Cloud Governance & Control Framework Architecture

Views & Beyond Documentation Package

Version: v0.1 | Status: Draft

Owner: Cloud Platform & Governance Team

Date: December 12, 2025

Purpose: Provide a stakeholder-ready architecture description for the Cloud Governance & Control Framework, organized using Views & Beyond. This package includes the Context, Logical, Process, and Deployment views, plus "Beyond Views" information such as control catalog, evidence model, rationale, risks, and documentation roadmap.

Contents

Document Control

Revision History

Version	Date	Author	Change Summary
0.1	December 12, 2025	Cloud Platform & Governance Team	Initial Views & Beyond package template and starter content.

Distribution

- Cloud Platform Engineering
- Security / IAM
- FinOps
- Compliance / Audit
- Application/Product Teams

Chapter 1

Architecture Overview

1.1 System Scope & Boundary

In scope: Policy definition, guardrails/control plane, automated enforcement, monitoring, evidence collection, reporting, and optimization loops for cloud usage across one or more providers.
Out of scope: Application business logic and domain-specific controls not related to cloud platform governance (unless explicitly integrated as control signals/evidence).

1.2 Business Goals & Outcomes

- Enable secure, compliant, and cost-effective cloud adoption without constraining delivery velocity.
- Provide standardized guardrails (preventive controls) and continuous monitoring (detective controls).
- Automate audit readiness through evidence-as-a-product (continuous control evidence).
- Establish a continuous improvement loop for policies, controls, and platform capabilities.

1.3 Stakeholders & Concerns (Driver Matrix)

1.3.1 Stakeholder Catalog

Stakeholder	Role	Primary Concerns
Cloud Governance Team / CCoE	Policy & guardrails	Consistency, exceptions, adoption, org-wide standards
Security / IAM	Security baseline	Least privilege, identity assurance, monitoring, incident response
FinOps	Cost governance	Allocation, budgets, anomaly detection, optimization
Platform Engineering	Enablement	Landing zones, IaC modules, drift prevention, operability
Compliance / Audit	Assurance	Evidence quality, control mapping, reporting cadence
Product Teams	Consumers	Self-service, speed, clarity of boundaries, minimal friction

1.3.2 Stakeholder–View Mapping

Stakeholder	Views Needed	Decisions Enabled
Security / IAM	Context, Logical, Process, Beyond (Controls/Evidence)	Baselines, access patterns, detections
FinOps	Context, Process, Beyond (Cost Controls/Metrics)	Budgeting, chargeback, optimization priorities
Audit	Beyond (Control Catalog/Evidence), Deployment	Audit readiness, evidence sufficiency
Platform Eng	Logical, Deployment, Process	Reference implementations, operations model
Product Teams	Context, Process (key scenarios)	How to build within guardrails; escalation paths

1.4 Architectural Drivers

1.4.1 Key Quality Attributes

- **Security:** enforce least privilege and secure-by-default configurations.
- **Compliance:** continuous compliance posture and auditable evidence trails.
- **Agility:** self-service enablement with minimal governance friction.
- **Scalability:** handle org-wide resource growth and multi-account/subscription expansion.
- **Reliability/Operability:** monitoring, alerting, runbooks, and clear ownership.

1.4.2 Constraints & Assumptions

- Governance must be **automation-first** (policy-as-code, IaC, continuous monitoring).
- Multi-cloud support may be required (or planned); provider-specific controls must map to common abstractions.

- All controls must have an **evidence model** (what is collected, where stored, retention, and access).

1.5 Document Roadmap (How to Use This Package)

- Start with **Context View** to understand boundary, actors, and external dependencies.
- Use **Logical View** to understand the core governance subsystems and responsibilities.
- Use **Process View** to see runtime behavior and key governance scenarios.
- Use **Deployment View** to understand how the system maps to cloud/provider/tooling infrastructure.
- Use **Beyond Views** for controls, evidence, rationale, risks, and mappings.

Chapter 2

Context View

2.1 View Packet Summary

View type: Context (System Boundary)
Primary stakeholders: Product teams, Security/IAM, FinOps, Compliance/Audit, Platform Engineering
Primary concerns: Who interacts with governance, what is inside vs. outside, trust boundaries, major integrations, and data/evidence flows.

