C2MD Compliance Agent — Security & Privacy Overview

v1.0 · April 2026

1. Executive Summary

This document describes the security and privacy posture of the C2MD Compliance Agent, the production deployment operated by Value Driven AI at `https://c2md.getvda.ai`. It is intended for prospective buyers, security reviewers, compliance leads, and procurement teams evaluating C2MD against enterprise vendor risk frameworks (SIG, CAIQ, ISO 27001 Annex A controls, NIST CSF, and similar).

C2MD is a pre-launch, single-operator product with strong technical security controls and an honest acknowledgement of the documentary and procedural gaps that come with early-stage commercial software. Where formal certifications and processes are not yet in place, this document describes the compensating technical controls and the trigger-based commitments for closing each gap.

Scope of this document

This describes the security and privacy posture of the C2MD service as deployed. It does not describe the security of customer-built integrations or the security of the AI agents under assessment by C2MD — those remain the responsibility of the customer.

What is in scope: authentication and authorisation, data flows, infrastructure, application security controls, sub-processors, incident handling, and the regulatory posture of the deployed service.

What is explicitly out of scope: the security of the customer's tenant, the customer's identity provider, the customer's network, the AI agents whose compliance bundles are produced by C2MD, and any third-party services the customer chooses to integrate with.

Operating context

C2MD is operated by Value Driven AI, a single-operator commercial entity. The product is pre-launch and pre-revenue at the date of this document. The roadmap commitments expressed throughout are trigger-based — most are anchored to "within N days of first paying customer" — to reflect the genuine sequencing of remediation work rather than aspirational calendar dates that pre-revenue products cannot honestly commit to.

How to read the roadmap entries

Throughout this document, when we describe a control as roadmapped, we use the following phrasing:

Phrase	Meaning
Trigger: First paying customer	Implementation begins within 30 days of contract signature with first paying customer; targeted complete within 90 days.
Trigger: Five paying customers	Implementation begins when the customer base reaches five organisations; targeted complete within a further 90 days.
Trigger: Series funding event	Implementation begins when the operating entity raises external capital sufficient to expand the team.
Trigger: Customer request	Implementation begins when a specific paying customer requests the control as a condition of procurement.
Calendar date	A small number of items have specific date commitments (publication of `SECURITY.md`, scheduling of first penetration test). These are explicitly dated.

Document availability

This document is published in two formats: a human-readable PDF and a machine-parseable Markdown source. Both are available at `getvda.ai/security`. The Markdown form is intended for buyers' AI-assisted procurement workflows and security questionnaire automation.

How to engage on items requiring custom configuration

Several controls described in this document are available under what Google Cloud Marketplace calls a **Private Offer** — a per-customer contractual mechanism that allows specific configurations, pricing, or contractual terms to be negotiated outside the standard Marketplace listing.

Examples of items configurable via Private Offer (full list in Section 11.6):

- Customer-Managed Encryption Keys (CMEK) configuration
- Multi-region EU deployment for resilience beyond single-region Cloud Run
- Increased Cloud Run maximum instance count for high-volume customers
- HIPAA Business Associate Agreement and assessment
- ISO 42001 certification under contractual commitment
- Engagement of a managed-service partner for operational continuity

To initiate a Private Offer conversation:

1. Review the standard Marketplace listing at `console.cloud.google.com/marketplace` and identify the item(s) you require beyond the standard offering
2. Contact `mike@getvda.ai` with subject line beginning "PRIVATE OFFER" and a description of the required configuration
3. We will respond within 2 business days with either: (a) a draft Private Offer with proposed pricing and terms, or (b) a request for clarification on the technical or contractual requirements
4. Once the Private Offer is mutually agreed, Google Cloud Marketplace handles the execution and billing

Private Offers are how enterprise procurement teams typically engage with Marketplace vendors for non-standard configurations. The mechanism is well-understood by procurement teams familiar with Google Cloud Marketplace, AWS Marketplace, and Azure Marketplace (all of which support similar contractual flexibility).

For customers without an existing Google Cloud Marketplace relationship, direct contact at `mike@getvda.ai` is also welcome — the Private Offer mechanism becomes available once a Google Cloud billing account is identified.

---

2. Architecture & Trust Boundaries

C2MD is a stateless API service running on Google Cloud Run in the `europe-west1` region. The service accepts authenticated requests from agent-to-agent invocation or REST clients, processes the input through a multi-stage pipeline, and returns a signed compliance bundle in the response. No customer data is persisted to disk or to any database.

2.1 Trust boundary

The trust boundary of the C2MD service is the HTTP request/response cycle. Each request:

1. Is authenticated by Google Cloud Run's IAM perimeter (anonymous requests rejected at HTTP 403 by Google Frontend before reaching the container)
2. Has its Bearer token validated by the application (Google OAuth2 or Microsoft Entra ID JWKS)
3. Is screened by Model Armor for prompt injection, malicious URIs, and sensitive data exposure
4. Has personally identifiable information pseudonymised by Microsoft Presidio before any LLM invocation
5. Is processed by the agent's skill logic, which calls Vertex AI within the same region
6. Has its outputs screened a second time by Model Armor before being included in the response bundle
7. Returns the signed bundle to the caller; no record of the input or output is retained by the service

2.2 Sub-processors

C2MD uses exactly one sub-processor: Google Cloud Platform. All compute, model inference, content screening, secret storage, and audit logging operations are performed by Google Cloud services within the European Union. There are no third-party logging providers, monitoring services, analytics tools, or marketing pixels.

Sub-processor	Service	Purpose	Region
Google Cloud	Cloud Run	Container hosting	europe-west1
Google Cloud	Vertex AI	LLM inference	europe-west1
Google Cloud	Model Armor	Input/output screening	EU multi-region
Google Cloud	Secret Manager	Cryptographic key storage	europe-west1
Google Cloud	Artifact Registry	Container image storage	europe-west1
Google Cloud	Cloud Build	Build pipeline	europe-west1
Google Cloud	Cloud Logging	Audit log retention	europe-west1

Google Cloud's compliance certifications (SOC 1/2/3, ISO 27001/27017/27018, PCI DSS, HIPAA, FedRAMP) apply to the underlying infrastructure. The full list is at `cloud.google.com/security/compliance`.

2.3 No customer data persistence

C2MD does not persist customer-supplied data. Specifically:

- Agent descriptions submitted via `/a2a` are processed in-memory during the request and discarded when the response is returned
- LLM prompts and responses are not logged (verified in `src/llm/vertex_client.py`; only skill identifier, model name, token counts, and latency are structured-logged)
- Generated compliance bundle artifacts are returned to the caller in the response and not stored server-side
- The Presidio entity map (mapping pseudonyms back to original PII) is held only in the request's in-memory data structure and explicitly never logged or persisted (per code comment in `src/pii/presidio.py`)

2.4 Attack surface reduction by design

C2MD's architecture eliminates entire categories of common vulnerabilities by structural choice rather than by control. Where a traditional SaaS application invests significant effort in defending against these attack surfaces, C2MD removes the surface itself.

