Defensible10

ISAUnited
INSTITUTE OF SECURITY
ARCHITECTURE UNITED

Defensible 10

Annex B (Normative): D02-Cloud Security Architecture & Resilience

Technical Standard

Standards Committee
11-17-2025

About ISAUnited

The Institute of Security Architecture United is the first dedicated Standards Development Organization (SDO) focused exclusively on cybersecurity architecture and engineering through security-by-design. As an international support institute, ISAUnited helps individuals and enterprises unlock the full potential of technology by promoting best practices and fostering innovation in security.

Technology drives progress; security enables it. ISAUnited equips practitioners and organizations across cybersecurity, IT operations, cloud/platform engineering, software development, data/AI, and product/operations with vendor-agnostic standards, education, credentials, and a peer community—turning good practice into engineered, testable outcomes in real environments.

Headquartered in the United States, ISAUnited is committed to promoting a global presence and delivering programs that emphasize collaboration, clarity, and actionable solutions to today's and tomorrow's security challenges. With a focus on security by design, the institute champions integrating security into every stage of architectural and engineering practices, ensuring robust, resilient, and defensible systems for organizations worldwide.

Disclaimer

ISAUnited publishes the ISAUnited Defensible 10 Standards Technical Guide to provide informational and educational content regarding security architecture and engineering practices. While efforts have been made to ensure accuracy and reliability, the content is provided "as is," without any express or implied warranties. This guide is for informational purposes only and does not constitute legal, regulatory, compliance, or professional advice. Consult qualified professionals before making decisions.

Limitation of Liability

ISAUnited - and its authors, contributors, and affiliates - shall not be liable for any direct, indirect, incidental, consequential, special, exemplary, or punitive damages arising from the use of, inability to use, or reliance on this guide, including any errors or omissions.

Operational Safety Notice

Implementing security controls can affect system behavior and availability. First, validate changes in non-production, use change control, and ensure rollback plans are in place.

Third-Party References

This guide may reference third-party frameworks, websites, or resources. ISAUnited does not endorse and is not responsible for the content, products, or services of third parties. Access is at the reader's own risk.

Use of Normative Terms ("Shall," "Should," "Must")

- Must / Shall: A mandatory requirement for conformance to the standard.
- Must Not / Shall Not: A prohibition; implementations claiming conformance shall not perform the stated action.
- Should: A strong recommendation; valid reasons may exist to deviate in particular circumstances, but the full implications must be understood and documented.

Acceptance of Terms

By using this guide, readers acknowledge and agree to the terms in this disclaimer. If you disagree, refrain from using the information provided.

For more information, please visit our Terms and Conditions page.

License & Use Permissions

Standards Development & Governance Notice

This standard is one of the ten Parent Standards in the Defensible 10 Standards (D10S) series. Each Parent Standard is governed by ISAUnited's Standards Committee, peer-reviewed by the ISAUnited Technical Fellow Society, and maintained in the Defensible 10 Standards GitHub repository for transparency and version control.

Contributions & Collaboration

ISAUnited maintains a public GitHub repository for standards development. Practitioners may view and clone materials, but contributions require:

- ISAUnited registration and vetting
- Approved Contributor ID
- Valid GitHub username

All Sub-Standard contributions must follow the Defensible Standards Submission Schema (D-SSF) and are peer-reviewed by the Technical Fellow Society during the annual Open Season.

Abstract

The ISAUnited Defensible 10 Standards provide a structured, engineering-grade framework for implementing robust and measurable cybersecurity architecture and engineering practices. The guide outlines the frameworks, principles, methods, and technical specifications necessary for designing, building, verifying, and operating reliable systems.

Developed under the ISAUnited methodology, the standards align with modern enterprise realities, integrating Security by Design, continuous technical validation, and resilience-based engineering to address emerging threats. The guide is written for security architects and engineers, IT and platform practitioners, software and product teams, governance and risk professionals, and technical decision-makers seeking a defensible approach that is testable, auditable, and scalable.

This document includes a series of Practitioner Guidance, Cybersecurity Students & Early-Career Guidance, and Quick Win Playbook callouts.	
	Practitioner Guidance- Actionable steps and patterns to apply the technical standards in real environments.
	Cybersecurity Student & Early-Career Guidance- Compact, hands-on activities that turn each section's ideas into a small, verifiable artifact.
	Quick Win Playbook- Immediate, evidence-driven actions that improve posture now while reinforcing good engineering discipline.

Together, these elements help organizations translate intent into engineered outcomes and sustain long-term protection and operational integrity.

Foreword

Message from ISAUnited Leadership

Cybersecurity is at a turning point. As digital systems scale, reactive and checklist-driven practices do not keep pace with adversaries. The ISAUnited position is clear: security must be practiced as engineered design, grounded in scientific principles, structured methods, and defensible evidence. Our mission is to professionalize cybersecurity architecture and engineering with standards that are actionable, testable, and auditable.

ISAUnited Defensible 10 Standards: First Edition is a practical framework for that shift. The standards in this book are not theoretical. They translate intent into measurable specifications, controls, and verification, and enable teams to design and operate resilient systems at enterprise scale.

About This First Edition

This edition publishes ten Parent Standards, one for each of the core domains of security architecture and engineering. Sub-standards will follow in subsequent editions, contributed by ISAUnited members and reviewed by our Technical Fellow Society, to add focused and technology-aligned detail. Adopting the Parent Standards now positions organizations for seamless integration of Sub Standards as they are released on the ISAUnited annual update cycle.

Why “Defensible Standards”

Defensible means the work can withstand technical, operational, and adversarial scrutiny. These standards are designed to be demonstrated with evidence, featuring clear architecture, measurable specifications, and verification, so that practitioners can confidently stand behind their designs.

Contents

Annex B (Normative): D02-Cloud Security Architecture & Resilience

Section 1. Standard Introduction

Cloud computing has transformed how organizations design, deploy, and operate IT systems, delivering scalability, flexibility, and efficiency beyond traditional models. This shift introduces security challenges that require disciplined, engineering-based approaches. This Parent Standard establishes an authoritative foundation for designing and maintaining secure, resilient cloud architectures. It is written for cybersecurity engineers, architects, and technical leaders who must implement measurable, defensible security strategies across multi-cloud, hybrid, and cloud-native environments while preserving interoperability, compliance, and operational effectiveness.

Objective

Define foundational principles for Cloud Security Architecture & Resilience that guide practitioners toward a structured, disciplined approach to securing cloud environments. The standard provides clear, methodical guidance to safeguard data, protect identities, and ensure resilient operation of cloud services, with emphasis on measurable outcomes, disciplined implementation, and practical defensibility.

