Vulnara

SECURITY WHITEPAPER

Security and Data Protection at Vulnara

A technical description of the controls that protect your source code, credentials and security findings: identity, isolation, cryptography, scanning, infrastructure and operations.

PUBLIC | VERSION 1.1 | NOVEMBER 2026

Document information

TITLE Security and Data Protection at Vulnara	VERSION 1.1, November 2026
CLASSIFICATION Public	SECURITY CONTACT security@vulnara.io
PUBLISHER The Origami Corporation	AUDIENCE Security engineers, architects, procurement and compliance teams

Describes the controls in place at the date above. Reviewed and reissued as the platform evolves; provided for information only and not part of any contract.

Contents

01	Executive summary	3
02	Scope and shared responsibility	4
03	Standards and frameworks	5
04	Security architecture	7
05	Identity and authentication	8
06	Authorisation and access control	10
07	Team isolation	11
08	Cryptography and credential protection	12
09	Protecting findings and sensitive output	13
10	Secure scanning pipeline	14
11	Web application scanning and outbound requests	15
12	Source code integrations	16
13	Cloud account scanning	17
14	Private network access	19
15	Application security	20
16	Infrastructure and network security	21
17	Resilience, availability and monitoring	22
18	Secure development lifecycle	23
19	Incident handling and responsible disclosure	24
20	Control matrix	25
21	Glossary	26

01 Executive summary

Vulnara scans source code, dependencies and their licenses, web applications, networks and cloud accounts so engineering teams can find and fix security issues in their delivery pipeline. To do that, customers entrust us with access to some of their most sensitive assets. We treat that trust as the foundation of the product.

Security is designed in independent layers. Identity is verified at a single front door using OAuth 2.0 and OpenID Connect. Every request is authorised against the caller's role and team on the server. Internal data services refuse callers that cannot prove they are part of the platform. Every query is scoped to the owning team, and every stored credential is encrypted with authenticated encryption before it reaches the database.

TLS 1.2+
Older protocols refused; forward-secret AEAD ciphers

AES-128
With HMAC-SHA256 for every stored credential

60 s
Lifetime of the single-use sign-in handover code

10 min
Lifetime of a federated cloud identity token

Key commitments

Your data stays yours
Every resource belongs to exactly one team and is visible only to that team's members, according to their role.

No secret leaves its vault
Credentials are encrypted at rest, decrypted only for the operation that needs them, and never returned in full.

Least privilege everywhere
People, automation, integrations and cloud connections receive only the access a task needs, for only as long as it needs it.

Standards, not inventions
We build on published standards from the IETF, NIST and OWASP rather than proprietary security schemes.

OUR COMMITMENT

Every control in this document is in place on the Vulnara platform today. Where we describe planned work, we label it explicitly as such.

02 Scope and shared responsibility

This document covers the Vulnara web application, its GraphQL API and command line interface, the scanning platform behind them, and the connector customers can install in their own networks.

Vulnara is responsible for

- Identity verification and sessions
- Authorisation and team isolation
- Encryption of stored credentials
- Masking of sensitive findings
- Security of the scanning pipeline
- Infrastructure and network security
- Backups, monitoring, availability
- Secure development of the product

Customers are responsible for

- Team membership and roles
- Strong, unique account passwords
- Which repositories they connect
- Least-privilege scopes on tokens
- Read-only roles for cloud accounts
- Revoking access when people leave
- Authority to scan their targets
- Acting on the reported findings

Figure 1. Shared responsibility model.

Guiding principles

Defence in depth
Independent layers, so one failure is caught by the next.

Zero trust inside
Network location alone never grants access to data (NIST SP 800-207).

Least privilege
Minimal permissions, minimal lifetimes, minimal exposure.

03 Standards and frameworks

Vulnara's controls are built on published, peer-reviewed standards. The table lists the standards we implement or align with and where each applies. Alignment describes our engineering practice; it is not a claim of third-party certification.

Identity and access

Standard	How Vulnara applies it	Section
RFC 6749 OAuth 2.0	Authorisation code grant for all interactive sign-in	Identity
RFC 7636 PKCE	S256 code challenge on every authorisation request	Identity
RFC 9700 OAuth 2.0 Security BCP	Exact redirect matching, state binding, no tokens in URLs	Identity
RFC 8252 OAuth for Native Apps	Loopback redirects for command line and desktop clients	Identity
OpenID Connect Core 1.0	Identity layer, ID tokens, RP-initiated sign-out	Identity
RFC 7519 / 7517 / 7518 JWT, JWK, JWA	RS256 access tokens verified against published keys	Identity, Cloud
NIST SP 800-63B	Password length, breached-password screening	Identity

ALIGNMENT, NOT CERTIFICATION

The standards on these pages describe how we engineer and review the platform. Vulnara does not claim third-party certification against them. Customers can request further detail through security@vulnara.io.

Cryptography, networking and secure engineering

Standard	How Vulnara applies it	Section
RFC 8446 / RFC 5246 TLS 1.3 / 1.2	Only TLS 1.2 and 1.3 accepted on public endpoints	Cryptography
RFC 2104 HMAC	Webhook signatures, signed tunnel and state tokens	Integrations
Fernet specification	AES-128-CBC with HMAC-SHA256 for credentials at rest	Cryptography
RFC 1918 / 6598 / 4193 / 3927	Private, shared, unique-local and link-local ranges refused for customer-supplied addresses	Scanning
NIST SP 800-207 Zero Trust	Service authentication inside the platform network	Isolation
NIST SP 800-190 Containers	Isolated scan workloads, pinned images, segmentation	Infrastructure
NIST SP 800-218 SSDF	Specification-first development, automated verification	Development
OWASP ASVS 4.0	Reference for authentication, access control and data protection reviews	Application
OWASP Top 10 and CWE	Threat categories mapped to controls (CWE-79, 89, 284, 918 and others)	Application
CycloneDX (ECMA-424)	Software bill of materials for core platform releases	Development
Semantic Versioning 2.0	Every release versioned and traceable to its changes	Development

04 Security architecture

Vulnara is organised into security zones. Each zone has a single purpose and reaches only the zones it needs, through authenticated and policy-controlled paths.