Common SaaS attack surface	C2MD posture
Session hijacking, fixation, replay	No sessions exist. Each request authenticates independently via Bearer token. No session cookies, no session storage, no session timeout management, no logout endpoint.
Password breach, credential stuffing, brute force	C2MD never handles passwords. Authentication is fully delegated to Google or Microsoft Entra. Password complexity, MFA enforcement, breach detection, and account lockout are the IdP's responsibility — and both Google and Microsoft do this at a scale and quality no individual SaaS can match.
SQL injection, ORM injection, NoSQL injection	No database. C2MD has no persistent customer data store. All persistence is via Google-managed APIs (Secret Manager, Cloud Logging) which are not user-input-driven.
Database exfiltration, backup theft	No customer database to exfiltrate. No customer-facing backups. The only secret in Secret Manager is the Card-signing keypair, which is itself a public-key/private-key construction where compromise is detectable via signature verification.
Account takeover via password reset, session theft	No accounts exist on C2MD. Identity is the IdP's account. There is no "forgot password" flow, no session token rotation, no account-recovery surface.
Privilege escalation through stored roles	No stored roles. Authorisation scopes are looked up per-request from a small in-code table; there is no admin UI, no role-management interface, no user-can-promote-self surface.
Cross-tenant data leakage	No tenant data is stored. Each request is processed and returned in isolation; there is no cross-request state that could leak between tenants.
Abandoned session, idle timeout	Not applicable; sessions don't exist. Token expiry is enforced by the IdP.
Internal admin panel compromise	No admin panel. There is no web UI for C2MD's operations; all administration is through Google Cloud's IAM-protected APIs.
CSRF on state-changing operations	C2MD's API is stateless and Bearer-token-authenticated; CSRF requires session cookies, which C2MD does not use.
Persistent XSS in stored content	No stored user content. Output is generated per-request, screened by Model Armor, and returned.
File upload vulnerabilities	No file uploads. Inputs are JSON-RPC payloads only, validated by Pydantic strict-mode schemas before processing.

This reduced attack surface is a deliberate architectural choice rather than an oversight. The cost is some loss of flexibility — C2MD cannot offer features that depend on persistent state, multi-step workflows, or user-managed data — but the security benefit is substantial: a buyer's threat model for C2MD is materially smaller than for typical enterprise SaaS.

---

3. Identity, Authentication & Access Control

3.1 Customer authentication

C2MD accepts authenticated requests from two identity providers:

Google OAuth2. Tokens are validated against Google's published JWKS at `https://www.googleapis.com/oauth2/v3/certs`. The validator (in `src/auth/oauth2.py`) verifies the token's RS256 signature, issuer (`accounts.google.com` or `https://accounts.google.com`), audience (matching the configured `GOOGLE_CLIENT_ID`), and expiry. Email and stable user identifier (`sub` claim) are extracted for entitlement lookup.

Microsoft Entra ID. Tokens are validated against Microsoft's `/organizations` JWKS at `https://login.microsoftonline.com/organizations/discovery/v2.0/keys`. The validator (in `src/auth/microsoft_oauth2.py`) verifies the token's RS256 signature, issuer (matching the regex pattern for any organisational tenant), audience (matching the configured `MICROSOFT_CLIENT_ID`), and expiry. **Personal**

Microsoft accounts are explicitly rejected by tenant-ID check against the well-known consumer tenant identifier; only work or school accounts are accepted.

Token routing is handled by an unverified-claim peek at the `iss` field (`src/auth/dispatcher.py`), which determines which validator runs. The chosen validator then performs full cryptographic verification independently — the routing decision is never trusted as a security boundary.

3.2 Authorisation

C2MD uses scope-based authorisation. Each skill declares its required scopes; the request handler verifies that the authenticated user's scopes (looked up from a subscription table by email) are a superset of the skill's requirements before dispatching.

The currently-defined scopes are:

Scope	Purpose
`c2md:assess`	Risk assessment of an agent description
`c2md:generate_starter`	Generate a Starter-tier compliance bundle
`c2md:generate_pro`	Generate a Pro-tier compliance bundle
`c2md:generate_journey`	Generate a Journey-tier continuous compliance lifecycle
`c2md:commercial_deploy`	Permit commercial deployment of generated artifacts

The free tier (anonymous or unrecognised authenticated users) has access only to `c2md:assess`.

3.3 Production safety guard

The codebase includes a dev-bypass mechanism (`src/auth/dev_bypass.py`) that allows authentication to be skipped during local development. This bypass is protected by a startup-time check that detects Cloud Run's `K_SERVICE` environment variable and raises `RuntimeError` if dev-bypass is enabled in any environment where `K_SERVICE` is set. In other words, the bypass cannot be turned on in production by an environment-variable mistake — the application refuses to start.

This guard is exercised in the test suite and is verified at every deployment.

3.4 Service account architecture (operator-side)

The C2MD service runs under a dedicated service account (`c2md-agent-runtime`) with narrowly-scoped IAM permissions:

- `roles/aiplatform.user` — to call Vertex AI for LLM inference
- `roles/modelarmor.user` — to call Model Armor for input/output screening
- `roles/secretmanager.secretAccessor` — limited to the single signing-key secret only

The runtime service account **cannot** deploy services, modify infrastructure, or read other secrets. Build operations are performed by a separate, similarly-scoped `cloudbuild-runner` service account. The legacy default Cloud Build service account is granted only the minimum permissions required for backwards compatibility and is not used in active builds.

This per-service-account least-privilege architecture means that a compromised runtime cannot compromise the build pipeline, and a compromised build pipeline cannot read customer-facing data (the runtime service account is the only one with access to customer-facing infrastructure).

3.5 Session management

C2MD is stateless. Each request authenticates independently via Bearer token. There are no server-side sessions, no cookies, no logout endpoint. Token expiry is enforced by `python-jose` (returns JSON-RPC error code `-32004` on expired tokens). There is no password complexity policy because C2MD does not handle passwords — all authentication is delegated to Google or Microsoft.

---

4. Privacy & Data Handling

4.1 Data flow and minimisation

A request to C2MD's `/a2a` endpoint contains a description of the AI agent under assessment. This description may include personal data (developer email addresses, customer references, employee names) depending on what the customer chooses to submit.

Before any LLM invocation, the description is passed through Microsoft Presidio for pseudonymisation. The following entity types are detected and replaced with placeholder tokens (e.g., `<PERSON_1>`, `<EMAIL_1>`):

- `PERSON` (personal names)
- `EMAIL_ADDRESS`
- `PHONE_NUMBER`
- `IBAN_CODE`
- `LOCATION`
- `DATE_TIME`

The mapping from placeholder to original value (the "entity map") is held only in the request's in-memory data structure. It is explicitly never logged, never persisted, and never returned to the caller.

This pseudonymisation is performed *before* any data reaches Vertex AI or Model Armor — the LLM sees only placeholders, not original PII.

Roadmap items in this area (trigger-based):

Item	Trigger
Add EU passport number detection (currently not in Presidio's coverage)	Trigger: First paying customer
Add national ID number detection (German Personalausweis, French INSEE)	Trigger: First paying customer
Per-instance numbered pseudonyms (`PERSON_1` , `PERSON_2` rather than collapsing all `PERSON` entities to one placeholder)	Trigger: First paying customer

4.2 Zero Data Retention (ZDR) configuration

Status: Not yet configured. Tracked in code; will be enabled when upstream SDK support is available.

This deserves prominent treatment because we anticipate it will be one of the first questions a privacy-sensitive buyer asks.

What ZDR is. Google's Vertex AI service operates by default under standard data-usage terms: customer prompts and responses are not used to train Google's models, but transient retention for abuse-monitoring purposes is permitted (typically 24-30 days). ZDR is a configuration mode that disables this transient retention entirely — Google retains nothing once a request completes.