Justification

As organizations adopt cloud services to drive digital transformation, they encounter complexities that foundational cybersecurity frameworks—for example, NIST and ISO—do not fully address at implementation depth. While such frameworks provide essential baselines and compliance guidance, they often lack the technical specificity necessary for precise implementation and validation of controls within complex cloud infrastructures.

Cloud environments differ from traditional IT through distributed architectures, dynamic workloads, and shared security responsibilities between providers and customers. These characteristics increase the likelihood of misconfiguration, data exposure, and unauthorized access. Industry experience and breach analyses demonstrate that conventional, checklist-driven methods are insufficient to consistently secure cloud deployments, leading to avoidable vulnerabilities and operational disruptions.
This Parent Standard closes that gap by emphasizing security-by-design, resilience, and measurable security outcomes. It provides explicit, implementation-oriented guidance that enables architects and engineers to construct cloud environments that are robust, verifiably secure, and operationally resilient against evolving threats.

By adopting this standard, organizations and academic programs equip practitioners with transparent, structured methodologies that proactively manage cloud risk, prevent breaches, and maintain continuous resilience. Subsequent sections specify the Requirements (Inputs), Technical Specifications (Outputs), Core Principles, Security Controls, Engineering Discipline, and Verification & Validation (V&V) methods to ensure implementations are auditable and defensible end-to-end.

Section 2. Definitions

Application Programming Interface (API) – Programmatic interface enabling services to communicate; requires authentication, authorization, rate limiting, schema validation, and logging.

Architecture Decision Record (ADR) – A Lightweight record of a significant design decision, including options, constraints, and assumptions, trade-offs, the chosen option, resulting invariants, and the test and evidence plan.

Artifact Signature and Attestation – Cryptographic verification of build provenance and integrity for images and packages; enforced before start and during scale events.

Attribute-Based Access Control (ABAC) – An Authorization model that evaluates attributes (user, resource, action, context) to make fine-grained, condition-aware access decisions.

Bastion (bastion host) – Controlled administrative entry point with strong authentication, mTLS where applicable, and full session recording.

Breach and Attack Simulation (BAS) – Automated, continuous validation of controls by emulating adversary techniques.

Cloud Encryption – Conversion of data to an unreadable form to prevent unauthorized access. Defaults in this Parent Standard: AES-256 for data at rest; TLS 1.3 for data in transit.

Cloud-Native Application Protection Platform (CNAPP) – Unified, tightly integrated capability bundle that protects cloud-native infrastructure and applications across build and run (for example, artifact scanning, configuration and compliance management, risk detection, behavioral analytics). Gartner

Cloud-Native Security – Security principles and controls designed for microservices, containers, serverless functions, and automated CI/CD pipelines.

Cloud Security Posture Management (CSPM) – Continuous evaluation of cloud resources for misconfiguration and drift with policy enforcement, auto-remediation, and promotion gates; applies common frameworks, regulatory requirements, and enterprise policies to assess and remediate configuration risk. Gartner

Cloud Workload Protection Platform (CWPP) – Runtime protection capability for VMs, containers, and serverless that detects and responds to anomalous process, network, and file behaviors across hybrid and public cloud environments. Gartner

Configuration as Code (CaC) – Management of system and service configurations as version-controlled code to enable repeatable, auditable deployments.

Continuous Integration / Continuous Delivery (CI/CD) – Automated build, test, and deployment pipelines that promote verified artifacts through environments under version control and policy gates.

Data Classification and Tagging – Categorization of data by sensitivity and application of metadata labels to drive protection, lifecycle, ownership, budget, and evidence routing.

Data-Flow Protection – Controls that protect data in motion and at use, including masking or anonymization of non-production datasets, tokenization where appropriate, and DLP on designated egress paths.

Data Loss Prevention (DLP) – Inspection and policy enforcement that detect and block unauthorized transfer of sensitive data at defined egress points.

Distributed Denial of Service (DDoS) – Volumetric or application-layer attack that exhausts resources to deny service; mitigated with rate controls, filtering, and scrubbing.

Drift – Divergence between approved, version-controlled configurations and the actual runtime state; must be detected and reconciled.

Egress Allowlist – Explicit set of permitted destinations for outbound traffic from a zone or workload; all other destinations are denied by default.

Evidence Pack (EP) – Structured bundle of artifacts (plans, policies, configs, logs, test results, decisions) that substantiates compliance with §5 Requirements, §6 Outputs, and §12 V&V.

Evidence Pack Convention (D02 – Cloud) – All artifacts for this Parent Standard SHALL be filed under EP-02 with child packs EP-02.1, EP-02.2, EP-02.3, … as referenced in §§6, 12, and 13.

Extended Detection and Response (XDR) – Integrated detection and response across endpoints, identities, networks, and cloud workloads to accelerate containment.

Failure Modes and Effects Analysis (FMEA) – Structured method to identify failure modes, effects, severity, and mitigations for critical paths; used to derive §12 tests.

Fault Tree Analysis (FTA) – Top-down hazard analysis that decomposes an undesired event into combinations of faults; used to inform safeguards and test design.

Hub-and-Spoke Segmentation – Network topology where a centralized hub enforces shared controls and spokes isolate workloads by purpose or tenant; validated with positive and negative contract tests.

Identity and Access Management (IAM) – Discipline ensuring the right individuals and services have appropriate access to cloud resources; governs authentication, authorization, and account lifecycle aligned to least privilege.

Infrastructure as Code (IaC) – Management and provisioning of cloud resources through machine-readable definitions stored in version control.

Interface Control Document (ICD) – Artifact that defines an interface's contract, including protocols, ports, identities, rate and size limits, error handling, telemetry, and security invariants.

JSON Web Token (JWT) – Compact, signed token format for conveying claims used in API authentication and authorization.

Just-in-Time (JIT) Privileged Access – Time-bound elevation of privileges with approval and session monitoring to eliminate standing administrative access.

Key Management System (KMS) – Centralized service for creating, storing, rotating, and controlling access to cryptographic keys with full auditability.

Landing Zone (LZ) – Pre-configured cloud baseline of identity, network, logging and telemetry, tagging, and policy guardrails that all workloads inherit; verified before first workload promotion.

Mean Time to Contain (MTTC) – Average time from detection to containment of an incident; a key SLO for response efficacy.

Mean Time to Detect (MTTD) – Average time from incident onset to reliable detection; a key SLO for detection efficacy.

Micro-Segmentation – Fine-grained isolation of workloads and services to restrict east–west movement based on identity and context.

Mutable Tag – Tag that can point to different artifact digests over time (for example, "latest"); prohibited for production.

Mutual TLS (mTLS) – Certificate-based mutual authentication between services to protect east–west and administrative paths.