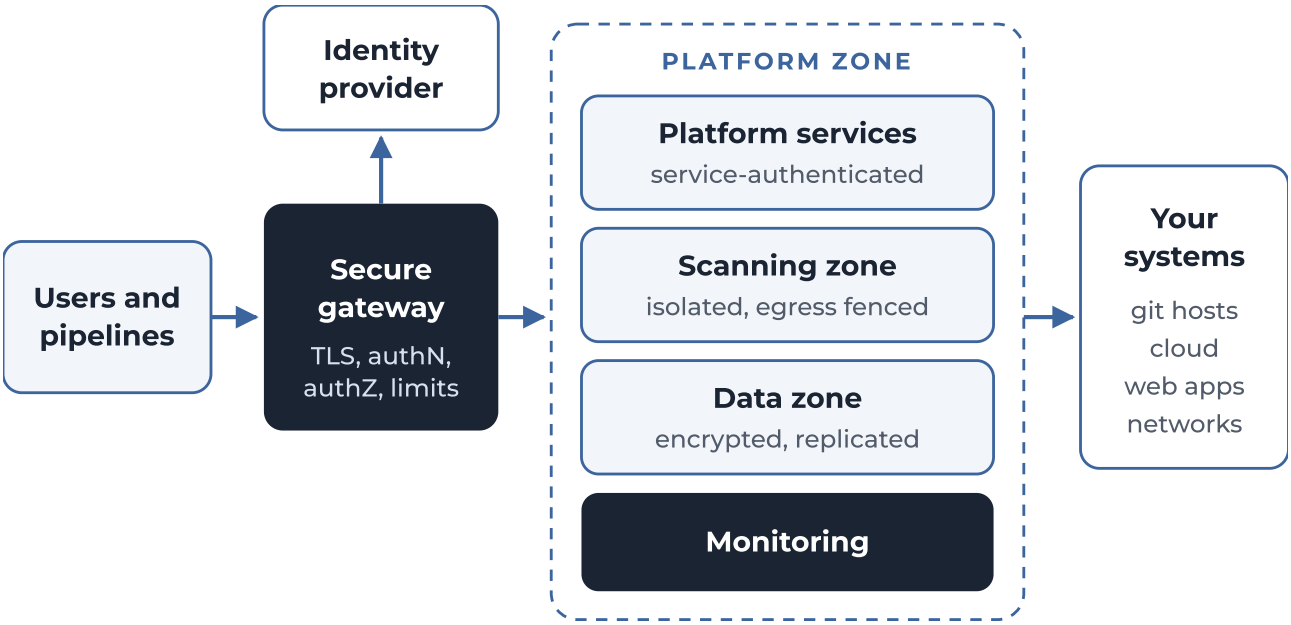


Figure 2. Security zones of the Vulnara platform.



Figure 3. Defence in depth: every request passes independent layers of control.

The gateway is the only path from the internet into the platform. Behind it, the services that hold customer data refuse any caller that cannot present platform service credentials, and network policy restricts which workloads may reach them at all. Scanning runs in its own zone, separated from customer data stores, with outbound connections fenced to the destinations a scan legitimately needs.

05 Identity and authentication

Every person and every automation is authenticated by a dedicated identity provider using OAuth 2.0 and OpenID Connect. The platform then verifies the resulting token on every request.