2.2 Context Diagram (Primary Presentation)

Insert your context diagram here. Recommended: C4 Context-style or a clean boundary diagram.

Figure 2.1: Context View: System boundary and external actors/systems.

2.3 External Actors & Systems

Actor/System	Type	Interaction
Product Teams	Human/org	Consume self-service catalog, deploy via IaC pipelines within guardrails
Security/IAM	Org	Defines security baseline; consumes alerts, posture reports
FinOps	Org	Defines budgets/allocations; consumes cost anomaly signals and KPI dashboards
Audit/Compliance	Org	Requests/consumes evidence, control mappings, audit reports
Cloud Provider(s)	External system	Policy enforcement primitives, telemetry, resource APIs
CI/CD Platform	External system	Policy gates, drift checks, release approvals, evidence emission
SIEM/Logging	External system	Centralized security telemetry ingestion and alerting
CMDB/Asset Inventory	External system	Asset ownership, service mapping, lifecycle tracking

2.4 Trust Boundaries & Data Flows

- Governance plane must be logically separated from workload planes (accounts/subscriptions/projects).
- Evidence and logs are sensitive assets; access must be least-privilege and audited.
- Key flows:
  - **Policy intent → Policy-as-code → Enforcement**
  - **Telemetry → Detection → Tickets/Notifications**
  - **Control results → Evidence store → Audit reporting**

2.5 Context View: Rationale & Notes

- This view is intentionally stable; details belong in Logical/Deployment views.
- Provider-specific services are abstracted as “Cloud Provider APIs/Telemetry” to support multi-cloud evolution.

Chapter 3

Logical View

3.1 View Packet Summary

View type: Logical / Module decomposition
Primary stakeholders: Platform Engineering, Security/IAM, FinOps, Architecture
Primary concerns: Responsibilities, module boundaries, interfaces, data stores, and integration points.

3.2 Logical Decomposition (Primary Presentation)

Insert your logical/module diagram here. Recommended: "Policy/Control Plane/Execution/Evidence" layered module diagram.

Figure 3.1: Logical View: Core modules and their primary responsibilities.

3.3 Module Catalog (Element Catalog)

Module	Type	Responsibilities
Policy Library	Data/Knowledge	Policies, standards, control objectives, exception rules, versioning
Policy-as-Code Engine	Service	Encodes policies into enforceable rules; integrates with provider/IaC/CI gates
Landing Zone Management	Service	Org/account/subscription baseline, network baseline, identity baseline hooks
Self-Service Catalog	Service	Approved patterns, IaC modules, golden paths, request workflows
Continuous Monitoring	Service	Posture checks, drift detection, vuln signals, compliance checks

Cost Governance	Service	Budgets, allocation rules, anomaly detection, optimization recommendations
Evidence Store	Data/Service	Control results, logs, attestations, snapshots, retention, access control
Reporting & Dashboards	Service	Security posture, compliance status, FinOps KPIs, operational SLOs
Exception/Waiver Workflow	Process/App	Approval, expiry, compensating controls, audit traceability

3.4 Interfaces & Contracts

3.4.1 Key Interfaces

- **Policy API:** publish/version policy sets; retrieve effective policies by scope (org/account/project).
- **Control Results API:** standard schema for checks (pass/fail, severity, resource identifiers, metadata).
- **Evidence API:** query evidence by control, timeframe, resource, or audit request.
- **Notification API:** route signals to SIEM/ticketing/on-call channels with enrichment.

3.4.2 Canonical Data Objects (Starter)

- `Policy`: id, scope, statement, parameters, owner, version, effective_date
- `Control`: id, objective, test_procedure, automation_level, frequency, evidence_type
- `ControlResult`: control_id, resource_id, status, severity, timestamp, evidence_ref
- `Exception`: policy/control reference, justification, approver, expiry, compensating_controls

3.5 Logical View: Variability

- **Single-cloud vs. multi-cloud:** provider adapters behind common policy/control abstractions.
- **Tooling choices:** multiple posture tools may exist; standardize output into ControlResult schema.
- **Org maturity:** begin with preventive guardrails + minimal evidence; evolve to continuous controls.

3.6 Logical View: Rationale

- Separating **policy intent** from **enforcement** enables policy iteration without re-architecting execution.
- Explicit **evidence store** turns audit readiness into a product, not an ad hoc activity.