Noisy-Neighbor Effect – Resource contention in multi-tenant environments where one tenant's load degrades another's performance; mitigated by quotas, runtime limits, and isolation.

OAuth 2.0 – Authorization framework that enables delegated access to APIs without sharing credentials.

OpenID Connect (OIDC) – Identity layer on top of OAuth 2.0 that provides authentication and user identity assertions.

Policy as Code (PaC) – Expression and validation of security and compliance policies as code, enforced automatically in pipelines and at runtime.

Private Endpoint – Private, provider-managed interface that exposes a service over internal networking only, eliminating public exposure.

Promotion Gate (fail-closed) – Non-bypassable CI/CD control that blocks environment promotion until specified policy checks and tests pass.

Public Key Infrastructure (PKI) – Mechanisms and services to issue, rotate, and revoke certificates used for mTLS and other cryptographic functions.

Role-Based Access Control (RBAC) – An Authorization model that grants permissions based on job roles to reduce privilege sprawl and simplify review.

Runtime Security (VMs/Containers/Serverless) – Controls that validate workload integrity and behavior during execution, including anomaly detection and allow/block actions.

Security Group / Access Control List (ACL) – Native filtering controls that allow or deny network flows at workload or subnet boundaries.

Security Information and Event Management (SIEM) – A Platform that aggregates and correlates logs and telemetry for detection, investigation, and reporting.

Service Level Objective (SLO) – Target level of performance or outcome for a service or control (for example, MTTD, MTTC, restore-time objectives).

Software-Defined Perimeter (SDP) – An Access model that hides services from unauthenticated entities and grants access dynamically after verification.

Unified Logging Schema – Standardized event fields (for example, timestamp, actor, action, resource, result, trace_id, control_id, env) used to normalize telemetry across providers and services.

Verification & Validation (V&V) – Structured activities that confirm build correctness against specifications (verification) and operational effectiveness under realistic conditions (validation).

Virtual Network (VPC/VNet) – Provider logical network construct used to segment and route cloud workloads (subnets, route policies, NAT, and related controls).

Web Application Firewall (WAF) – Layer-7 control that inspects and filters HTTP(S) requests to protect web applications and APIs from common attacks.

Workload Identity – Cryptographically verifiable identity assigned to a service, application, or function, used to obtain least-privilege access to other services and data.

Zero Trust Cloud Architecture (ZTCA) – Cloud security approach that assumes no implicit trust; continuously verifies identity, device, and context, and enforces least-privilege, segmented access.

Section 3. Scope

Cloud computing introduces dynamic, distributed environments that demand clear architectural boundaries and defensible engineering practices. This Parent Standard applies to all major cloud service models, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), and to organizations operating in public, private, hybrid, or multi-cloud deployments. It defines architectural expectations and technical guardrails needed to achieve measurable resilience, helping practitioners anticipate misconfigurations, enforce access boundaries, and mitigate evolving threats while adopting cloud-native capabilities.

Applicability

- **All cloud service models:** IaaS, PaaS, and SaaS across diverse architectures.
- **Deployment models:** Public, private, hybrid, and multi-cloud environments.
- **Enterprise and academic settings:** Technical teams advancing engineering-grade cloud security practices.

Key Focus Areas

- **Identity and access security controls:** Workload and human identities, least privilege, and Zero Trust enforcement across platforms.

- **Network segmentation and isolation:** Landing zone baselines, hub-and-spoke topology, private endpoints, micro-segmentation, and mTLS for east–west paths.
- **Cloud-native security models:** Policy-as-code and runtime controls for containers and serverless, including verify-before-start and artifact integrity.
- **Data protection and API security:** Strong cryptography, lifecycle and data-flow protections, DLP on designated egress paths, and resilient API gateways.
- **Continuous monitoring and incident response:** Unified telemetry and CSPM-driven gates, automated detection and response, and evidence suitable for Verification & Validation.
- **Elasticity and multi-tenant controls:** Quotas and runtime limits, scale-safe admission checks, and usage/egress telemetry for anomaly detection.

Outcomes

By defining this scope, the standard makes a cloud security architecture:

- **Defensible:** Based on clear, enforceable boundaries and technical controls.
- **Measurable:** Validated through assessment, testing, and repeatable evidence.
- **Adaptive:** Capable of evolving with new technologies and threat conditions.
- **Aligned:** Consistent with organizational, regulatory, and industry requirements.

Cloud characteristics, including self-service, network-accessible services, pooled resources, elastic scale, and metered usage, are assumed. The Technical Specifications in §6 convert those characteristics into enforceable behaviors, and §12 Verification & Validation confirms that the behaviors hold under load, change, and adversarial conditions.

Section 4. Use Case

A robust cloud security standard must demonstrate practical applicability in complex, real-world environments. The following consolidated use case presents a technical scenario typical of modern enterprises operating in multi-cloud environments. It highlights common architectural weaknesses, maps them to targeted technical solutions based on Zero Trust Cloud Architecture (ZTCA), and defines measurable outcomes. This integrated approach enables technical teams to align engineering actions with defensible security objectives.

Table B-1. Consolidated Use Case

Use Case Name	**Securing Multi-Cloud Infrastructure with Zero Trust Cloud Architecture**
Objective	

	Implement ZTCA to protect multi-cloud environments from unauthorized access, misconfiguration, and lateral-movement attacks.
Scenario	A global enterprise operates workloads across AWS, Azure, and Google Cloud. Teams provision resources through self-service. Inconsistent access-control policies, frequent misconfigurations, and visibility gaps persist. Security investigations have observed privilege-escalation attempts and unmonitored lateral movement across cloud workloads.
Actors	Cloud Security Architect; Cloud Engineer; Identity and Access Management (IAM) Team; DevOps Team; Security Operations Center (SOC) Analysts; Platform Engineering Team
Challenges Identified	• Missing landing-zone guardrails: Identity, network, logging, tagging, and policy baselines not verified before first workload promotion. • Unrestricted cloud access: Over-privileged accounts and misconfigured IAM policies enable broad access to workloads. • Cloud misconfigurations: Inconsistent policies across providers expose public storage and insecure network rules. • Flat network structures: Excessive east–west movement due to weak segmentation. • Limited visibility: Fragmented logging and absent central correlation. • Unverified artifacts at deploy/scale: Images and functions start without signature or attestation checks. • Internet edge exposure: No generic layer-7 filtering or DDoS rate controls on public endpoints.
Technical Solution	• Identity and access hardening: Enforce RBAC/ABAC and Just-in-Time privileged access; remove standing admin; conduct periodic access reviews. • CSPM gates: Continuously detect and auto-remediate critical misconfigurations; integrate posture checks into IaC/PaC pipelines as non-bypassable gates. • Network segmentation and Zero Trust controls: Establish a verified landing zone; apply hub-and-spoke segmentation with private endpoints and micro-segmentation; require mTLS for east–west and administrative paths. • Artifact integrity at run and scale: Enforce verify-before-start and verify-at-scale for image signatures and attestations; deny mutable tags and unapproved registries or namespaces. • Edge protections and telemetry: Apply layer-7 WAF policies and DDoS rate controls for Internet-exposed services; centralize logs and metrics in a SIEM/XDR for correlation and alerting.
Expected Outcome (Target KPIs; calibrate to baseline)	• Reduce over-privileged access and shrink the effective attack surface by approximately 60 %. • Cut public-exposure and insecure-policy findings by approximately 75 % through posture gates and auto-remediation. • Block unauthorized lateral movement across workloads via enforced segmentation and mTLS. • Achieve 100 % verify-before-start for production artifacts and deny unapproved registries/tags.