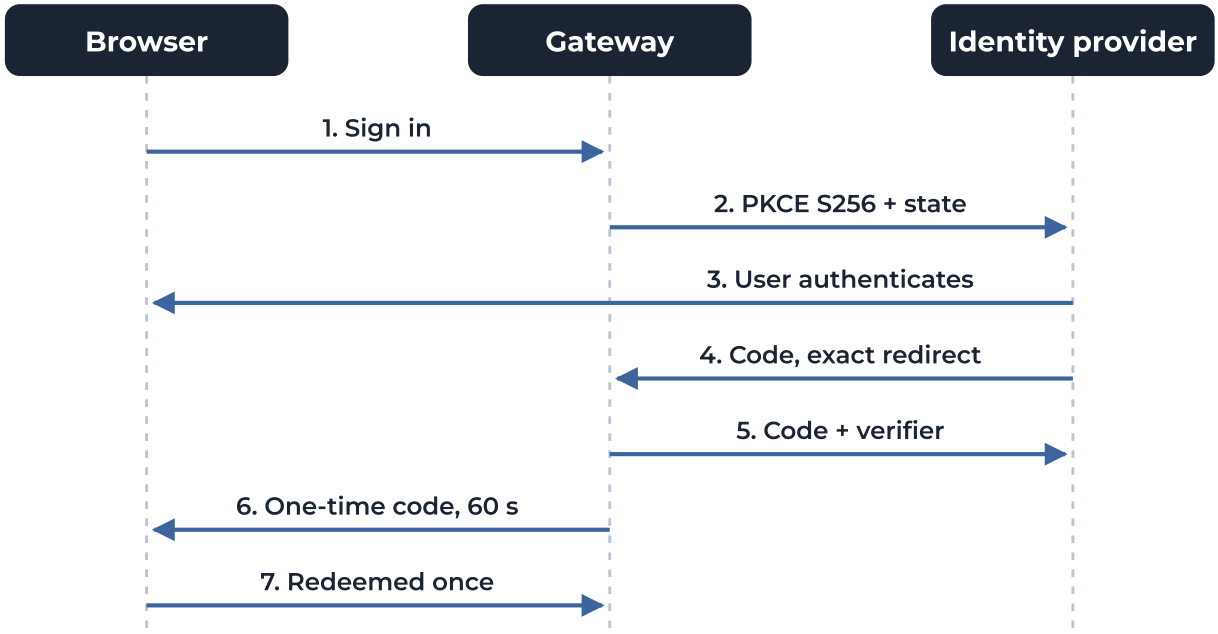


Figure 4. Authorisation code flow with PKCE (RFC 6749, RFC 7636).

Technical parameters

Parameter	Value
Grant type	Authorisation code with PKCE, method S256
Request binding	Random state and code verifier in HttpOnly cookies, 10-minute lifetime
Redirect URIs	Exact match against a registered list; loopback only for native clients
Browser handover	Single-use code, 60-second lifetime, stored only as a SHA-256 hash
Token format	JWT, RS256, verified against the provider's published JWKS
Token validation	Signature, algorithm pinned to RS256, issuer and audience allow-lists, expiry
Sign-out	RP-initiated logout at the identity provider; refresh token revoked

Password policy

Control	Implementation	Reference
Minimum length	12 characters	NIST SP 800-63B
Breached passwords	Every new password is screened against known breach corpora with a k-anonymity range query; only a 5-character hash prefix leaves the platform. Any match is refused.	NIST SP 800-63B
Where screening applies	Sign-up, invitation acceptance, password change and password reset	OWASP ASVS V2
Email changes	A new address must be confirmed before it is used	OWASP ASVS V2

Token handling in the browser

- Tokens never appear in a URL, browser history or server access log. The callback carries only the single-use handover code.
- The handover code is bound to the sign-in that created it and cannot be redeemed twice.
- Tokens issued by the identity provider for any other application are refused by audience validation.

Social sign-in

- Customers can sign in with GitHub, GitLab or Google through the identity provider, using each provider's OAuth or OpenID Connect flow.
- A social identity is matched to an account by the provider's stable user identifier, never by email address, so a provider account with a matching email cannot take over an existing account.

WHY IT MATTERS

These choices follow the OAuth 2.0 Security Best Current Practice (RFC 9700): PKCE for every client, exact redirect matching and no bearer tokens in front-channel URLs remove the most common ways OAuth deployments are attacked.

06 Authorisation and access control

Authentication establishes who is calling. Authorisation decides what they may do, and Vulnara makes that decision on the server for every operation.

Role model

Role	Purpose	Typical permissions
Viewer	Read access for stakeholders	View findings, scans, dashboards and reports
Editor	Day-to-day security work	Run scans, manage repositories, triage findings
Admin	Team administration	Members, roles, integrations, billing, reveal of protected values
Service account	CI pipelines and automation	Scoped to one team, with its own assigned role

Enforcement

- Roles are read from the identity provider's group membership, never from anything a browser or client sends.
- Every mutation declares the minimum role it requires and is checked on the server, so a lower role cannot perform a higher role's action by calling the API directly.
- Membership is per team: a person who belongs to several teams acts in exactly one team per request, with that team's role.
- CI pipelines authenticate as team service accounts, so automation can be granted exactly the role it needs and removed without affecting anyone's personal access.

Abuse protection

- API operations are rate-limited per client, with tighter limits on the most sensitive operations.
- Sign-in is additionally protected by the identity provider's own throttling and breached-password screening.

Staff access

- Vulnara staff administration runs through a separate internal console with its own sign-in and its own permission checks on every action.
- Platform administrator rights are granted to named staff only and managed as code, so access cannot be added outside a reviewed change.

07 Team isolation

Vulnara is multi-tenant. Keeping each team's code, findings and settings separate from every other team is the most important promise we make, and it is enforced in three independent layers.

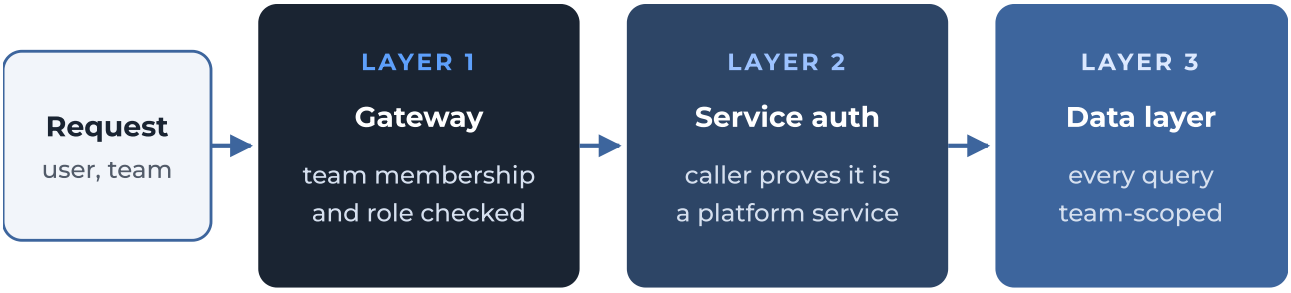


Figure 5. Three independent isolation layers. A request for another team's resource behaves as if it did not exist.