Why C2MD does not currently have ZDR enabled. The `google-genai` Python SDK does not yet expose ZDR as a first-class configuration option. ZDR is documented in Google's published Vertex AI data governance materials (`cloud.google.com/vertex-ai/generative-ai/docs/data-governance`), but configuring it requires either an account-level setting negotiated with Google or programmatic access through SDK features that are still under development.

What is in place today. The C2MD code that calls Vertex AI (`src/llm/vertex_client.py`) contains an explicit `TODO` marker for ZDR configuration, with a reference to Google's documentation. The intent to enable ZDR is committed to the code, not merely planned in a roadmap document.

What this means for customer data. Today, customer agent descriptions submitted to C2MD are processed by Vertex AI under Google's standard data-usage terms. Specifically:

- Google does not use the data to train their models (this is contractually committed in Google Cloud's terms of service)
- Google may retain the data transiently for abuse-monitoring purposes (typically 24-30 days)
- Data does not leave the EU at any point in this flow

Trigger and commitment. ZDR will be enabled within 30 days of the `google-genai` SDK exposing first-class support, OR within 30 days of a paying customer requesting it as a procurement condition (in which case the operator will work with Google support to configure ZDR via account-level settings). Whichever comes first.

For customers requiring ZDR-equivalent guarantees today, this can be achieved through a Private Offer with a contractual commitment to enable ZDR via account-level configuration on or before the customer's first production request.

4.3 Encryption

In transit. All traffic to and from C2MD is TLS-encrypted. The `c2md.getvda.ai` endpoint uses a Google-managed certificate provisioned via ACME (TLS 1.2+ enforced by Google Frontend by default). Internal calls between C2MD and other Google APIs (Vertex AI, Model Armor, Secret Manager) use Google's internal encrypted backbone. There is no plaintext data path at any layer.

At rest. Secrets in Google Secret Manager and container images in Artifact Registry are encrypted at rest using Google-managed encryption keys. Audit logs in Google Cloud Logging are similarly encrypted at rest.

Customer-Managed Encryption Keys (CMEK) are not currently configured. CMEK would allow a customer to provide their own KMS key for encrypting any C2MD-side data at rest. **Trigger: Customer request** — CMEK can be configured on a per-customer basis through a Private Offer and is technically supported by all the Google Cloud services C2MD uses.

4.4 Output screening (Model Armor)

Every artifact in a generated compliance bundle passes through Model Armor's outbound screening before being returned to the caller. The screening configuration includes:

- Prompt injection and jailbreak detection (HIGH confidence threshold)
- Malicious URI detection
- Sensitive Data Protection (SDP) — pattern matching for sensitive data types
- Responsible AI category screening (DANGEROUS category at MEDIUM_AND_ABOVE confidence)

If outbound screening detects an issue, the bundle is rejected with JSON-RPC error code `-32005` and the artifact is not returned to the caller. This protects customers against the model producing inadvertently harmful, biased, or sensitive output.

Model Armor configuration is defined in `infra/terraform/modelarmor.tf` and applies uniformly to every Vertex AI call.

4.5 GDPR roles and Data Processing Agreement (DPA)

Under GDPR, C2MD operates as a **processor** for the agent descriptions submitted by customers. The customer is the **controller**. The single sub-processor is **Google Cloud Platform**, which operates under its own GDPR-compliant DPA available at `cloud.google.com/terms/data-processing-addendum`.

A C2MD-customer DPA template is not yet drafted. This is a documentation gap that will be addressed before the first paying customer signs (Trigger: First paying customer). The DPA will be drafted using IAPP's standard GDPR processor DPA template as a starting point, customised for C2MD's specific data flow, and made available for customer review during procurement.

For customers requiring a DPA today, the operator will work with the customer's legal team to execute a customer-supplied DPA template, modified as necessary to reflect C2MD's actual sub-processor (Google Cloud only) and data-flow characteristics.

4.6 Data subject rights

Because C2MD does not persist customer-supplied data, the practical mechanics of data subject rights requests (access, erasure, portability, rectification) are largely moot at the C2MD layer. The data subject's data exists only transiently during request processing.

Audit logs (Cloud-managed) contain timestamps and request metadata but not the customer's submitted content. If a Data Subject Access Request (DSAR) is received that names a specific data subject and requires evidence of any processing, the operator will work with the customer (as the controller) to provide the available metadata from audit logs. There is no automated DSAR tooling.

Roadmap: Formal DSAR runbook and customer-facing DSAR API. Trigger: Five paying customers.

---

5. Application Security

5.1 Input validation

Every input to C2MD is validated by Pydantic strict-mode schemas before any further processing. The validation layers are:

1. **JSON-RPC envelope validation** (`src/a2a/schemas.py`). Every `/a2a` request is validated as a well-formed JSON-RPC 2.0 request before any handler dispatch.
2. **Skill parameter validation** (each skill module). Every skill validates its parameters against a Pydantic model before any LLM call. Invalid parameters return JSON-RPC error code `-32602` (Invalid params) without invoking any model.
3. **Output schema validation** (`src/output/schemas.py`, `src/output/validator.py`). Every governance artifact (AGENTS.md, SOP.md, etc.) is validated against the corresponding Pydantic model before being included in the response. Cross-field validators enforce, for example, that declared clause counts match actual heading counts.

`pyproject.toml` configures Pydantic in strict mode globally, so type coercion is disabled. A field declared as `int` rejects strings, even string-encoded integers; a field declared as `Literal["pro", "starter"]` rejects any other value.

5.2 Output screening and language safety

Before any artifact is returned to the caller, three layers of output safety apply:

1. **Schema validation** (above): structural correctness
2. **Banned phrase substitution and rejection** (`src/output/language.py`): prevents the model from producing language that overclaims compliance (such as "ensures compliance," "fully compliant," "guarantees regulatory adherence" etc.). Such phrases are either substituted with more accurate alternatives or rejected outright.
3. **Model Armor outbound screening** (above): semantic and pattern-based safety

This three-layer approach means that a single layer's failure (a rule that misses a case) does not produce an unsafe output — subsequent layers catch what the previous missed.

5.3 SQL injection, NoSQL injection

Not applicable. C2MD has no SQL or NoSQL databases. All persistence is through Google-managed APIs (Secret Manager, Artifact Registry, Cloud Logging) that are not user-input-driven.

5.4 Cross-site scripting (XSS), Cross-site request forgery (CSRF)

Not applicable to C2MD's API surface. C2MD has no HTML output, no user-facing web UI, no session cookies, and no browser-rendered content. The API returns JSON-RPC responses to programmatic callers.

5.5 File upload security

Not applicable. C2MD does not accept file uploads. All inputs are JSON-RPC payloads with strict schema validation.

5.6 Static and dynamic application security testing (SAST/DAST)

Status: Automated Software Composition Analysis (SCA) is active in CI as of v1.0 of this document. Static Application Security Testing (SAST) via CodeQL is configured but pending GitHub Advanced Security enablement on the private repository. Dynamic Application Security Testing (DAST) and external penetration testing remain on the roadmap.

Active in CI today:

- **Dependabot** — automated SCA scanning the full Python dependency tree (including transitive dependencies) and GitHub Actions workflow dependencies. Configured for weekly scheduled scans plus on-demand scans triggered by upstream advisories. Opens pull requests for vulnerable or outdated dependencies, grouped by Google Cloud SDK family and by minor/patch versions. Configuration in `.github/dependabot.yml`. Five pull requests opened within the first hours of activation.
- **Dependabot security updates, alerts, and malware alerts** — all enabled at the repository level.