	• Improve mean time to detect and contain by approximately 40 % through centralized visibility and automated playbooks.

This consolidated chart provides an explicit, actionable reference for technical teams, enabling direct mapping of identified risks to engineering solutions and measurable improvements in the cloud security posture.

Section 5. Requirements (Inputs)

A defensible cloud security architecture is grounded in clearly defined, actionable inputs. These inputs establish the technical, procedural, and policy conditions that SHALL be present before implementation begins. By setting these preconditions, ISAUnited's Defensible Standards ensure organizations are prepared for disciplined, engineering-driven integration—moving beyond generic guidance to enforceable readiness criteria.

5.1 Zero Trust Cloud Security

- All access requests—user, workload, or service—SHALL be continuously verified before permission is granted.
- Mechanisms for ongoing authentication and dynamic authorization SHALL be implemented; no implicit trust is permitted.
- Multi-factor authentication (MFA) SHALL be enforced for all privileged actions; least privilege SHALL be the default posture.

5.2 Shared Responsibility Model Alignment

- Security controls SHALL be explicitly aligned with each provider's published shared responsibility model.
- The delineation of provider-managed versus organization-managed functions SHALL be documented and reviewed at defined intervals.
- Organizations SHALL monitor and validate the effectiveness of both provider and organization-managed controls to prevent misconfigured or unmonitored boundaries.

5.3 Automated Security Enforcement

- Automation such as Cloud Security Posture Management (CSPM), Infrastructure as Code (IaC), and Policy as Code (PaC) SHALL enforce consistent security across environments.
- Detection and remediation of misconfigurations, policy violations, and gaps SHALL be automated.
- Security validation workflows SHALL be integrated into CI/CD pipelines; promotion SHALL fail closed on critical findings.
- Posture findings from CSPM SHALL operate as non-bypassable promotion gates; critical severities SHALL auto-remediate or block promotion with recorded evidence.

- Where a CNAPP is used, it SHALL provide the same gating behavior and evidence outcomes defined in §6 and §12.

5.4 Cloud Network Segmentation

- Logical network segmentation using provider-agnostic virtual network constructs (for example, virtual networks/VPCs, subnets, overlays) SHALL isolate workloads, control planes, and services.
- Isolation boundaries SHALL be enforced to prevent lateral movement and contain potential breaches.
- Micro-segmentation and software-defined perimeters (SDPs) SHALL restrict intra-cloud traffic based on identity and context.

5.5 Cloud Data Encryption & Compliance

- Encryption SHALL be enabled by default for all data at rest (AES-256) and in transit (TLS 1.3).
- Encryption and data handling practices SHALL align with applicable residency, privacy, and compliance requirements.
- Centralized key management (KMS) SHALL be implemented; cryptographic keys SHALL rotate on a defined cadence via automated policies.

5.6 Landing Zone Baseline (readiness)

- A Landing Zone with identity, network, logging, and telemetry, tagging, and baseline policies SHALL be verified before first workload promotion; promotion SHALL fail closed if any guardrail is missing.

5.7 Unified Telemetry and Evidence Readiness

- A unified logging schema, authenticated time synchronization, and a designated Evidence Pack repository with ID conventions SHALL be established prior to deployment; controls SHALL emit logs and metrics to this destination.

5.8 Artifact Integrity and Approved Sources

- Policy for image/signature/attestation verification, approved registries/namespaces, and prohibition of mutable tags SHALL be defined and enforced in CI/CD and at admission prior to runtime.

5.9 Interface Contracts and Private Endpoints

- Traffic contracts (protocols, ports, identities, rate limits) for exposed interfaces SHALL be documented; designated managed services SHALL use private endpoints where feasible; any public exposure SHALL require a time-bounded exception with owner and expiry.

5.10 Multi-Tenant Isolation and Quotas

- Compute, storage, and network quotas and runtime limits per tenant or team SHALL be defined; verification tests SHALL confirm isolation and prevent noisy-neighbor effects.

By rigorously assessing and establishing these foundational inputs, organizations create the necessary conditions for a secure, resilient, and defensible cloud architecture. This approach supports consistent engineering practices, reduces the risk of misconfiguration, and embeds security from the outset of any cloud initiative.

Practitioner Guidance:

Use these requirements as readiness gates before implementing §6 and scheduling tests in §12.

- Map each §5 item to exactly one evidence artifact and one verification/validation test in §12.
- Maintain single sources of truth (one diagram set, one policy set, one repository) to minimize drift.
- Assign clear ownership per item (who approves, who maintains, who audits).
- Record baselines (privilege counts, segmentation maps, encryption coverage, CSPM posture) so §12 improvements are measurable.
- Fail closed in pipelines: block deployments when §5 gates are not met (for example, missing MFA, absent segmentation policies, or unenforced PaC checks).
- Review readiness gates after major architectural changes, and at least quarterly, to keep inputs current and defensible.

Section 6. Technical Specifications (Outputs)

Technical specifications define the concrete, defensible outputs that must be implemented to satisfy this standard. Each output is a required engineering area that turns policy into measurable, actionable security outcomes. Together, these specifications establish a robust, resilient foundation for cloud-native and hybrid enterprise environments.

Outputs must be:

- **Measurable:** validated by scans, logs, audits, or tests
- **Actionable:** implementation-ready, not policy slogans
- **Aligned:** traceable to §5 Requirements and sub-standards

6.1 Identity & Access Security in Cloud

- Enforce multi-factor authentication (MFA) for all privileged and administrative accounts.
- Apply RBAC and/or ABAC so every identity is constrained to least privilege.
- Use cloud-native identity and access services to manage identities, policies, and authorization across resources.
- Define workload identities with least-privilege roles per service/function; prohibit shared long-lived credentials; use Just-in-Time (JIT) elevation with session monitoring for interactive administration.
- Perform periodic access reviews and automatically remove orphaned or excessive privileges.