Layer	Implementation
1. Single entry point	All traffic enters through the gateway, which validates the token, confirms membership of the team the request acts in, and checks the role required. Data services have no public route.
2. Service authentication	Data services require a platform service credential on every request, compared in constant time and enforced application-wide, so new endpoints are covered by default. A missing or wrong credential receives HTTP 401. Identity-based network policy further restricts which workloads may connect.
3. Team-scoped data	Every read, update and delete passes through a shared data-access layer that adds the owning team to the query. Cross-team reads return "not found", never another team's row.
Verification	Automated cross-team tests attempt to read and modify other teams' data and run on every change.

08 Cryptography and credential protection

To scan on your behalf, Vulnara holds credentials for git providers, cloud accounts, web applications and package registries. Each is protected from the moment it is submitted.



Figure 6. Lifecycle of a credential on Vulnara.

Cryptographic inventory

Purpose	Mechanism
Data in transit	TLS 1.2 and 1.3 only; TLS 1.1 and earlier refused. ECDHE key exchange (forward secrecy) with AES-GCM.
Credentials at rest	Fernet: AES-128-CBC encryption with HMAC-SHA256 authentication, random IV per value, versioned ciphertext prefix
Key management	Keys held in dedicated secret management, outside the database. Multi-key rotation: the newest key encrypts, older keys only decrypt.
Fail-closed storage	For the most sensitive credentials, the platform refuses to write a value when no encryption key is configured.
User tokens	RS256 (RSA PKCS#1 v1.5 with SHA-256) signed JWTs
Cloud identity tokens	Asymmetrically signed JWTs published through an OpenID discovery document and JWKS
Webhooks, tunnel and state tokens	HMAC-SHA256, compared in constant time
Sign-in handover codes	Stored only as SHA-256 hashes

Credentials protected this way

Git provider access tokens, git application private keys, cloud account secrets, client and webhook secrets, web application sign-in details, package registry tokens and integration secrets. None are ever returned in full by the API.

09 Protecting findings and sensitive output

When Vulnara finds a secret or personal data in your code, the value itself is the most sensitive thing on the page. By default, nobody sees it in full.



Figure 7. How a sensitive value is handled on every surface.

- Sensitive values are masked in the application, the API and downloadable PDF reports, leaving only enough characters to recognise which value it is.
- Revealing a value in full is reserved for team admins, requires a deliberate action, and is written to the team's audit log.
- Alert notifications and webhook payloads carry the context of a finding, never the secret itself.
- Placeholder values, such as example keys in documentation, are recognised and set aside automatically, with the reason recorded.

Request bodies and errors

- Error reports never include the bodies of requests that carry credentials.
- Validation errors name the field that was refused, never the submitted value.

Audit trail

Changes to a team, its members, roles, integrations and settings are recorded in a team audit log that admins can review and export. Retention of read events is configurable per team. Every issuance of cloud or registry credentials to a running scan is also recorded.

10 Secure scanning pipeline

Scanners read untrusted input by design. The pipeline is built so that scanning can never become a path into your data, another team's data or the platform itself.

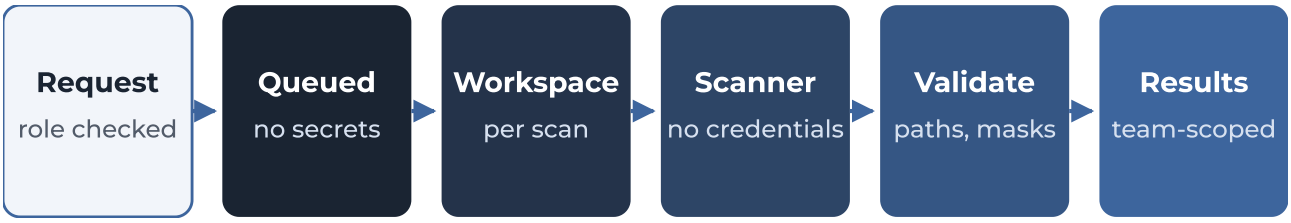


Figure 8. The path of a repository scan.

Control	Implementation
Queue messages	Carry identifiers, never git, registry or cloud credentials. Workers fetch credentials over an authenticated internal call at the moment of use.
Workspace	Each scan clones into its own short-lived working copy, removed when the scan ends.
Credential separation	The token used to fetch code, and repository metadata that could carry it, are kept out of the scanner's reach.
Scanner invocation	Platform-defined images and arguments, passed as argument lists, never through a shell. Repository content cannot change how a scanner runs.
Image pinning	Scanner images are pinned to exact versions or content digests.
Result validation	Reported locations are resolved against the scanned copy; a path outside the repository is rejected.
Scanner catalogue	Controlled by the platform; new scanners are disabled by default, and a team can run only the scanners its plan includes.
Deduplication	Findings are fingerprinted so the same issue reported across commits and tools is stored and counted once.

11 Web scanning and outbound requests

Whenever Vulnara connects to an address a customer supplies (a web application, a self-managed git server, a package registry or an alert webhook), it applies the same server-side request forgery (CWE-918) protections.

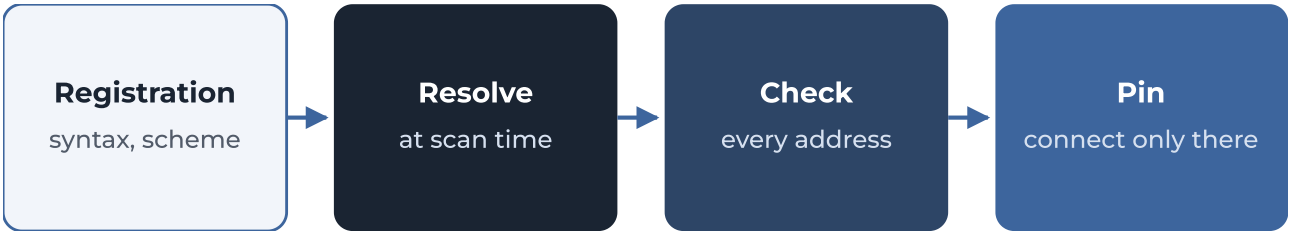


Figure 9. Resolve, check and pin: defeating DNS rebinding.