Chapter 4

Process View

4.1 View Packet Summary

View type: Process / Runtime behavior
Primary stakeholders: Platform Engineering, Security Operations, FinOps, Product Teams
Primary concerns: Control lifecycle, event flows, concurrency, escalation paths, and operational procedures.

4.2 Key Runtime Scenarios (Primary Presentation)

4.2.1 Scenario A: Provisioning a New Workload (Golden Path)

1. Product team requests/initiates a workload via self-service catalog.
2. IaC pipeline applies approved modules; preventive policies validate configuration.
3. Landing zone baselines attach monitoring/logging and identity constraints.
4. Evidence artifacts emitted (deployment attestation, policy evaluation results).

4.2.2 Scenario B: Policy Violation Detection & Remediation

1. Continuous monitoring detects drift/misconfiguration (control check fails).
2. Alert enriched with ownership, severity, and remediation guidance.
3. Ticket/notification created; remediation runbook executed (manual or automated).
4. Evidence updated with resolution and timestamps; metrics captured for trend analysis.

4.2.3 Scenario C: Cost Anomaly & Optimization Loop

1. Cost governance detects anomaly (budget threshold, spike, or idle resources).
2. FinOps triage assigns action: rightsizing, shutdown, reservation/commitment planning.
3. Changes implemented via IaC; evidence and KPI dashboards updated.

4.3 Process Diagrams

Insert sequence/activity diagrams here (e.g., BPMN-like flow for "Define → Implement → Monitor → Optimize").

Placeholder: Governance Lifecycle Flow Diagram (PNG/SVG)

Figure 4.1: Process View: Continuous governance lifecycle and operational workflows.

4.4 Process View: Concurrency & Scaling Considerations

- Controls execute continuously across many accounts/projects; prioritize by risk (severity tiers).
- Evidence ingestion must be idempotent and support high-frequency signals (posture checks, log events).
- Notifications require rate limiting and deduplication to avoid alert fatigue.

4.5 Process View: Operational Policies

- **SLOs:** time-to-detect, time-to-remediate for high severity violations.
- **Change management:** policy changes are versioned and rolled out with staged enforcement.
- **Exception handling:** every exception has an owner, expiry, and compensating controls.

Chapter 5

Deployment View

5.1 View Packet Summary

View type: Deployment / Allocation
Primary stakeholders: Platform Engineering, Security, Operations
Primary concerns: Where modules run, network boundaries, identity boundaries, integrations, resilience, and operational ownership.

5.2 Deployment Diagram (Primary Presentation)

Insert your deployment diagram here. Recommended: governance plane services + provider environments + tooling integrations (CI/CD, SIEM, CMDB, ticketing).

Placeholder: Deployment Diagram (PNG/SVG)

Figure 5.1: Deployment View: Mapping of logical modules to runtime infrastructure and integrations.

5.3 Node/Environment Catalog

Node/Env	Type	Notes
Governance Control Plane	Runtime env	Hosts policy-as-code engine, orchestration, evidence ingestion
Cloud Provider Workload Planes	Runtime env	Accounts/subscriptions/projects with guardrails applied
CI/CD Platform	External system	Runs IaC pipelines, policy gates, attestations
Logging/SIEM	External system	Central collection, detection rules, alert routing
Evidence Store	Data platform	Immutable storage, retention, access controls, audit queries
Dashboards/Reporting	Service	Role-based access; exports for audit packages

5.4 Deployment View: Availability & Resilience

- Evidence store is a critical asset; enforce backups, retention controls, and least-privilege access.
- Control plane services should be horizontally scalable and support retry/backoff for provider APIs.
- Separate dev/test/prod governance environments to validate policy changes before enforcement.

5.5 Deployment View: Security Considerations

- Strong identity boundary for governance administrators (PIM/PAM, MFA, approvals).
- Encrypt evidence/log data at rest and in transit; audit all access to evidence.
- Ensure separation-of-duties for exception approval vs. implementation.

Chapter 6

Beyond Views

6.1 Beyond Views Overview

Purpose: Provide the "glue" information that makes the views usable: rationale, mappings, control catalog, evidence model, risks, glossary, and documentation practices.