Configured but pending activation:

- **CodeQL** — automated SAST workflow committed at `.github/workflows/codeql.yml` running GitHub's `security-extended` and `security-and-quality` query suites against the Python codebase. The workflow file is in place and triggers on every push to `main`, every pull request, and weekly on a scheduled cron. Activation requires GitHub Advanced Security (GHAS) on the repository. GHAS is only available for repositories owned by GitHub Organizations; the c2md-agent repository currently sits under a personal GitHub account. Migration to a GitHub Organization and GHAS enablement are sequenced under the v1.1 hygiene roadmap (Trigger: First paying customer with revenue ≥ £100K ARR — at which point the recurring GHAS cost is justified by revenue).

Both Dependabot and CodeQL run as free GitHub-hosted services. CodeQL becomes active as soon as the repository sits under an Organization with GHAS enabled.

Compensating controls supporting the automated tooling:

- Comprehensive test suite (133 tests passing as of v1.0, including 15 authentication tests and 27 Model Armor screening tests)
- mypy strict mode enforced; type errors block the build
- Pydantic strict validation throughout the codebase
- Manual security review of every commit by the operator
- Small attack surface (see Section 2.4) reduces the population of vulnerabilities that SAST/DAST typically detect

Roadmap (remaining items):

Item	Trigger
Migrate c2md-agent to a GitHub Organization	First paying customer
Enable GHAS on the c2md-agent repository, activating CodeQL	First paying customer with revenue ≥ £100K ARR
Schedule first external penetration test	Calendar: within 60 days of first paying customer; tester to be a CREST-accredited firm
Add DAST tooling (such as OWASP ZAP) to CI	First paying customer
Establish dependency review SLA (severity-tiered: critical within 7 days, high within 30 days)	First paying customer

5.7 Secrets management

The only secret managed by C2MD is the Ed25519 keypair used to sign Agent Cards and bundle artifacts. This key is stored in

Google Secret Manager with user-managed replication restricted to the `europe-west1` region (single-region for EU residency).

There are no API keys, database credentials, or third-party service secrets in the codebase or in environment variables. OAuth Client IDs (Google and Microsoft) are public identifiers and are intentionally embedded in `cloudbuild.yaml`.

Past incident relevant to this section. On 2026-04-29, the Card-signing private key was briefly included in source uploads to Google Cloud Build's source bucket due to a `.gcloudignore` configuration gap. The exposure was detected during a pre-deploy preview of upload contents, contained by deletion of the affected tarballs from the bucket, and remediated by rotating the keypair (v1 disabled, v2 generated and deployed). The full timeline and remediation are described in Section 9.

---

6. Infrastructure Security

6.1 Hosting and region

C2MD runs on Google Cloud Run in the `europe-west1` region (St. Ghislain, Belgium). EU residency is enforced at multiple layers:

- Cloud Run service: `europe-west1` (configured in `cloudbuild.yaml`)
- Vertex AI client: `europe-west1` hardcoded in `src/llm/vertex_client.py:17` (not just deployment config — a US migration requires a code change and a new commit)
- Model Armor: EU multi-region (configured in `infra/terraform/modelarmor.tf`)
- Secret Manager: user-managed replication, `europe-west1` only (configured in deployment scripts)
- Cloud Build: `europe-west1`
- Artifact Registry: `europe-west1`

A buyer concerned about data residency can verify this end-to-end by reviewing the configuration in the public-facing Agent Card and by inspecting build logs (available on request under NDA).

6.2 Network posture

TLS termination. TLS is terminated at the Google Frontend before traffic reaches the Cloud Run container. The certificate is Google-managed (provisioned via ACME). TLS 1.2+ is enforced by default; older versions are rejected.

Anonymous request rejection. Cloud Run is deployed with `--no-allow-unauthenticated`. All anonymous requests are rejected at HTTP 403 by Google Frontend before any application code executes. This is a separate authentication layer from the application-level Bearer token validation; both must succeed for a request to reach the skill handler.

CORS. Cross-origin requests are restricted to `https://gemini.google.com` only (configured in `src/main.py`). This permits the Gemini agent ecosystem to invoke C2MD as a third-party agent. No wildcard origins are permitted.

Web Application Firewall (WAF). Google Cloud Armor (a WAF service) is not currently configured. Default DDoS protection from Google Frontend applies. **Trigger: First paying customer** for evaluation of whether Cloud Armor is required for the customer's threat model.

VPC connector. None. C2MD calls only Google-managed APIs (Vertex AI, Model Armor, Secret Manager) which are reachable without a VPC connector. No private-network egress is required.

6.3 Container security

The C2MD container is built from `python:3.12-slim` with the following hardening:

- Runs as a non-root `appuser` (added in container Dockerfile)
- Application directory `/app` and Python virtual environment `/app/.venv` are owned by `appuser`
- Container is built with `uv sync --no-dev --frozen` against the committed `uv.lock`, ensuring reproducible builds
- Runtime command uses `uv run --no-sync` — no package management runs at startup, so the runtime image is fully immutable after build
- No SSH access is permitted (Cloud Run does not expose SSH)
- Filesystem is treated as ephemeral; instances are destroyed on scale-down

6.4 Build pipeline security

The build pipeline (`cloudbuild.yaml`) runs under a dedicated `cloudbuild-runner` service account with narrowly-scoped permissions. The pipeline performs:

1. Sign the Agent Card using the keypair in Secret Manager
2. Build the container image
3. Push to Artifact Registry (`europe-west1`)

4. Deploy to Cloud Run with `--service-account=c2md-agent-runtime`

Build sources are uploaded from the developer machine via `gcloud builds submit`. The `.gcloudignore` configuration excludes secrets, infrastructure state files, and binary artifacts from the upload (extensively reviewed and tightened following the 2026-04-29 incident).

6.5 Audit logging

Google Cloud Audit Logs are enabled by default for the C2MD project. The relevant log sinks are:

- `cloudaudit.googleapis.com/activity` — administrative actions (IAM changes, deployments, service account modifications)
- `cloudaudit.googleapis.com/system_event` — Google-managed system events
- `cloudaudit.googleapis.com/access_transparency` — Google personnel access to customer infrastructure (typically zero for this project's configuration)

These logs are retained per Google's defaults (400 days for `_Required` sinks, 30 days for `_Default` sinks). They cover all administrative changes and are not erasable by the operator.

The application also emits structured logs via Python's `structlog` library to Google Cloud Logging. Six event categories are defined (`AGENT_DECISION` , `FRAMEWORK_INTEGRITY` , `COMPLIANCE_BOUNDARY` , `AGENT_LIFECYCLE` , `A2A_PROTOCOL` , `DATA_GOVERNANCE`). Customer prompt and response content is **never** included in logs.

---

7. Supply Chain & Dependencies

7.1 Dependency surface

C2MD uses a deliberately small dependency tree. Production dependencies declared in `pyproject.toml` are:

Package	Purpose
`fastapi`	HTTP framework
`uvicorn[standard]`	ASGI server
`pydantic`	Schema validation
`google-genai`	Vertex AI client
`google-cloud-modelarmor`	Model Armor client
`google-cloud-secret-manager`	Secret retrieval
`google-cloud-firestore`	(declared, not yet used; reserved for future entitlement migration)
`presidio-analyzer` , `presidio-anonymizer`	PII pseudonymisation
`structlog`	Structured logging
`python-jose[cryptography]`	JWT validation
`httpx`	HTTPS client (used for JWKS fetch)
`pyyaml`	Configuration parsing
`tenacity`	Retry logic

All packages are mainstream and actively maintained. The full transitive dependency tree is locked in `uv.lock` (committed to the repository) and reproduced exactly on every build via `uv sync --frozen` .