Control	Implementation
Refused ranges	Loopback, RFC 1918 private, RFC 6598 shared, RFC 3927 link-local (including cloud metadata), RFC 4193 unique-local and unspecified addresses
Encoding tricks	Integer and shorthand IPv4 forms, embedded credentials and internal name suffixes refused at registration
DNS rebinding	Names resolved once per scan; every address must be public; the connection is pinned to the checked address
Redirects	Never followed off the registered host; integration calls follow no redirects
Transport	HTTPS required for git servers, registries and alert webhooks
Alert destinations	Slack and Discord alerts are delivered only to Slack and Discord hosts

Web application (DAST) scans

- Only non-destructive checks run: nothing that modifies data, floods a service or brute-forces a login.
- Requests are rate-limited, stay within the registered host and scope, and identify themselves as Vulnara.
- Sign-in details are write-only, stored with fail-closed encryption, and released only to a scan while it is running. They are never placed on a message queue.

12 Source code integrations

Vulnara connects to GitHub, GitLab, Bitbucket, Azure DevOps and self-managed git servers. Each connection uses the least access possible, for the shortest time possible.



Figure 10. Least-privilege access through a git application installation.

Control	Implementation
Installation tokens	No long-lived repository token is held. Each step requests a fresh token scoped to one repository and the permissions it needs, such as read-only access to clone.
Customer-owned apps	Teams can register their own git application, so access is granted by an application they control. Its private key is verified before it is saved and never returned.
App-only policy	Teams can require the git application, after which personal access tokens cannot be added or used.
Installation linking	Requires an OAuth proof from the provider that the person linking it can access that installation; the identifier in the callback URL alone is never trusted.
Inbound webhooks	Verified with HMAC-SHA256 signatures or the provider's shared token, compared in constant time, before processing. Events from one application only affect that application's own installations.
Token renewal	OAuth connections are renewed under a per-token lock; single-use refresh tokens are never replayed.
Platform credentials	The platform's own credentials are never sent to a customer's self-managed git server.

13 Cloud account scanning

Vulnara reviews the configuration of AWS, Google Cloud, Microsoft Azure, Oracle Cloud Infrastructure, OpenStack, Proxmox VE and IBM Cloud with read-only access. Where a cloud supports it, access is keyless, through workload identity federation that you configure and control. Otherwise, or if you prefer, you provide a read-only secret, which is stored encrypted and exchanged for short-lived credentials at scan time.

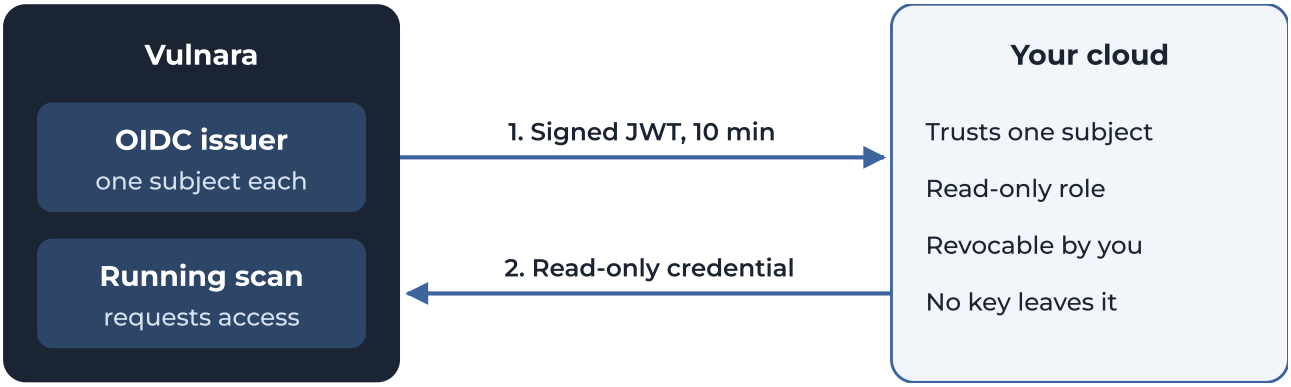


Figure 11. Keyless cloud access through workload identity federation.

Control	Implementation
Federation	Vulnara operates its own OpenID Connect issuer with a public discovery document and JWKS. Credentials are obtained with each provider's native federation: AWS STS AssumeRoleWithWebIdentity, Google Cloud workload identity federation, Microsoft Entra federated credentials.
Subject binding	Each connection has a unique subject that only Vulnara can mint, so no team can obtain a token for another team's trust relationship.
Token lifetime	Identity tokens are valid for 10 minutes and carry a unique identifier (jti).
Issuance	Credentials are issued only to a scan that is pending or running for that connection, over an authenticated internal call, and every issuance is audited with its jti.
Over-privilege check	Before each scan, the identity is probed for permission to read stored secrets without reading any. A role that can is refused.
Customer endpoints	OpenStack and Proxmox VE addresses must use HTTPS and may not carry embedded credentials. Direct connections refuse private addresses; private endpoints are reached only through a connector.
Engines	Scan engines are pinned by content digest and run without extra privileges. Output that resembles a secret value is discarded rather than stored.

