories and clusters where those images are built and deployed.

2. Key Capabilities

- Registry coverage: AWS ECR, Google Artifact Registry, Azure ACR, Docker Hub, and GitHub Container Registry, with saved-registry management and image discovery.
- Dual-engine vulnerability scanning (Trivy and Grype) with CISA KEV exploited-vulnerability enrichment.
- SBOM generation (Syft) in CycloneDX, SPDX, and syft-json formats.
- Dockerfile linting (Hadolint) and image signature verification (Cosign).
- Embedded-secret detection inside image layers (TruffleHog).
- Scheduled registry re-validation so image findings stay current.
- CI/CD integration: image and artifact scanning from pipelines via a GitHub Action and API-key-scoped endpoints.
- Registry authentication failures classified and surfaced honestly (auth vs. infrastructure vs. scan errors).

3. How It Works

Every scan runs in an isolated, single-use scanner container — the platform never installs scanning tools into its own runtime. Registry credentials are encrypted at rest, decrypted only in the backend process, and passed to scanners through a read-only authentication volume.

A single registry-authentication module owns all registry login flows, so ECR, Artifact Registry, and ACR authentication behave identically across every scan path. Results are normalized per image with severity summaries and stored with scan history for trending.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.

Aspect	How it is handled
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Image findings join unified vulnerability management, link to the repositories that build them and the clusters that run them, contribute to the risk register and compliance posture, and can gate CI/CD pipelines through the release-gate integration.

OFFLOAD SECURITY

Unified Vulnerability Management

Module Whitepaper

One queue, one lifecycle, one history — across cloud, code, containers, and Kubernetes

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

Unified Vulnerability Management is the platform's center of gravity: findings from every scanning module and integrated tool are normalized into a single occurrence model with deduplication, exploit-aware prioritization, governed lifecycle, SLA tracking, and full history — replacing per-tool consoles and spreadsheets.

2. Key Capabilities

- Single occurrence model across cloud, code, SCA, containers, Kubernetes, web/API, and integrated third-party sources.
- Normalization to canonical severities and stable fingerprints; duplicates collapse across rescans.
- Exploit-aware prioritization: CVSS enriched with CISA KEV (known-exploited) and EPSS (exploit probability).
- Governed lifecycle: open → triaged → resolved / risk-accepted / false-positive; risk acceptance requires an owner, justification, and expiry — expired acceptances automatically re-open.
- Ownership, SLA tracking, and vulnerability aging measurement.
- Full historical record: every occurrence retains its status history across scans.
- Resource-lifecycle awareness: occurrences auto-close when their resource is deleted, through a shared deletion cascade.
- Dashboards and executive reporting over live data — counts, trends, aging, and SLA posture.

3. How It Works

Every scanning module syncs its results through a single synchronization service that owns the canonical vocabulary — severities, statuses, and source types are defined once and imported everywhere. Fingerprinting keeps a finding's identity stable across scans, so triage decisions persist and re-scans update rather than duplicate.

Grouped-finding reads use a two-phase query pattern engineered for scale, keeping dashboards responsive as occurrence volumes grow into the millions.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.

Aspect	How it is handled
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Occurrences mint into the enterprise risk register with business context, drive compliance posture where controls require vulnerability management evidence, power central alerting, and are the substrate for attack-path and threat-intelligence correlation.

OFFLOAD SECURITY

Compliance & Risk Management (GRC)

Module Whitepaper

Compliance that reflects technical reality — controls, risks, evidence, and BIA connected to live findings

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The GRC module unifies compliance posture, enterprise risk management, and business impact analysis — and, unlike standalone GRC tools, wires them to the platform's live technical findings so dashboards, risk scores, and audit evidence reflect the actual state of the environment.

2. Key Capabilities

- Framework assessments and control posture: ISO 27001, SOC 2, SCF, and additional frameworks, with per-team control state.
- Technical findings mapped to compliance controls — a failed control is backed by the findings that fail it.
- Enterprise risk register with composite scoring: CVSS, EPSS, KEV, and business context.
- Cross-module correlators: attack-path toxic combinations, failed compliance controls, and BIA criticality all feed risk scoring.
- Treatment plans with SLA escalation; risks auto-close when their underlying findings resolve.
- Business Impact Analysis: RTO/RPO and financial impact per business process, correlated into risk prioritization.
- Security-questionnaire automation: AI-assisted answering of customer and vendor questionnaires from an approved knowledge base with evidence linkage.
- Automated audit, customer, and management reporting.

3. How It Works

A dedicated integration layer connects the GRC modules: correlators bridge compliance posture, attack-path analysis, and BIA into risk scoring without duplicating any module's data. Failed controls mint risks; resolved findings close them; every transition is observable and audited.

Assessments combine automated control evaluation (where technical evidence exists on-platform) with guided manual attestation, producing framework-mapped posture with a defensible evidence trail.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted

Aspect	How it is handled
	per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

GRC consumes every other module's findings and produces the executive layer above them: compliance dashboards, the risk register, treatment workflows, and generated reports — one system where the scan result and the audit answer trace to the same record.

OFFLOAD SECURITY

DPDP Act Compliance (India)

Module Whitepaper

A Data Fiduciary's control panel for the DPDP Act, 2023 and CERT-In Directions

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The DPDP module operationalizes India's Digital Personal Data Protection Act, 2023 and the CERT-In Directions 2022 for Data Fiduciaries: breach notification with statutory deadlines, DPIA lifecycle, Significant Data Fiduciary classification, vendor due diligence, and audit-ready evidence packs.