6.2 Control Catalog (Template + Starter)

6.2.1 Control Model

Each control should have:

- **Control ID** (stable identifier)
- **Control Objective** (what it ensures)
- **Control Type** (preventive / detective / corrective)
- **Implementation Mechanism** (policy-as-code, IaC gate, monitoring check, process)
- **Frequency** (continuous, daily, per-deploy, quarterly)
- **Evidence Artifact(s)** (what proves it)
- **Owner** and **Escalation Path**

6.2.2 Starter Control Catalog

ID	Objective	Type	Mechanism	Evidence
IAM-01	Enforce least privilege via RBAC	Preventive	Policy baseline + role templates	Role assignment snapshots; approval logs
NET-01	Restrict public exposure by default	Preventive	Policy-as-code + IaC module	Policy eval results; deployment attestations
LOG-01	Centralize logs for all workloads	Preventive/Detective	Landing zone baseline	Log ingestion status; coverage report
CFG-01	Detect config drift from baselines	Detective	Continuous posture checks	ControlResult stream; drift reports
COST-01	Budget thresholds per cost center	Detective	Budget alerts/anomaly detection	Budget config; alert records; KPI trends

EXC-01	Time-bound exceptions with approvals	Corrective	Exception workflow	Exception record; expiry; compensating controls

6.3 Evidence Model (Evidence-as-a-Product)

6.3.1 Evidence Types

- **Automated evaluations:** policy results, posture scans, CI/CD gate outputs
- **Telemetry-derived:** logs, alerts, detections, incident records
- **Attestations:** approvals, exception waivers, quarterly access reviews
- **State snapshots:** inventory exports, configuration baselines, role assignment snapshots

6.3.2 Evidence Mapping Template

Control ID	Evidence Artifact	Retention	Access/Owner
IAM-01	RBAC assignment export (daily)	1–7 years	Security/IAM (read-only for Audit)
LOG-01	Log coverage report (weekly)	1 year	Platform Ops
CFG-01	Drift findings (continuous)	1 year	Platform Ops + Security

6.4 Rationale & Key Design Decisions

6.4.1 Architecture Decisions (ADR Index)

Maintain an ADR log for materially significant governance decisions:

- ADR-001: Policy-as-code approach and enforcement tiers (warn/block)
- ADR-002: Evidence store technology choice and retention strategy
- ADR-003: Exception workflow requirements (expiry, compensating controls)
- ADR-004: Canonical schema for ControlResult and evidence referencing

6.4.2 Design Principles

- **Automation-first:** policies and controls must be machine-enforceable where feasible.
- **Least privilege by default:** identity is the primary control plane.
- **Evidence is a product:** audit readiness is continuous, not event-driven.
- **Golden paths:** enablement reduces risk more effectively than "after-the-fact policing."

6.5 Mappings

6.5.1 Mapping Between Views

Mapping	Purpose
Context ↔ Logical	Trace external actors to the modules they interact with
Logical ↔ Process	Trace runtime scenarios to responsible modules
Logical ↔ Deployment	Trace modules to runtime nodes/environments and trust boundaries
Controls ↔ Evidence	Prove each control has objective, mechanism, and auditable artifacts

6.5.2 Standards/Framework Mapping (Template)

Control ID	Framework Ref	Notes
IAM-01	NIST / ISO / SOC2 (fill)	Map objective to requirement language; reference internal policy
LOG-01	NIST / ISO / SOC2 (fill)	Evidence demonstrates continuous logging coverage

6.6 Risks, Issues, & Exceptions

6.6.1 Risk Register (Template)

ID	Severity	Risk	Mitigation
R-01	High	Policy enforcement too strict blocks delivery	Start with warn/tiered enforcement; golden paths; exception workflow
R-02	High	Evidence store access becomes a data exposure risk	Least privilege; auditing; encryption; segregation of duties
R-03	Medium	Tool sprawl produces inconsistent control results	Canonical ControlResult schema; normalization layer

6.6.2 Exception Policy (Template)

- All exceptions require: justification, owner, approver, expiry date, compensating controls.
- Exceptions are reviewed periodically; expired exceptions auto-escalate.