Sign-in methods by cloud

Cloud	Keyless	With a stored secret
AWS	Federated role assumption	Access key, used only to assume the read-only role
Google Cloud	Workload identity federation	Service account key, used only to sign a short-lived token request
Microsoft Azure	Federated credential	Client secret of a read-only application
Oracle Cloud Infrastructure	Token exchange of a Vulnara identity token, authenticated with an identity domain client secret	
OpenStack	Not available	Application credential bound to one project
Proxmox VE	Not available	Privilege-separated API token
IBM Cloud	Not available	API key of a Service ID; personal user keys are refused

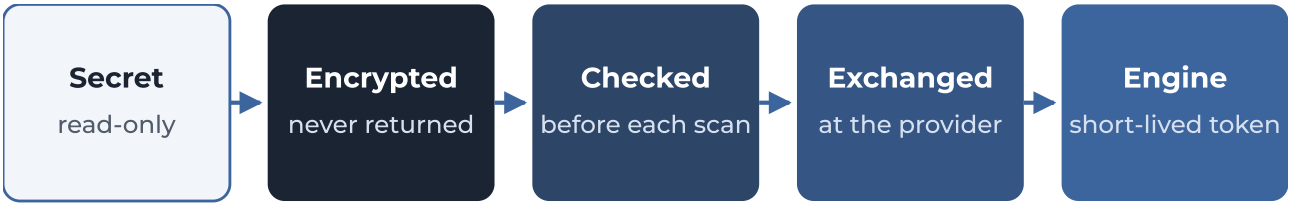


Figure 12. How a stored cloud secret becomes a short-lived scan credential.

Control	Implementation
Stored secrets	Encrypted before storage in fail-closed mode: with no encryption key configured, the secret is refused rather than stored. The API reports only whether a secret is set, never its value.
Short-lived scan credentials	The secret is exchanged with the provider at scan time, and the engine receives only the resulting short-lived token. Proxmox VE has no token exchange, so its privilege-separated token is passed to the engine for that scan only.
Over-privilege refusal	Before each scan, the connection's permissions are read and checked. Refused: Oracle policies that read secrets or use keys, unrestricted or other-project OpenStack credentials, IBM roles beyond Viewer and Reader or Reader on data services, and Proxmox privileges beyond auditing.
Self-signed certificates	A Proxmox VE connection can be pinned to the server certificate's SHA-256 fingerprint, which is checked during the handshake, before the token is sent.

14 Private network access

For systems not reachable from the internet, such as self-hosted git servers, internal registries and private cloud endpoints, customers run a lightweight connector inside their own network. Nothing is exposed inbound.

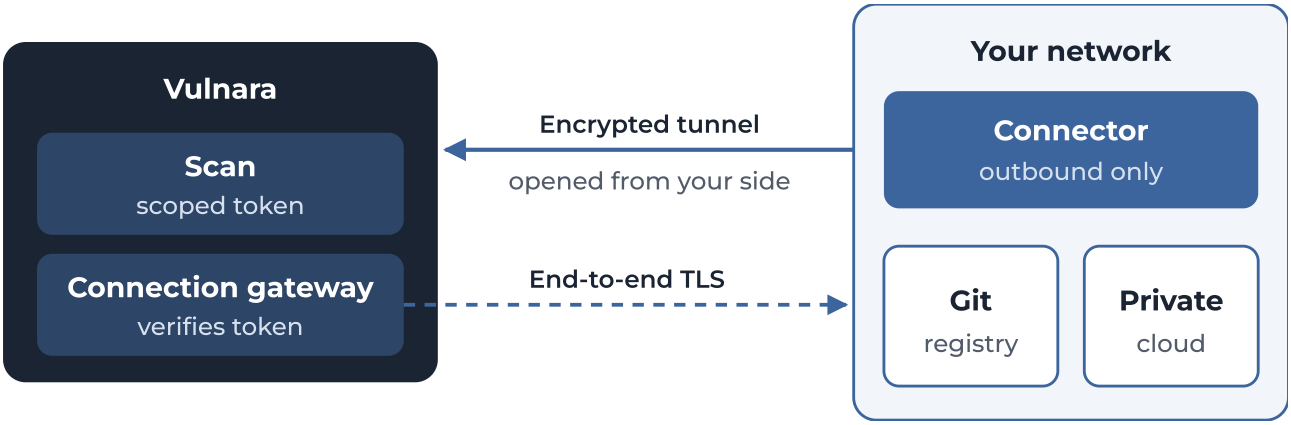


Figure 13. Reaching private systems without opening an inbound port.

Control	Implementation
Connectivity	The connector makes outbound connections only, over an encrypted tunnel. No inbound port or public service is required.
Tenant separation	Each customer's connectors are isolated from every other customer's.
Connection tokens	Every connection carries an HMAC-SHA256 signed token bound to the team, the connector and one destination host and port, valid for 10 minutes, verified before forwarding and never placed on a queue.
Customer allow-list	The connector can enforce its own local allow-list, so the customer decides the final set of reachable systems. Loopback and link-local destinations are always refused.
Cloud destinations	For a cloud connection, the connector forwards only that cloud's API domains, the customer's own OpenStack or Proxmox VE endpoint, or a host list the customer sets, at most 50 entries.
Encryption	Connections to git servers, registries and cloud APIs stay TLS-encrypted end to end between Vulnara and the customer's system, including custom CA trust per instance.
Availability	An offline connector is reported, and no connection is attempted through it until it returns.

15 Application security

The Vulnara web application and API are built to resist the common classes of web attack catalogued in the OWASP Top 10 and the CWE.