6.2 Cloud Network Security & Segmentation

- Implement logical network segmentation using provider-agnostic virtual network constructs (for example, virtual networks/VPCs, subnets, overlays) to isolate workloads, control planes, and services.
- Verify a Landing Zone baseline (identity, network, logging and telemetry, tagging, baseline policies) before first workload promotion; fail closed on missing guardrails.
- Use a hub-and-spoke topology for centralized boundary control and spoke isolation; execute positive and negative ingress/egress contract tests per spoke and record results.
- Use private endpoints and software-defined perimeters (SDPs) to secure intra-cloud communications and reduce lateral movement.
- Enforce default-deny ingress and egress at trust boundaries with layer-7 filtering and inspection; apply generic layer-7 WAF policies and DDoS rate controls for Internet-exposed services.
- Implement micro-segmentation to restrict east–west traffic based on identity, context, and application sensitivity; require mTLS for service-to-service and administrative paths.
- Define and enforce network access controls (security groups, ACLs, route policies) for least-privilege ingress and egress per workload; enforce per-tenant quotas and runtime limits to prevent noisy-neighbor effects.

6.3 Cloud Data Protection & Encryption

- Enable encryption by default: AES-256 for data at rest; TLS 1.3 for data in transit (internal and external).
- Use a centralized Key Management System (KMS) with automated rotation, least-privilege key access, and audit logging.
- Apply automated data classification, tagging, and lifecycle policies aligned to data sensitivity; validate lifecycle transitions in CI/CD.
- Use advanced cryptography (for example, homomorphic encryption) where processing requirements and data sensitivity justify it.
- Apply data-flow protections: mask or anonymize non-production datasets, and enforce DLP on designated egress paths.

6.4 Cloud API & Workload Security

- Use API gateways to enforce authentication, authorization, rate limiting, schema validation, and comprehensive logging for internal and external APIs.
- Use OAuth 2.0 / OpenID Connect with JWT-based tokens for standards-based API authentication and authorization.
- Implement runtime security for VMs, containers, and serverless, including image signing and scanning, integrity checks, and least-privilege execution; detect and act on anomalous process, network, and file behaviors with defined allow/block criteria.

- Verify artifact integrity before start and during scale events (signature/attestation); deny mutable tags and unapproved registries or namespaces.
- For serverless/functions, enforce per-function least-privilege roles, event-source allowlists, and explicit time, memory, and concurrency caps.
- Store secrets (keys, credentials, tokens) in dedicated vaults with automated rotation and access controls; inject secrets at runtime only; do not embed secrets in code or images.
- Require mTLS for service-to-service and administrative paths where applicable.
- Runtime behavior enforcement (CWPP-class). Workload controls SHALL detect and act on anomalous process, network, and file behaviors with defined allow/block criteria; results SHALL be captured in the Evidence Pack.
- Artifact verification at admission and scale. Signature/attestation verification SHALL run before start and during scale events; mutable tags and unapproved registries/namespaces SHALL be denied.

6.5 Cloud Security Monitoring & Incident Response

- Use a centralized security event platform (for example, SIEM/XDR) to collect, correlate, and analyze logs and telemetry from all cloud providers and services; adopt a unified logging schema with authenticated time synchronization.
- Use rules-based and/or machine-learning anomaly detection to identify novel threats and suspicious behavior in near real time; include usage and egress metrics (compute, storage, requests, bytes) for anomaly detection.
- Implement automated response playbooks to contain, eradicate, and recover from incidents with defined service-level objectives; correlate scale events with admission and policy outcomes.
- Ensure audit trails are immutable, tamper-evident, and retained according to policy and regulatory requirements.
- Automate continuous posture assessment (CSPM) to detect configuration drift, control gaps, and deviations; treat critical findings as non-bypassable promotion gates with auto-remediation or block-and-fix workflows.
- Conduct periodic disaster-recovery and failover exercises with objective SLOs; store plans, results, corrective actions, and re-test evidence.
- Posture gate telemetry. CSPM decisions (allow, auto-remediate, block) SHALL be logged to the unified schema and linked to Evidence Pack entries; if a CNAPP is used, it SHALL emit the same telemetry and evidence.

By adhering to these specifications, organizations can achieve measurable, defensible improvements in their cloud security posture and resilience.

Practitioner Guidance:

To ensure the successful implementation of these technical specifications:

- Baseline assessment: Begin with a comprehensive assessment of current posture, identifying gaps against each output area.
- Integration with DevOps: Embed controls and automation into CI/CD pipelines; include verify-before-start checks, contract tests for exposed paths, and CSPM gates.
- Continuous improvement: Review and update controls in response to evolving threats, cloud-service changes, and incident lessons learned; exercise DR/BCP playbooks on a defined cadence.
- Cross-functional collaboration: Coordinate among security, cloud engineering, DevOps, and compliance so controls are practical, scalable, and aligned with service objectives.
- Documentation and training: Maintain clear documentation of implemented controls and provide ongoing training on secure cloud architecture practices; record outcomes in Evidence Packs.

Quick Win Playbook:

Title: API Gateway Authentication Gate for an Internet-Exposed API (§6.4)

Objective: Establish standards-based authentication and rate control on one Internet-exposed API, validate allow/deny behavior in stage, and produce evidence artifacts for V&V.

Target: Gate one Internet-exposed API behind the API gateway with standards-based authentication (§6.4).

Component/System: API gateway + identity platform + secrets vault + workload service.

Protects: External API surface from unauthorized access and credential abuse; reduces abuse paths and token replay risk.

Stops/Detects: Missing/invalid tokens, weak authentication flows, unthrottled requests, secrets embedded in code/images.

Action: Require OAuth 2.0 / OpenID Connect with JWT; apply rate limiting; move secrets to a vault; deploy in log-only/stage mode; run a smoke test with 1) valid JWT (allow) and 2) no/invalid JWT (deny); ensure allow/deny events export to centralized telemetry; record owner and review cadence.

Proof: Gateway policy export + JWT validation sample + vault access log + gateway access logs (last 15 minutes); attach to Evidence Pack ID <EP-02.5>.

	Metric: 100 % of requests present a valid JWT; unauthorized requests are blocked; no secrets present in code/images; events appear in SIEM/XDR within target detection time. **Rollback**: Revert to the previous gateway policy version in the repository and redeploy the prior commit; archive new artifacts under <EP-02.5> as superseded.

Section 7. Cybersecurity Core Principles

The following ISAUnited Cybersecurity Core Principles are foundational to the design, implementation, and ongoing management of a secure cloud security architecture. Each principle guides architectural decisions, technical controls, and operational practices to ensure cloud environments are resilient, measurable, and engineered to withstand real-world threats.