6.7 Glossary

Term	Definition
Guardrail	Preventive constraint (policy baseline) that blocks or restricts unsafe configurations
Control	A mechanism that enforces or verifies a policy objective (prevent/detect/correct)
Evidence	Artifact demonstrating control operation (logs, snapshots, attestations, control results)
Landing Zone	Standardized cloud environment baseline (identity, network, logging, policies)
Policy-as-Code	Declarative encoding of policy intent into machine-evaluable rules
Golden Path	Approved deployment pattern that is secure and compliant by default

6.8 Documentation Packaging, Release, and Governance

6.8.1 Documentation Release Cadence

- Release this package versioned alongside policy sets (e.g., quarterly or per major baseline change).
- Each release includes: updated views, updated control catalog, updated evidence mapping, updated ADR index.

6.8.2 Confluence Page Tree Mapping (Suggested)

- **Space Home:** Cloud Governance & Control Framework
- **1. Overview & Strategy**
  - Architecture Overview (this chapter)
  - Stakeholders & Concerns
  - Principles & ADR Index
- **2. Architecture Views**
  - Context View
  - Logical View
  - Process View
  - Deployment View
- **3. Controls & Evidence**
  - Control Catalog
  - Evidence Model & Retention
  - Standards Mapping
  - Exception Policy
- **4. Operations**
  - Monitoring & Alerting
  - Runbooks & Playbooks
  - SLOs & Metrics
- **5. Risks & Roadmap**
  - Risk Register
  - Backlog / Improvement Roadmap

Appendix A

Appendix A: Diagram Source (Optional)

If you maintain diagram sources (PlantUML/Mermaid) alongside exported images, include them here for reproducibility.

A.1 PlantUML: High-Level Reference (From Earlier Draft)

```
@startuml
title Cloud Governance & Control Framework - Reference Architecture (High Level)
skinparam componentStyle rectangle

package "Strategy & Policy (Intent)" {
  [Governance Policies\n(Security, Identity, Cost, Ops, Compliance)]
  [Standards & Control Mapping\n(NIST/ISO/SOC2/etc.)]
  [Risk Appetite & Exceptions\n(waivers, approvals, expiry)]
}

package "Guardrails & Control Plane (Prevent/Detect)" {
  [Landing Zones & Org Structure]
  [IAM/RBAC Guardrails\n(least privilege, PIM)]
  [Policy-as-Code\n(tagging, encryption, regions)]
  [Config/Posture Monitoring\n(drift, misconfig)]
  [Central Logging/SIEM Feed]
  [Cost Controls\n(budgets, alerts, allocation)]
}

package "Execution (Build/Run)" {
  [Self-Service Catalog]
  [IaC Pipelines\n(modules, reviews, gates)]
  [Change/Release Mgmt]
  [Incident & Vulnerability Mgmt]
  [Asset/CMDB Inventory]
}

package "Evidence & Reporting (Prove/Improve)" {
  [Dashboards\n(Security/Compliance/FinOps/Ops)]
  [Automated Evidence Store]
```

```
  [Audit Reporting]
  [Optimization Backlog\n(remediation + improvements)]
}

[Governance Policies\n(Security, Identity, Cost, Ops, Compliance)] -->
↪  [Policy-as-Code\n(tagging, encryption, regions)]
[Standards & Control Mapping\n(NIST/ISO/SOC2/etc.)] --> [Automated Evidence Store]
[Landing Zones & Org Structure] --> [Self-Service Catalog]
[IaC Pipelines\n(modules, reviews, gates)] --> [Config/Posture Monitoring\n(drift,
↪  misconfig)]
[Central Logging/SIEM Feed] --> [Dashboards\n(Security/Compliance/FinOps/Ops)]
[Cost Controls\n(budgets, alerts, allocation)] -->
↪  [Dashboards\n(Security/Compliance/FinOps/Ops)]
[Automated Evidence Store] --> [Audit Reporting]
[Dashboards\n(Security/Compliance/FinOps/Ops)] --> [Optimization Backlog\n(remediation +
↪  improvements)]
@enduml
```

Appendix B

Appendix B: View Packet Checklist (Template)

For each view, ensure the packet includes:

- View packet summary (stakeholders, concerns, scope)
- Primary presentation (diagram)
- Element catalog (tables of elements and responsibilities)
- Interfaces/assumptions/constraints relevant to the view
- Variability (what can change, extension points)
- Rationale (why structured this way)
- Cross-references to other views and to controls/evidence