Threat	How Vulnara addresses it	CWE
Broken access control	Server-side role checks on every mutation, team scoping on every query, cross-team tests on every change	CWE-284, 639
Cryptographic failures	Authenticated encryption at rest, TLS 1.2+ in transit, keys held outside the database	CWE-311, 327
Injection	Parameterised queries through an ORM throughout; external tools invoked with argument lists, never a shell	CWE-89, 78
Cross-site scripting	User-supplied text rendered as text, never markup; a Content Security Policy restricts what the browser may load and run	CWE-79
Authentication failures	OAuth 2.0 with PKCE, breached-password screening, rate limiting, full token validation	CWE-287, 521
Integrity failures	Signed webhooks, locked dependencies, CI actions pinned to commit hashes	CWE-345, 494
Server-side request forgery	Public-only addresses, resolve-check-pin, HTTPS-only webhooks	CWE-918
Sensitive data exposure	Masking on every surface, audited reveal, credential-bearing requests excluded from error reports	CWE-200, 532
Vulnerable components	Lock files, automated dependency updates, static analysis and SBOMs	CWE-1104
Timing attacks	Secrets and signatures compared in constant time	CWE-208

SECURE BY DEFAULT

Service authentication is applied to every route of the data services, and team scoping lives in the shared data-access layer. A new endpoint is protected the moment it exists, not when someone remembers to add a check.

16 Infrastructure and network security

The infrastructure behind Vulnara runs on containers, is declared as code, segmented by default and managed with the same discipline we ask of our customers.

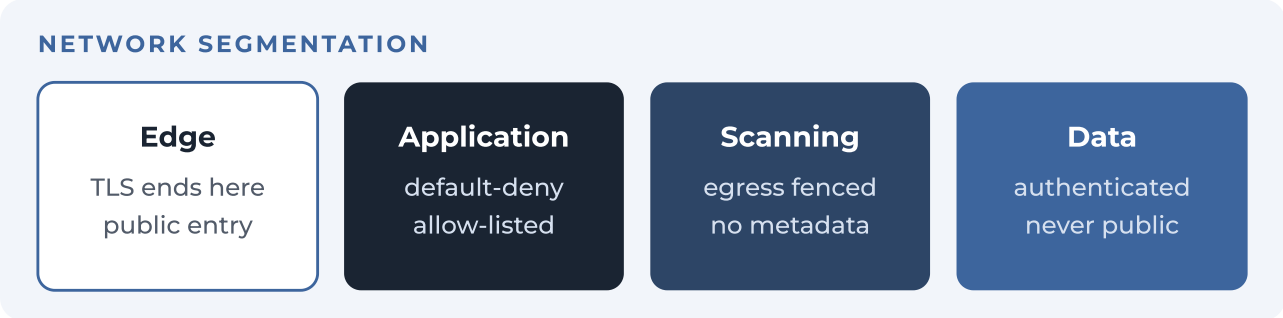


Figure 14. Network segmentation of the platform.

Control	Implementation
Ingress policy	Identity-based network policies enforced in the kernel: data services accept connections only from explicitly allow-listed workloads.
Egress policy	Web and cloud scanners, which reach the internet, are blocked from private ranges, internal services and cloud metadata endpoints.
Storage	Databases, queues and object storage expose authenticated interfaces only, and none is reachable from the internet.
Infrastructure as code	The platform is declared in version control and reconciled continuously (GitOps): what runs is what is committed, and drift is reverted.
Secret management	Secrets live in dedicated, encrypted secret management, or encrypted in version control with public-key encryption. Never in plain text in a repository.
Secret scanning	Every infrastructure change is scanned for accidentally committed secrets before merge.
Rotation	Workloads restart onto a rotated secret automatically.
Identity provider	Applications, flows, redirect URIs and administrator membership are managed as code; changes made outside code are reverted on the next apply.

17 Resilience, availability and monitoring

Customers rely on Vulnara in their delivery pipelines. The platform is built to keep running through failures and to recover data if it ever needs to.



Figure 15. Point-in-time recovery window.

Area	Implementation
Database topology	Three replicated instances on separate nodes with automatic failover
Backups	Continuous write-ahead log archiving plus daily full backups to object storage
Recovery point	Point-in-time recovery to any moment in the last 7 days
Backup monitoring	Backup freshness and archiving are monitored continuously, with alerts if either falls behind
Redundancy	The gateway and core API run as multiple replicas with disruption budgets, so node maintenance never takes them offline
Releases	Rolling updates: new versions must pass readiness checks before old ones are stopped
Monitoring	Error rates, latency, queue depth, scan health, database capacity and backup jobs, with alerts routed to engineering
Logging	Logs from all services collected centrally for investigation

18 Secure development lifecycle

We build Vulnara the way we help customers build their software: security checks in the pipeline, on every change, before anything ships. Our process aligns with the NIST Secure Software Development Framework (SP 800-218).

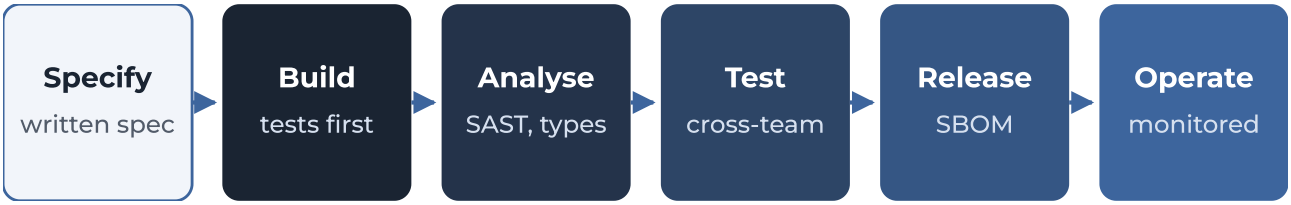


Figure 16. From specification to production.