7.2 Vulnerability management

Status: Automated dependency vulnerability scanning is active in CI as of v1.0 of this document via Dependabot. Static code analysis via CodeQL is configured but pending GitHub Advanced Security enablement (see Section 5.6).

Active in CI today:

- **Dependabot** scans the full dependency tree (Python production dependencies via the `uv` ecosystem, plus GitHub Actions workflow dependencies) on a weekly schedule and on-demand when upstream advisories are published. Pull requests are opened automatically for vulnerable or outdated dependencies. Configuration in `.github/dependabot.yml` .
- **Dependabot security alerts** are enabled at the repository level — vulnerabilities discovered in the dependency graph

generate alerts visible in the GitHub Security tab.

- **Dependabot security updates** are enabled — Dependabot opens pull requests automatically to resolve open security alerts where patches are available.
- **Dependabot malware alerts** are enabled — alerts trigger when malware is detected in dependencies.

Compensating controls supporting the automated tooling:

- Small dependency surface (13 direct production dependencies, all from major vendors)
- Reproducible builds via committed `uv.lock` — dependency versions cannot drift unintentionally
- All Google Cloud SDK dependencies receive security updates from Google directly through SDK version bumps
- Manual review of every Dependabot pull request by the operator before merge

Roadmap (remaining items):

Item	Trigger
Activate CodeQL by migrating to GitHub Organization and enabling GHAS	First paying customer with revenue ≥ £100K ARR
Add Trivy or Grype scanning of container images in CI (complementary to dependency-level Dependabot scanning)	First paying customer
Establish a dependency review SLA (severity-tiered: critical within 7 days, high within 30 days)	First paying customer
Subscribe to security advisories for Google Cloud SDK packages	Calendar: within 30 days of v1.0 publication

7.3 Container base image

The C2MD container is built from `python:3.12-slim` (a Debian-based minimal Python image maintained by Docker). This base image receives security updates from upstream and is refreshed each time the build pipeline runs.

Known limitation: the build pipeline does not currently use `docker pull --pull=always` to force base image refresh on every build. Layer caching may serve a previously-pulled base image even when newer versions are available upstream.

Roadmap: Add explicit base-image refresh policy with periodic forced rebuilds. Trigger: First paying customer.

7.4 Build artifact integrity

Container images pushed to Artifact Registry are tagged with both `:latest` and the immutable Git commit SHA (e.g., `:89b005f...`). This allows precise rollback to any historical revision and provides a verifiable link between deployed code and committed source.

The Agent Card is signed with the C2MD signing keypair at build time, before the container is deployed. This signature is verifiable by any client (the public key is published at `c2md.getvda.ai/.well-known/agent-card-public-key.pem`) and serves as evidence that the deployed agent matches the signed configuration.

7.5 Sub-processor management

C2MD's only sub-processor is Google Cloud Platform (see Section 2.2). There are no other third-party services in the runtime path.

Changes to the sub-processor list (e.g., adding a third-party logging provider, monitoring service, or analytics tool) would constitute a material change to the data processing posture and would be communicated to customers in advance via a sub-processor change notice. **Trigger: First paying customer** for the establishment of a formal sub-processor change notification process.

---

8. Operational Resilience

8.1 Service availability

C2MD inherits the Cloud Run service-level objective: 99.95% monthly uptime per the Google Cloud Run SLA. There is no application-layer SLA on top of this — the operator does not currently commit to an availability target beyond what Cloud Run provides.

The Cloud Run deployment is configured for scale-to-zero with a maximum of 10 concurrent instances. Cold-start latency is typically 3-5 seconds. There is no minimum-instance configuration, which means the first request after a period of inactivity incurs cold-start latency.

Roadmap: Evaluate minimum-instance configuration based on actual usage patterns. Trigger: Five paying customers.

8.2 Disaster recovery

Status: No application-layer DR plan. Single-region by design.

C2MD operates in a single Google Cloud region (`europe-west1`) by deliberate choice — the EU residency commitment forecloses multi-region failover that would route traffic to non-EU regions.

In the event of a sustained `europe-west1` outage:

- The C2MD service is unavailable until the region recovers
- Existing customer requests in flight at the time of failure are dropped
- No data is lost (no persistent customer data exists)
- The signing keypair in Secret Manager is replicated within `europe-west1` only (single-region replication is required for EU residency); during a regional outage, the Card cannot be re-signed

The Cloud Run SLA covers regional availability. Sustained regional outages are rare events historically (Google has not had a multi-day `europe-west1` outage in the service's history) but are not impossible.

For customers requiring multi-region resilience while preserving EU residency: a deployment to a second EU region (e.g., `europe-west4` Netherlands) is technically feasible but not currently configured. **Trigger: Customer request** — multi-region EU deployment can be configured under Private Offer with the additional cost of running redundant infrastructure.

8.3 Backup and restore

Source code: Two copies exist — the GitHub origin and the operator's local working tree. There is no third copy. **Roadmap:** Establish an automated mirror to a second Git host (likely GitLab or Codeberg). Trigger: First paying customer.

Container images: Stored in Artifact Registry, which Google manages with their own backup and durability commitments.

Secret Manager (signing keypair): The keypair v1 is disabled but retained in Secret Manager's version history; v2 is the active key. If v2 were lost, v1 could be re-enabled to recover signature continuity, after which a new v3 would be generated and rotated in. The procedure has not been formally tested.

Roadmap: Document and test the keypair recovery procedure. Trigger: Calendar — within 60 days of first paying customer.

Customer data: Not applicable. C2MD does not persist customer data.

8.4 Capacity management

Cloud Run's `--max-instances=10` setting caps concurrent compute. Each instance handles one request at a time by default. The system can therefore process up to 10 concurrent requests; the 11th and beyond are queued by Cloud Run with a brief delay.

For customers with high-burst usage requirements, the maximum instance count can be increased per-customer under Private Offer. Vertex AI quota (which is typically the binding constraint on throughput, not Cloud Run instances) can also be increased through Google Cloud quota requests.

Roadmap: Establish a capacity model based on actual usage patterns. Trigger: First paying customer.

8.5 Business continuity

C2MD is operated by a single individual. In the event of operator unavailability (illness, incapacitation, or worse), the service continues to run on Cloud Run without intervention until something breaks (e.g., a TLS certificate fails to renew, a Google Cloud quota is exhausted, a security update requires manual deployment). This is a genuine business continuity risk that is not solved by technical controls.

Compensating controls today:

- All operational state is in Google Cloud (audit logs, IAM, Secret Manager, deployments) — not in any operator-personal storage
- The codebase is on GitHub with a recovery email tied to the operator's identity
- Documentation (this document, the V4.1 framework whitepaper, deployment scripts) is comprehensive enough that a competent successor could resume operations

Roadmap:

Item	Trigger
Establish a documented operational handover plan	Customer request
Evaluate adding a second authorised operator	Series funding event
Evaluate engaging a managed-service partner for operational continuity	Customer request

9. Incident Response

9.1 Status

No formalised incident response plan currently exists. This is a documented gap. The compensating reality is that the small attack surface (Section 2.4) and the comprehensive audit trail (Section 6.5) make most incidents reconstructible from primary evidence, and the operator has handled one real incident successfully (Section 9.3 below).