Table B-2:

Principle Name	Code	Applicability to Cloud Security Architecture & Resilience
Least Privilege	ISAU-RP-01	Cloud IAM policies, roles, and permissions grant only the minimum necessary access; periodic reviews remove excess privileges.
Zero Trust	ISAU-RP-02	No implicit trust; every request (user, service, workload) is authenticated and authorized with context before access.
Defense in Depth	ISAU-RP-04	Layered controls (IAM, micro-segmentation, encryption, monitoring) reduce single points of failure in cloud deployments.
Secure by Design	ISAU-RP-05	Security requirements and controls are embedded from initial cloud design through deployment and operations.
Minimize Attack Surface	ISAU-RP-06	Reduce exposed services and endpoints by segmenting, using private endpoints, and implementing restrictive API policies.
Secure Defaults	ISAU-RP-10	Cloud services, resources, and pipelines are configured securely; deviations require explicit, reviewed change.

Principle Name	Code	Applicability to Cloud Security Architecture & Resilience
Resilience & Recovery	ISAU-RP-14	Architect for failure: redundancy, recovery plans, and tested restore procedures maintain operations during incidents.
Evidence Production	ISAU-RP-15	Centralized, immutable logging and audit trails enable defensibility, detection, and forensic analysis.
Protect Confidentiality	ISAU-RP-18	End-to-end protection using encryption (at rest/in transit) and strict key management (KMS) with least-privilege access.
Protect Availability	ISAU-RP-20	High-availability designs, autoscaling, and protection against disruption sustain service continuity.

Practitioner Guidance:

Embed these principles as design defaults—do not treat them as optional.

- Map each principle to at least one §6 output and one §12 test.
- Prefer RP-10 (Secure Defaults) and RP-15 (Evidence Production) to strengthen enforceability and auditability.
- Record principle-to-control traceability in your evidence pack to support V&V.

Section 8. Foundational Standards Alignment

Cloud security architecture and resilience must closely align with globally recognized foundational standards to ensure interoperability, regulatory compliance, and a consistent risk management approach. ISAUnited Defensible Standards provide detailed technical guidance and engineering rigor; however, alignment with established frameworks remains crucial for auditability, industry acceptance, and seamless integration into existing security and compliance programs.

Table B-3: Foundational standards relevant to this Parent Standard

Framework	Standard ID	Reference Focus
NIST	Cybersecurity Framework 2.0	Organizational risk framework organized by six Functions—Govern, Identify, Protect, Detect, Respond, Recover—used to align strategy and measurement; technology-agnostic and applicable to cloud programs.
NIST	SP 800-53 Rev. 5	Security and privacy controls for systems and services (identity, boundary, data protection, monitoring).
NIST	SP 800-144	Guidelines on security and privacy in public cloud computing.
NIST	SP 800-207	Zero Trust Architecture principles and reference models applicable to cloud.
ISO/IEC	27001:2022	ISMS requirements for establishing, implementing, maintaining, and continually improving information security.
ISO/IEC	27002:2022	Code of practice for information security controls (guidance for selecting and implementing controls).
ISO/IEC	27017	Code of practice for information security controls for cloud services.
ISO/IEC	27018	Protection of personal data in public cloud services.
ISO/IEC/IEEE	15288	System life cycle processes; aligns §5 Requirements → §6 Outputs → §10.5 Implementation → §12 V&V with a V-model trace.
ISO/IEC/IEEE	29148	Requirements engineering; strengthens §5 inputs to be singular, testable, traceable, and reviewable with acceptance criteria.

As sub-standards are developed and published under this Parent Standard, more specific references (for example, clause-level mappings to NIST and ISO/IEC) will be included to facilitate precise implementation and validation.

Practitioner Guidance:

Use this Parent Standard to drive design; cite ISO/IEC and NIST here for audit traceability.

- Map each §6 output to at least one clause in NIST or ISO/IEC; where helpful, use CSF 2.0 Functions as a high-level overlay for program alignment and reporting. NIST Computer Security Resource Center
- Maintain a single NIST ↔ ISO/IEC crosswalk per sub-standard to prevent drift.
- Reserve control frameworks (for example, CSA CCM, CIS Controls, MITRE ATT&CK) for §9 Security Controls, not for foundational alignment.

Section 9. Security Controls

This section identifies the technical control families and control references directly supported or enforced by the Cloud Security Architecture & Resilience Parent Standard. These controls link architectural and engineering guidance to recognized cybersecurity frameworks, ensuring traceability, auditability, and consistent implementation across diverse cloud environments.

Purpose and Function

Security controls translate the architectural intent of this standard into actionable, measurable safeguards. They provide the tactical foundation for enforcing confidentiality, integrity, availability, authentication, authorization, and auditability in cloud environments.

By mapping to accepted frameworks such as the CSA Cloud Controls Matrix (CCM), CIS Controls v8, and OWASP API Security, ISAUnited enables:

- Clear alignment with recognized best practices and regulatory expectations.
- Interoperability across organizational contexts and providers.
- Consistency and reusability of controls in sub-standards aligned to this Parent Standard, supporting structured implementation and validation.

These mappings allow engineers and auditors to measure and validate the defensibility of implementations guided by this standard.

Implementation Guidance

Authors and practitioners must:

- Reference at least three concrete technical controls from one or more authoritative control frameworks.
- Cite framework name, control identifier, and a concise description.
- Align chosen controls explicitly to §6 Technical Specifications and §7 Core Principles.
- Select implementation-level controls rather than high-level policy statements.

Table B-4. Control Mappings for Cloud Security Architecture & Resilience

Framework	Control ID	Control Name / Description
CSA CCM	IAM-09	Identity & Access Management — Enforce strong authentication (for example, MFA) for cloud access to reduce unauthorized access risk.
CSA CCM	DSI-03	Data Security & Information Lifecycle — Encrypt data at rest and in transit (e.g., AES-256, TLS 1.3) using managed keys.
CSA CCM	IVS-09	Virtualization & Network Security — Use segmentation and micro-segmentation to isolate sensitive workloads and reduce lateral movement.
CIS Controls v8	5.1	Inventory of Service Accounts — Identify, manage, and restrict cloud service accounts to minimize attack surface.
CIS Controls v8	13.1	Network Protections — Segment cloud networks and apply least-privilege access controls.
CIS Controls v8	14.4	Centralized Security Event Logging — Aggregate and monitor cloud security events in a SIEM with automated alerting.
OWASP API Security Top 10	API2	Broken Authentication — Enforce consistent API authentication (OAuth 2.0, OpenID Connect with JWT, or mutual TLS).

Additional References

As the domain matures, authors may include supplementary controls from these frameworks to maintain robustness and relevance.