Practice	Implementation	SSDF
Specification first	Behaviour, including security behaviour, is specified as requirements with scenarios; tests are traced to the scenario they prove	PW.1
Code review gates	Changes land through pull requests running lint, type checking and the automated test suite	PW.7
Static analysis	Security-focused static analysis gates every backend service	PW.8
Dependency locking	Exact versions and hashes in lock files; builds fail on a lock that does not match its manifest	PS.3
Automated updates	Dependency updates proposed automatically across repositories	RV.1
Build supply chain	Third-party CI actions pinned to full commit hashes	PO.3
SBOM	CycloneDX software bill of materials for core platform releases	PS.3
Release automation	Images built and deployed by CI, versioned with Semantic Versioning, never from a developer's machine	PO.5

CURRENT FOCUS

Planned work: cryptographic signing of every container image we ship, narrower per-service permissions at the identity provider, and encrypted off-site copies of database backups.

19 Incident handling and disclosure

We prepare for security events so we can respond quickly and transparently, and we welcome reports from researchers and customers.



Figure 17. Incident handling stages, following the NIST SP 800-61 life cycle.

- Monitoring and alerting surface anomalies to our engineering team for prompt assessment.
- Suspected incidents are investigated, contained and resolved, and each ends with a review of what to change so it does not recur.
- If an incident affects customer data, we notify affected customers without undue delay and meet our obligations under applicable data protection law.

Responsible disclosure

If you believe you have found a vulnerability in Vulnara, please report it privately to **security@vulnara.io**, or see vulnara.io/.well-known/security.txt. Do not open a public issue, and give us a reasonable window to resolve it before disclosure. We do not pursue researchers who act in good faith and within these guidelines.



Figure 18. What happens after you report a vulnerability.

20 Control matrix

A summary of key controls and the standards they align with.

Domain	Key controls	Reference
Authentication	Authorisation code + PKCE S256, exact redirects, 60 s single-use handover, RS256 with issuer and audience validation	RFC 6749, 7636, 9700; OIDC Core
Passwords	12+ characters; k-anonymity breached-password screening on every set or reset	NIST SP 800-63B
Authorisation	Server-side roles on every mutation; service accounts for automation	OWASP ASVS V4
Isolation	Single gateway, service authentication, team-scoped data layer, cross-team tests	NIST SP 800-207
Cryptography	TLS 1.2/1.3 with ECDHE and AES-GCM; AES-128-CBC + HMAC-SHA256 at rest; key rotation	RFC 8446; Fernet; ASVS V6
Sensitive data	Masking on every surface, audited reveal, credentials kept out of queues and error reports	ASVS V8; CWE-200
Outbound requests	Public-only addresses, resolve-check-pin, HTTPS-only, no off-host redirects	CWE-918; RFC 1918, 6598
Integrations	Repository-scoped short-lived tokens, OAuth-proven linking, HMAC-SHA256 webhooks	RFC 2104; ASVS V13
Cloud access	Workload identity federation or an encrypted read-only secret, short-lived scan credentials, over-privilege refusal, audited issuance	RFC 7519; OIDC Discovery
Private networks	Outbound-only connector, destination-bound signed tokens, customer allow-list	NIST SP 800-207
Infrastructure	GitOps, kernel-enforced segmentation, encrypted secret management, secret scanning	NIST SP 800-190
Resilience	3 database replicas, continuous archiving, 7-day point-in-time recovery, disruption budgets	Industry practice
Development	Specification-first, SAST, locked and pinned dependencies, CycloneDX SBOM	NIST SP 800-218
Incidents	Detect, assess, contain, resolve, learn; customer notification	NIST SP 800-61

21 Glossary

Term	Meaning
AEAD	Authenticated encryption with associated data, such as AES-GCM: confidentiality and integrity in one operation.
CWE	Common Weakness Enumeration, the industry catalogue of software weakness types.
DAST	Dynamic application security testing: testing a running web application from the outside.
DNS rebinding	An attack where a name resolves to a public address when checked and an internal one when used.
ECDHE	Ephemeral elliptic-curve Diffie-Hellman key exchange, which gives TLS forward secrecy.
GitOps	Operating infrastructure by reconciling it continuously against a version-controlled declaration.
HMAC	Hash-based message authentication code (RFC 2104), proving a message was produced by a key holder.
JWT / JWKS	JSON Web Token (RFC 7519) and the published set of keys used to verify it (RFC 7517).
k-anonymity query	A lookup that sends only a short hash prefix, so the full password hash never leaves the platform.
OIDC	OpenID Connect, an identity layer on top of OAuth 2.0.
PKCE	Proof Key for Code Exchange (RFC 7636), binding a sign-in to the client that started it.
SAST / SCA	Static analysis of source code / analysis of third-party dependencies.
SBOM	Software bill of materials: an inventory of the components in a release.
SSRF	Server-side request forgery (CWE-918): tricking a server into contacting an address of the attacker's choosing.
Workload identity federation	Granting cloud access through trusted, short-lived identity assertions instead of stored keys.

Questions about security?

Our team is happy to answer security questionnaires and to discuss how Vulnara fits your organisation's requirements.

security@vulnara.io

For vulnerability reports, please use the same address and see the responsible disclosure section of this document.

Vulnara

Intelligent Security for Modern Development.

vulnara.io | Built by The Origami Corporation

This document is provided for information only. It describes the platform at the date of publication and may be updated as the platform evolves.