9.2 What incident handling looks like today

Incident detection sources, in approximate order of likely first-detection:

- **Cloud Build pre-deploy preview** — the operator routinely inspects the upload contents before triggering a build. This is what detected the 2026-04-29 incident.
- **Google Cloud Audit Logs** — administrative changes are logged automatically. Anomalous patterns (unexpected IAM grants, unfamiliar service accounts, unusual API calls) would be visible in the audit log review.
- **Cloud Run application logs** — error rates, authentication failures, and Model Armor block events are structured-logged via `structlog` and visible in Cloud Logging.
- **Customer report** — customers experiencing issues can email `security@getvda.ai` (active mailbox forwarding to the operator).
- **Google Cloud security alerting** — Google may notify the project owner of security-relevant events (compromised credentials detected by Google's monitoring, unusual API access patterns, etc.).

Incident response, when triggered, follows a four-step pattern that emerged from the 2026-04-29 incident:

1. **Detect** — identify the incident scope and severity
2. **Contain** — stop the active threat (delete exposed data, rotate credentials, disable compromised service accounts)
3. **Remediate** — fix the underlying cause (configuration change, code change, process change)
4. **Document** — capture the timeline in commit messages, audit log queries, and (going forward) a formal postmortem

Roadmap:

Item	Trigger
Publish `SECURITY.md` at the repository root with disclosure policy	Calendar: within 14 days of v1.0 publication
Draft formal IR plan with severity classification (S1-S4), notification timelines, and customer communication templates	First paying customer
Establish IR runbook covering common incident types (credential compromise, service degradation, data exposure, sub-processor outage)	First paying customer
Schedule first IR tabletop exercise	Five paying customers
Establish customer incident notification SLAs (e.g., 24 hours for material incidents)	First paying customer

9.3 Case study: 2026-04-29 — Card-signing private key exposure

This is the only real incident in C2MD's operational history to date. It is described in detail here because the response demonstrates how incident handling works in practice, even without a formal IR plan.

Summary. During the initial Cloud Run deployment work, the Card-signing private key file (`.secrets/card-signing-key.pem`) was inadvertently included in source uploads to the Cloud Build source bucket due to a configuration gap in `.gcloudignore`. The exposure was detected during a pre-deploy review of upload contents, contained within minutes, and remediated within hours. No customer impact occurred (no customers existed at the time).

Detection (T+0). During iteration on the build pipeline, the operator observed that the Cloud Build source upload was approximately 139 MiB — significantly larger than the expected ~10 MiB. Investigation via `gcloud meta list-files-for-upload` surfaced that `.secrets/card-signing-key.pem`, Terraform state files, and a 153 MB Windows binary (a Terraform provider) were all being uploaded.

Containment (T+15 minutes). The two Cloud Build source tarballs containing the private key (from previous failed build attempts) were deleted from `gs://c2md-493808_cloudbuild/source/` using `gcloud storage rm`. This removed the key from Google Cloud's storage. Audit logs were verified to show only the operator's own account had read access to the bucket; no third-party access was observed in the audit trail during the exposure window.

Remediation (T+1 hour).

1. The `.gcloudignore` was extended to exclude `.secrets/`, `*.pem`, `*.key`, `*.crt`, `*.tfstate`, `.terraform/`, `*.exe`, `.claude/`, and related patterns. Upload size dropped from 139 MiB to 553 KiB (approximately 250x reduction). Committed in `293fd65`.
2. The Card-signing keypair was rotated: a new Ed25519 keypair was generated, the new private key uploaded to Secret Manager as a new version, the old version (v1) disabled. The public key in the repository was updated. The Agent Card was re-signed with the new key (v2). Committed in `6d1acd3`.
3. The signing key identifier (`keyId`) field was updated from `c2md-card-v1` to `c2md-card-v2` in a follow-up commit (`e2e2fa3`).

Outcome. The exposure window was approximately 4 hours from first upload to keypair rotation. During that window, the only IAM principals with read access to the Cloud Build source bucket were the operator's own Google account, the Cloud Build runner service account, and Google's internal Cloud Build service agent. No external access occurred per audit log review. The rotated keypair v2 is the active signing key as of v1.0 of this document.

Lessons applied.

- `.gcloudignore` is now reviewed before every significant build pipeline change
- Pre-deploy upload review using `gcloud meta list-files-for-upload` is now part of the standard pre-build checklist
- The keypair rotation procedure is documented (informally, in commit history) and can be repeated if needed

This is offered as evidence that incident handling at C2MD, while not formalised, is functional. A formalised IR plan will codify this informal practice into documented procedures, severity classification, and customer notification timelines (per the roadmap above).

---

10. Compliance & Regulatory

10.1 Regulatory regimes the operator complies with

C2MD operates as a software-as-a-service product based in the European Union. The following regulatory regimes apply directly to C2MD's operations:

General Data Protection Regulation (GDPR / Regulation 2016/679). C2MD operates as a processor for any personal data submitted by customers via `/a2a` requests. The customer is the controller. The single sub-processor is Google Cloud Platform. EU residency is enforced at the architectural level (Section 6.1).

EU AI Act (Regulation 2024/1689). C2MD is itself an AI system under the EU AI Act. It is classified as **limited-risk** under the Act because it generates content (compliance documentation) but is not deployed in any of the high-risk Annex III contexts. C2MD's transparency obligations under Article 50 are met via the publicly verifiable Agent Card and the published methodology (`github.com/mikerawsonnz/vda-md-framework`).

UK GDPR and Data Protection Act 2018. For UK-resident data subjects whose data is submitted to C2MD by an EU controller, the same processor obligations apply.

10.2 Regulatory regimes C2MD does not currently address

For full transparency, the following regimes are *not* currently in scope:

- **HIPAA (US health data).** C2MD has not undergone HIPAA business associate certification. Customers processing protected health information should not submit it to C2MD until a Business Associate Agreement is executed. **Trigger: Customer request** for HIPAA assessment and BAA execution.
- **PCI DSS.** C2MD does not handle payment card data. Customers must not submit cardholder data via `/a2a` requests. PCI DSS scoping is therefore not applicable.
- **CCPA (California Consumer Privacy Act).** C2MD has not formally assessed CCPA applicability. The processor/controller framing under GDPR provides functionally equivalent protections for California residents whose data is submitted by EU controllers. **Trigger: First California-domiciled customer** for formal CCPA compliance assessment.
- **Sector-specific regimes** (FINRA for US broker-dealers, SR 11-7 for US bank model risk management, EBA guidelines for EU banking, etc.). Customers in regulated sectors are responsible for evaluating C2MD's fit within their sector's regime. The published methodology and security overview (this document) are designed to support that evaluation.

10.3 Certifications

Status: No third-party-attested certifications currently held.

C2MD does not currently hold SOC 2, ISO 27001, ISO 27701, ISO 42001, or any other third-party-audited certification. This is a material gap relative to enterprise procurement expectations, and is acknowledged here without qualification.

Compensating reality:

- C2MD inherits Google Cloud's certifications (SOC 1/2/3, ISO 27001/27017/27018, PCI DSS, FedRAMP, and others) for the underlying infrastructure layer. The full list is at `cloud.google.com/security/compliance`. This covers physical security, environmental controls, and Google's own organisational controls.
- The application-layer security posture is described in detail in this document. A buyer's security team can verify the architectural claims by inspection of the public Agent Card, the published methodology, and (under NDA) the private repository.