Sub-Standard Expectations

Sub-standards developed under this Parent Standard must:

- Select and enforce explicit technical controls relevant to their focus area (for example, IAM, encryption, segmentation, Zero Trust enforcement).
- Provide detailed mappings of these controls to defined validation, implementation, and operational criteria.
- Justify and document any deviation from control families referenced at this Parent Standard level to ensure transparency and defensibility.

This structured approach ensures cloud security architectures derived from ISAUnited's Defensible Standards are consistently defensible, auditable, and measurable against recognized best practices.

Section 10. Engineering Discipline

This section defines the architectural thinking, rigorous engineering processes, and disciplined operational behaviors required to implement the Cloud Security Architecture & Resilience Parent Standard. ISAUnited's Defensible Standards are not compliance checklists; they are engineered systems, grounded in systems thinking, critical reasoning, and Verification & Validation (V&V), that produce measurable, auditable, defensible outcomes in cloud security.

10.1 Purpose & Function

Purpose. Establish a repeatable, auditable way of working that integrates systems thinking, lifecycle controls, adversary-aware design, and measurable outcomes.

Function in D10S. Parent Standards set expectations and invariants. Sub-Standards convert them into controls-as-code, test specifications, and evidence artifacts embedded in delivery and operations.

10.2 Systems Thinking

Goal: Make the system legible end-to-end, components, interfaces, dependencies, and failure modes—so controls are placed where risk manifests.

10.2.1 System Definition & Boundaries

- Declare system purpose, scope, stakeholders, and in-scope and out-of-scope assets.
- Model trust zones, segmentation, and interconnects (accounts/subscriptions/projects, VPC/VNet, peering, service endpoints, private links).

10.2.2 Interfaces & Contracts

- Maintain Interface Control Documents (ICDs) for every interconnection (APIs, queues, data stores, identity providers).
- For each interface, specify authentication and authorization model, data classification, rate and flow limits, error handling, telemetry, and security invariants.

10.2.3 Dependencies & Emergent Behavior

- Map shared services (KMS, DNS, IAM, logging) and blast radius per dependency.
- Identify emergent risks from composition (for example, benign configuration at A + default at B → exploitable path).

10.2.4 Failure Modes & Safeguards

- For critical paths, document failure modes (misconfiguration, drift, overload, credential abuse) and safeguards (deny by default, least privilege, rate caps, circuit breakers, canary deploys, immutable infrastructure).
- Treat security invariants as non-negotiable requirements (for example, “no public ingress to the management plane,” “customer data exits the zone only via approved egress”).

Required Artifacts (minimum): Context diagram with trust boundaries; interface map with ICDs; dependency and blast-radius matrix; invariants register.

10.3 Critical Thinking

Goal: Replace assumptions with explicit reasoning that survives review, attack, and audit.

10.3.1 Decision Discipline

- Use Architecture Decision Records (ADRs): problem → options → constraints and assumptions → trade-offs → decision → invariants → test and evidence plan.

10.3.2 Engineering Prompts

- **Boundaries:** What is the system? Where are the trust boundaries and why?
- **Interfaces:** What must always be true at each interface (invariants)? How do we test it?
- **Adversary:** Which attack techniques are credible here? What is the shortest attack path?
- **Evidence:** What objective signals prove this control works today and after change?
- **Failure:** When this fails, does it fail safe? What is the operator’s next action?

Required Artifacts (minimum): ADRs; assumptions and constraints log; evidence plan per decision.

10.4 Domain-Wide Engineering Expectations

Secure System Design

- Define cloud security boundaries (accounts/subscriptions/projects, VPC/VNet, subnets, security groups/ACLs, routing).
- Validate boundaries and trust relationships through structured architecture reviews with artifacts from §10.2.

Implementation Philosophy — "Built in, not bolted on"

- Integrate controls at design time and in pipelines; avoid post hoc patching.
- Express controls as policies and configurations as code bound to the invariants in §10.2.4.

Lifecycle Integration

- Embed controls into DevSecOps (IaC/PaC), change management, and immutable deployments.
- Enforce version-controlled reviews with required ADRs and evidence updates.

Verification Rigor (V&V)

- Combine automated checks (policy validation, IaC scanning, runtime guardrails) with manual tests (penetration testing, adversary emulation).
- Require continuous validation in pipelines and runtime monitoring tied to invariants.

Operational Discipline

- Monitor for drift and unauthorized change; auto-remediate where safe.
- Maintain pre-approved playbooks for misconfiguration, key rotation, incident containment, and rollback.

10.5 Engineering Implementation Expectations

- **Policy and configuration as code.** Manage policies and configurations as code under version control with peer review and provenance.
- **Structured enforcement pipelines.** CI/CD gates for unit and policy tests → security integration tests → canary/blue-green → rollback.
- **Explicit security boundaries.** Maintain diagrams and ICDs; perform continuous validation with posture checks and targeted audits.
- **Automated security testing.** Integrate IaC scanning, configuration validation, secrets detection, dependency checks, and adversary emulation before production.
- **Traceable architecture decisions.** Link ADRs to controls, tests, and evidence; update ADRs and evidence on every change request.

Required Artifacts (minimum): Controls-as-code repository; pipeline policy gates; boundary/ICD set; automated test results; evidence ledger (see §10.7 and §12).

10.6 Sub-Standard Alignment (inheritance rules)

Sub-Standards must operationalize this discipline with domain-specific detail:

- **Cloud IAM (for example, ISAU-DS-2010).** IAM policies managed as IaC; least-privilege baselines; versioned policy definitions; automated policy validation; pipeline enforcement; mandatory peer review and automated tests before deploy.
- **Zero Trust Cloud Access.** Continuous authentication and authorization; session risk evaluation; per-request policy evaluation; telemetry-verified decisions; attack-path tests relevant to access abuse.

10.7 Evidence & V&V (what proves it works)

Establish an Evidence Pack per system containing:

- **Design evidence:** diagrams with trust boundaries, ICDs, invariants register, ADRs.
- **Build evidence:** IaC/PaC repositories, signed artifacts, pipeline logs, test results.
- **Operate evidence:** runtime policy decisions, drift reports, control telemetry, incidents, and rollback records.
- **Challenge evidence:** red team and penetration-test reports, adversary-emulation outcomes, remediation closure with re-test.

Each control requires objective pass/fail criteria, a test frequency, a responsible owner, and a retention policy. Map Evidence Pack IDs into §12 traceability.