2. Key Capabilities

- Breach notification lifecycle per Rule 7: detect → declare → assess reportability → report → close, with deadline tracking for the Data Protection Board (72 hours) and CERT-In (6 hours), including watchdog alarms as deadlines approach.
- Rule 7(2) nine-section breach report validation and rendering; CERT-In Annexure I coverage across 18 incident categories.
- DPIA (Data Protection Impact Assessment) lifecycle: draft → review → approve / reject / expire, satisfying Section 13 obligations.
- Significant Data Fiduciary (SDF) classification scoring with the additional Section 10 obligations surfaced.
- Vendor/processor due-diligence assessments (18-question) with PASSING / CONDITIONAL PASS / FAILING outcomes.
- SCF-mapped readiness scoring with a Section 8(6) penalty-exposure estimate.
- Immutable audit-event log and generated audit packs for regulator or board review.
- Transfer impact analysis for cross-border data flows.

3. How It Works

The module is built as a pure domain core — state machines, validators, and scoring with no I/O — wrapped by orchestration services that persist state, emit audit events, and integrate with the platform risk register. This layering makes every statutory workflow deterministic and testable.

All DPDP collections are strictly team-scoped, and every material action lands in an immutable audit-event log from which audit packs are generated.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.

Aspect	How it is handled
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

DPDP readiness and breach posture feed the enterprise risk register and compliance dashboards; breach workflows can raise central platform alerts; and evidence generated here joins the same reporting layer as the rest of the platform.

OFFLOAD SECURITY

Threat Intelligence

Module Whitepaper

Real feeds, IOC correlation, and MITRE ATT&CK context on your own findings

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Threat Intelligence module ingests indicators from nine external providers, manages IOCs with confidence and aging, models actors and campaigns in STIX 2.1, and — most importantly — correlates threat context onto the platform's own findings to sharpen prioritization.

2. Key Capabilities

- Nine live feed providers: CISA KEV, AlienVault OTX, Abuse.ch, Feodo Tracker, SSL Blacklist, PhishTank, Blocklist.de, Spamhaus DROP, and OpenPhish — real ingestion and parsing, per-team feed management.
- Per-team encrypted API keys for authenticated feeds.
- IOC management: indicators with confidence, severity, and aging; IOC-to-finding correlation.
- Vulnerability prioritization with threat context: known-exploited and actively-abused signals applied to the platform's own vulnerability queue.
- Threat actors and campaigns modeled in STIX 2.1 with kill-chain, TLP, and MITRE ATT&CK mapping.
- MITRE ATT&CK heatmap built from ingested actor techniques.
- Alerting rules, threat-hunting queries, and landscape reports.

3. How It Works

Feeds are pulled through a provider factory with real HTTP ingestion and per-provider parsers; failures are classified (auth, rate-limit, format) rather than silently swallowed. Indicators are stored team-scoped with confidence scoring and aging so stale intelligence decays.

Correlation runs against the platform's unified findings: an IOC or KEV match on one of your vulnerabilities raises its priority in the same queue your team already works from — threat intelligence as an input to operations, not a separate console.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no

Aspect	How it is handled
	security data leaves your environment.

5. One Platform, One Risk View

Threat context enriches unified vulnerability management and the risk register, powers the MITRE heatmap, and can trigger central alerts; feed health is monitored like any other platform job.

OFFLOAD SECURITY

Integrations & SIEM Connectivity

Module Whitepaper

Your existing tools, one operational layer — including Wazuh SIEM

Version 1.0 · July 2026 · Classification: Customer-Shareable

1. Overview

The Integrations module connects the platform to the tools an organization already runs — SIEM, ticketing, chat, and security scanners — with team-scoped, encrypted credential storage, verified connections, and real data ingestion, so existing investments feed the unified risk view instead of another silo.

2. Key Capabilities

- Integration marketplace: a catalog of connectable tools with connect, test-connection, sync, and disconnect flows.
- Team-scoped credential store — integrations are team resources, usable by any teammate, isolated across tenants, encrypted at rest.
- Wazuh SIEM integration: agent inventory, alert ingestion, and posture summaries bridged into platform dashboards.
- Connection tests that verify real reachability and authentication at connect time — not fire-and-forget configuration.
- Scheduled global sync keeps integration data current; per-integration data summaries are stored and surfaced on the integration cards.
- Central alerting: integration failures and high-value events raise alerts in the platform's unified alert center.
- Audit and compliance connectors so integrated-tool evidence supports control posture.

3. How It Works

Each integration stores its configuration with sensitive fields encrypted at rest. Connect and sync flows execute a real test against the target system and record classified results. Ingested data lands in a common integration-data store with per-tool adapters, giving every integration a consistent freshness and health model.

The Wazuh bridge treats the SIEM as a first-class data source: agents and alerts are ingested, summarized, and correlated into the platform's posture views rather than merely linked out.

4. Data Handling & Security

Aspect	How it is handled
Credential protection	All connection credentials are encrypted at rest with Fernet authenticated encryption (AES-128-CBC + HMAC), decrypted in memory only at the moment of use, and never returned by APIs.
Tenant isolation	Every record is scoped to the owning team; all reads and writes filter on the tenant identifier, enforced by database indexes.

Aspect	How it is handled
Retention & deletion	Results are subject to configurable retention windows; data can be hard-deleted per scan and is removed by the tenant-offboarding cascade.
Deployment	Available as SaaS or fully self-hosted / on-premises — in the on-premises model no security data leaves your environment.

5. One Platform, One Risk View

Integrated-tool findings and events join unified vulnerability management and central alerting, extend compliance evidence, and appear in executive reporting — the consolidation layer that makes existing tools more valuable rather than replacing them.