SOC 2 readiness posture.

While C2MD does not hold a current SOC 2 attestation, the *substance* of many SOC 2 Trust Services Criteria controls is already implemented at the technical layer. A buyer evaluating C2MD against the SOC 2 control framework will find evidence in this document for the following criteria:

Trust Services Category	Substantive controls in place
Security (Common Criteria)	Multi-factor authentication via delegated IdPs (Sections 3.1-3.2); least-privilege IAM with per-service service accounts (Section 3.4); encryption in transit and at rest (Section 4.3); audit logging via Cloud Audit Logs and structured application logs (Section 6.5); production safety guards preventing dev-bypass in production (Section 3.3); incident response demonstrated in practice (Section 9.3)
Availability	Cloud Run 99.95% SLA inheritance (Section 8.1); stateless architecture eliminating single-instance failure modes (Section 2.3); honest acknowledgement of single-region constraint and the trade-off it represents (Section 8.2)
Processing Integrity	Pydantic schema validation at every boundary (Section 5.1); schema validation of bundle outputs including cross-field validators (Section 5.2); Model Armor double-screening on every LLM interaction (Section 4.4); banned-phrase filtering preventing overclaiming (Section 5.2)
Confidentiality	EU residency hardcoded across all infrastructure components (Section 6.1); single sub-processor (Section 7.5); no persistent customer data store (Section 2.3); PII pseudonymisation before LLM calls (Section 4.1)
Privacy	Processor-not-controller posture (Section 4.5); transient-only data handling (Section 2.3); PII pseudonymisation (Section 4.1); explicit ZDR roadmap (Section 4.2)

What is missing for SOC 2 attestation is therefore not the underlying controls but rather:

1. **Formal documented policies** — security policy, access control policy, change management policy, vendor management policy, etc. These exist in operational practice but are not yet written as standalone documents.
2. **Third-party auditor attestation** — an independent CPA firm examining evidence and producing a SOC 2 report.
3. **Observation period** (Type II only) — 6+ months of operational evidence under examined controls.

A buyer evaluating C2MD's distance to SOC 2 should read this as: the technical readiness is substantively in place; the formal attestation work is the gap.

Roadmap:

Certification	Trigger	Notes
SOC 2 Type I	First paying customer with revenue ≥ £100K ARR	Achievable in 3-6 months given the small scope; would establish the control framework before Type II observation period
SOC 2 Type II	Twelve months after Type I commencement	Type II requires 6+ months of operational evidence; can begin observation as soon as Type I controls are documented
ISO 27001	Series funding event	Larger scope and cost; likely deferred until team expansion makes the operational discipline sustainable
ISO 42001 (AI management systems)	Customer request	Newer standard specifically for AI; may become a required differentiator for AI-specific procurement

10.4 Privacy programme

A formal privacy programme (data protection officer designation, privacy impact assessments for material changes, data breach notification procedures, data subject rights handling) is not currently in place at the documented-process level. The technical realities (no persistent customer data, EU residency, single sub-processor, processor-not-controller posture) reduce the practical surface area, but the programme documentation is a gap.

Roadmap:

Item	Trigger
Designate a Data Protection Officer (or external DPO-as-a-service if scale doesn't justify internal hire)	First paying customer in a sector requiring DPO designation under GDPR Article 37
Draft privacy programme documentation (privacy policy, breach notification procedure, DSAR handling procedure, vendor risk assessment for Google Cloud)	First paying customer
Conduct a DPIA on C2MD's own processing operations (recursive — using C2MD itself or a manual process)	First paying customer
Publish a public privacy notice at `getvda.ai/privacy`	Calendar: within 30 days of v1.0 publication

10.5 Public disclosure documents

The following documents are publicly available and form part of C2MD's transparency posture:

- **Agent Card:** `https://c2md.getvda.ai/.well-known/agent-card.json` — declares all capabilities, security schemes, signing keys, and integration patterns
- **Public signing key:** `https://c2md.getvda.ai/.well-known/agent-card-public-key.pem` — for independent verification of any signed bundle
- **VDA-MD Framework v4.1 whitepaper:** `https://github.com/mikerawsonnz/vda-md-framework` — the methodology underlying C2MD's compliance bundles, published under CC-BY-4.0
- **This document:** `https://getvda.ai/security` — security and privacy overview

The following documents are *not yet* publicly available but are planned:

- **`SECURITY.md`** at the repository root with vulnerability disclosure policy. **Calendar: within 14 days of v1.0 publication.**
- **Privacy notice** at `getvda.ai/privacy`. **Calendar: within 30 days of v1.0 publication.**
- **Sub-processor list** as a standalone, dated document. **Trigger: First paying customer.**
- **Customer-facing changelog** for material security or privacy changes. **Trigger: First paying customer.**

---

11. Roadmap & Commitments — Consolidated

This section consolidates every roadmap commitment made throughout this document into a single table. Buyers can use this as a single-page summary of what changes when, without having to re-read each section.

11.1 Calendar-dated commitments

These have specific date commitments tied to the publication of v1.0 of this document.

Item	Status
Provision `security@getvda.ai` mailbox	✅ Completed ahead of v1.0 publication
Publish `SECURITY.md` at repository root	Within 14 days of v1.0 publication
Subscribe to security advisories for Google Cloud SDK packages	Within 30 days of v1.0 publication
Publish privacy notice at `getvda.ai/privacy`	Within 30 days of v1.0 publication

11.2 Trigger: First paying customer

These items begin within 30 days of contract signature with the first paying customer, targeted complete within 90 days unless otherwise noted.

Item	Section
Add Trivy or Grype scanning of container images in CI (complementary to dependency-level Dependabot scanning)	7.2
Establish dependency review SLA (severity-tiered)	7.2
Add DAST tooling to CI	5.6
Add EU passport number detection to PII scrubber	4.1
Add national ID number detection to PII scrubber	4.1
Add per-instance numbered pseudonyms in PII scrubber	4.1
Draft customer DPA template	4.5
Draft formal IR plan with severity classification	9.2
Establish IR runbooks for common incident types	9.2
Establish customer incident notification SLAs	9.2
Establish automated source code mirror to second Git host	8.3
Establish capacity model based on actual usage patterns	8.4
Draft privacy programme documentation	10.4
Conduct DPIA on C2MD's own processing operations	10.4
Establish formal sub-processor change notification process	7.5
Evaluate Google Cloud Armor (WAF) requirement	6.2
Add explicit base-image refresh policy for container builds	7.3
Publish standalone sub-processor list	10.5
Establish customer-facing changelog	10.5
Migrate c2md-agent repository to a GitHub Organization (prerequisite for enabling GHAS at higher revenue trigger)	5.6 / 7.2

Item	Section	Calendar-anchored
Document and test keypair recovery procedure	8.3	Within 60 days of first paying customer
Schedule first external penetration test (CREST-accredited firm)	5.6	Within 60 days of first paying customer

11.3 Trigger: First paying customer with revenue ≥ £100K ARR

Item	Section
Begin SOC 2 Type I assessment	10.3
Enable GitHub Advanced Security on the c2md-agent repository, activating CodeQL SAST	5.6 / 7.2

11.4 Trigger: Five paying customers

Item	Section
Formal DSAR runbook and customer-facing DSAR API	4.6
Schedule first IR tabletop exercise	9.2
Evaluate minimum-instance Cloud Run configuration	8.1

11.5 Trigger: Twelve months after SOC 2 Type I commencement

Item	Section
Begin SOC 2 Type II observation period	10.3

11.6 Trigger: Customer request

These items can be negotiated into procurement on a per-customer basis under Private Offer.

Item	Section
Customer-Managed Encryption Keys (CMEK) configuration	4.3
Multi-region EU deployment for resilience	8.2
Increased Cloud Run maximum instance count	8.4
Engagement of managed-service partner for operational continuity	8.5
Documented operational handover plan	8.5
HIPAA assessment and Business Associate Agreement	10.2
ISO 42001 (AI management systems) certification	10.3

11.7 Trigger: Series funding event

Item	Section
Evaluate adding a second authorised operator	8.5
Begin ISO 27001 certification programme	10.3

11.8 Trigger: First California-domiciled customer

Item	Section
Formal CCPA compliance assessment	10.2

11.9 Trigger: First paying customer in sector requiring DPO

Item	Section
Designate Data Protection Officer (internal or DPO-as-a-service)	10.4

11.10 Trigger: Upstream SDK feature availability

Item	Section	Detail
Enable Zero Data Retention (ZDR) for Vertex AI calls	4.2	Within 30 days of `google-genai` SDK exposing first-class ZDR support, OR within 30 days of customer request (whichever first)

12. Appendix: Frequently Asked Questions

This appendix addresses questions that experienced security reviewers and procurement teams typically ask when evaluating a vendor in C2MD's category and stage. Each answer is grounded in the substance described elsewhere in this document.

12.1 "Where is customer data processed and stored?"

All processing happens in Google Cloud's `europe-west1` region (St. Ghislain, Belgium). No customer data is persisted; data exists only transiently in memory during request processing. EU residency is enforced at the architectural level — the Vertex AI client has the region hardcoded in source (`src/llm/vertex_client.py:17`), so a misconfigured deployment cannot accidentally route inference outside the EU.

12.2 "What's your SOC 2 status?"

No SOC 2 currently. SOC 2 Type I is roadmapped for "first paying customer with revenue ≥ £100K ARR" — see Section 10.3. Type II requires a 6+ month observation period and follows Type I. C2MD inherits Google Cloud's SOC 2 certifications for the infrastructure layer; the application-layer SOC 2 is the gap. The substantive technical controls behind a SOC 2 attestation are largely in place — see the SOC 2 readiness table in Section 10.3.

12.3 "Can you sign our DPA?"

A C2MD-side DPA template is not yet drafted (Trigger: First paying customer in Section 4.5). For customers needing a DPA today, the operator will work with the customer's legal team to execute a customer-supplied DPA template, modified to reflect C2MD's actual single-sub-processor architecture (Google Cloud Platform).

12.4 "What happens if a region outage takes you down?"

Service is unavailable until `europe-west1` recovers. There is no application-layer DR or multi-region failover by deliberate choice — multi-region failover risks routing traffic to non-EU regions, which conflicts with the EU residency commitment. For customers requiring multi-region resilience while preserving EU residency, deployment to a second EU region (e.g., `europe-west4`) is configurable under Private Offer (Trigger: Customer request).

12.5 "Have you been penetration tested?"

Not yet. The first penetration test is calendar-committed for "within 60 days of first paying customer" using a CREST-accredited firm (Section 5.6). The compensating reality is the small attack surface (Section 2.4) and the comprehensive automated test suite (133 tests as of v1.0).

12.6 "Is your code peer-reviewed before deployment?"

Single-operator development, so no peer review process exists. Compensating controls: comprehensive Pydantic schema validation, mypy strict-mode type checking, structured logging across six event categories, output screening via Model Armor, banned-phrase filtering, and the production safety guard (`K_SERVICE` check that prevents dev-bypass in production). All commits use conventional commit prefixes; the full history is auditable via Git.

12.7 "What's your most serious security incident to date?"

The 2026-04-29 Card-signing key exposure, described in detail in Section 9.3. Detection within minutes via pre-deploy upload review; containment within 15 minutes by tarball deletion; full remediation including key rotation within 4 hours. No customer impact (no customers at the time). The incident is described openly in this document because the response demonstrates how incident handling works in practice, even without a formal IR plan.

12.8 "Do you scan dependencies for vulnerabilities?"

Yes — Dependabot is active in CI as of v1.0 of this document. It scans the full Python dependency tree and GitHub Actions workflow dependencies on a weekly schedule and on-demand when upstream advisories are published, opens pull requests for vulnerable or outdated packages, and is paired with Dependabot security alerts and security updates at the repository level. Static code analysis via CodeQL is configured (workflow committed at `.github/workflows/codeql.yml`) but pending GitHub Advanced Security enablement, which is sequenced under the v1.1 hygiene roadmap. See Sections 5.6 and 7.2 for full detail.

12.9 "Is ZDR enabled on your LLM calls?"

Not yet — see the dedicated treatment in Section 4.2. Blocked on `google-genai` SDK exposing first-class ZDR support, tracked in code with a `TODO` marker. Will be enabled within 30 days of either SDK availability or customer request as a procurement condition. Today, customer data is processed by Vertex AI under standard Google data-usage terms (no model training; transient retention for abuse monitoring; data does not leave the EU).

12.10 "What if you (the single operator) become unavailable?"

This is named openly in Section 8.5 as a real business continuity risk. Compensating controls: all operational state is in Google Cloud rather than operator-personal storage; codebase is on GitHub; comprehensive documentation (this document, the framework whitepaper, deployment scripts) supports successor competence. For customers requiring stronger continuity guarantees, a managed-service partner engagement is configurable under Private Offer (Trigger: Customer request, Section 8.5).

12.11 "Can we audit your code?"

The methodology is published openly at `github.com/mikerawsonnz/vda-md-framework` under CC-BY-4.0. The C2MD agent's source code is in a private GitHub repository; access can be granted under NDA on request. The Agent Card and public signing key are continuously published and verifiable.

12.12 "Why is so much of your roadmap triggered by 'first paying customer' rather than calendar dates?"

C2MD is pre-revenue. Calendar commitments without funding to back them are aspirational rather than honest. Trigger-based

commitments mean: the moment the operating entity has paying customers, remediation work begins immediately and is sequenced by genuine procurement priority. Calendar commitments are reserved for items the operator can deliver without additional resources (publishing `SECURITY.md`, provisioning a security mailbox, publishing a privacy notice).

12.13 "How do we reach you for security disclosures?"

Send security disclosures to `security@getvda.ai` with subject line beginning "SECURITY DISCLOSURE." Acknowledgement within 24 hours. Coordinated disclosure preferred; PGP key available on request.

For privacy enquiries, use `privacy@getvda.ai`. For all other business correspondence including Private Offer requests, use `mike@getvda.ai`.

12.14 "Are you GDPR-compliant?"

C2MD operates as a processor under GDPR. The architectural claims (EU residency, no customer data persistence, pseudonymisation before LLM calls, single sub-processor, processor-not-controller posture) provide the substantive basis for GDPR compliance. The documentation gap (formal privacy programme, DPA template, DPIA on C2MD's own processing) is a roadmap item with first-paying-customer trigger. "GDPR-compliant" is a status that requires programmatic evidence, not a single declaration; the substance is here, the formal documentation is in progress.