10.8 Example: Sub-Standard Discipline Alignment (Cloud IAM)

Scope: ISAU-DS-2010 Cloud IAM & Access Security
Design: Define identity trust zones; enumerate principals, roles, and interface invariants (for example, “no human keys for production”).
Implement: Manage IAM as code; enforce deny-by-default, scoped roles, and conditional policies; block long-lived credentials.
V&V: Automated policy tests and negative tests (ensure over-privilege fails); drift detection; periodic adversary emulation focused on privilege escalation and token misuse.
Operate: The Evidence Pack includes policy repository history, pipeline-gate results, runtime policy decisions, incident records, and closed-loop remediation.

10.9 Systems Engineering Overlay (normative)

- **V-model alignment:** Map §5 (Requirements) → §6 (Design Outputs) → §10.5 (Implementation) → §12 (Verification & Validation). Every change SHALL update the bidirectional trace.
- **Requirements quality:** Each §5 item SHALL be singular, testable, and measurable with acceptance criteria, owner, and verification method.
- **Hazard analysis:** Perform FMEA (and, where useful, fault-tree analysis) on critical paths from §10.2.4; record failure modes, effects, severity, and safeguards; link resulting tests to §12.

- **Interface budgets:** For each ICD, define quantitative limits (for example, rate, latency, allowed methods, payload size), plus security invariants; test budgets via synthetic positive/negative cases before release.
- **Tolerances and acceptance:** Define pass/fail thresholds for block rates, false-positive ceilings, latency overhead of controls, and recovery time objectives; store in the Evidence Pack.

Required Artifacts (minimum): V-model trace map; FMEA worksheet with mitigations; ICD limits and test cases; acceptance-criteria register linked to Table B-6.

Section 11. Associate Sub-Standards Mapping

Purpose of Sub-Standards

ISAUnited Defensible Sub-Standards are detailed, domain-specific extensions of the Cloud Security Architecture & Resilience Parent Standard (ISAU-DS-CS-1000). Each Sub-Standard delivers:

- Granular technical guidance tailored to specialized cloud security domains.
- Actionable implementation strategies translating architectural intent into practical operational controls.
- Precise validation methodologies so outputs are measurable and auditable.
- Alignment with the foundational architectural principles and Technical Specifications of this Parent Standard.

Sub-Standards bridge the gap between broad architectural direction and the detailed technical requirements necessary for robust engineering, validation, and auditing in cloud environments.

Scope and Focus of Cloud Sub-Standards

Sub-Standards under ISAU-DS-CS-1000 will address topics including, but not limited to:

- Cloud network segmentation and isolation: VPC/VNet patterns, hub-and-spoke, private endpoints, micro-segmentation, contract tests.
- Cloud IAM and access security: Least-privilege roles, JIT elevation, periodic access reviews, workload identities.
- Zero Trust cloud access: Continuous verification at boundaries, identity, and context policy evaluation.

- Cloud monitoring, detection, and response: Unified telemetry, CSPM gates, automated playbooks.
- Data protection and key management: Encryption defaults, KMS rotation, data-flow protections, and lifecycle validation.
- API security and workload runtime controls: OAuth 2.0/OIDC with JWT, gateway enforcement, image signature/attestation, serverless caps, and allowlists.

Table B-5. Example Future Sub-Standards (Cloud Security Architecture & Resilience):

Sub-Standard ID	Sub-Standard Name	Focus Area
ISAU-DS-CS-1001	Cloud IAM and Access Security	IAM
ISAU-DS-CS-1002	Cloud Network Segmentation and East–West Control	Segmentation
ISAU-DS-CS-1003	Egress Control and Zone Allowlisting	Egress Control
ISAU-DS-CS-1004	Cloud Data Protection and Key Management	Data Protection & Encryption
ISAU-DS-CS-1005	API Security and Gateway Enforcement	API & Runtime Security
ISAU-DS-CS-1006	Workload Runtime Security (VMs, containers, serverless)	Runtime Security
ISAU-DS-CS-1007	Posture Management and Drift Control (CSPM + IaC/PaC gates)	Posture & Drift
ISAU-DS-CS-1008	Centralized Logging, Telemetry, and Evidence Production	Logging & Evidence
ISAU-DS-CS-1009	Zero Trust Cloud Access	Zero Trust
ISAU-DS-CS-1010	Cloud Backup, Restore, and Resilience Drills	Resilience & Recovery
	SaaS Integration Governance (Identity & Telemetry)	SaaS Governance

Sub-Standard ID	Sub-Standard Name	Focus Area
ISAU-DS-CS-1011		
ISAU-DS-CS-1012	Private Service Endpoints and Managed Service Isolation	Managed Services Isolation
ISAU-DS-CS-1013	Elasticity-Safe Controls (Autoscale and Re-Admission Checks)	Elasticity Controls
ISAU-DS-CS-1016	Interface Contracts and Synthetic Testing for Exposed Paths	Interface Contracts
ISAU-DS-CS-1017	Multi-Tenant Isolation and Quotas	Multi-Tenant Isolation
ISAU-DS-CS-1019	Cloud Change Control in CI/CD (Fail-Closed Gates)	Change Control
ISAU-DS-CS-1020	Incident Response Playbooks for Cloud Workloads	Incident Response

Note: As the suite of Sub-Standards expands, each inherits the engineering discipline, V&V rigor, and architectural alignment required to maintain consistency, defensibility, and auditability across all implementations under the ISAUnited Defensible Standards framework.

Each Sub-Standard Will Specify

- Inputs (Requirements): Preconditions for implementation.
- Outputs (Technical Specifications): Measurable engineering deliverables.
- Verification & Validation: Test and verification methods with Evidence Pack IDs.
- Implementation Guidelines: Practical, scalable deployment guidance.

Development and Approval Process

- Open Season submission: Members and registered contributors submit proposed Sub-Standards aligned with ISAU-DS-CS-1000 objectives and scope.
- Technical peer review: The Technical Fellow Society evaluates submissions for engineering validity, technical accuracy, alignment with core principles, and practical applicability.
- Approval and publication: Approved Sub-Standards receive formal versioning and publication as authoritative extensions of ISAU-DS-CS-1000.

Publication Timeline

ISAUnited will publish approved cloud security Sub-Standards on a rolling basis. Each publication will include scope, versioning, effective dates, and mappings to §5 Requirements, §6 Technical Specifications, and §12 Verification & Validation.

Practitioner Guidance:

Keep a single cross-walk that ties each Sub-Standard's §5 Requirements to §6 Outputs and §12 V&V activities. Assign Evidence Pack IDs at draft creation to prevent drift, and update the cross-walk with every change.

Section 12. Verification and Validation

Ensuring the defensibility and effectiveness of cloud security architecture requires a comprehensive, engineering-driven approach to testing and validation. This section offers actionable recommendations for rigorously assessing cloud environments, whether utilizing cloud-native capabilities or third-party solutions, ensuring validation remains technology-agnostic, adaptable, and robust across various deployment models.

> **Verification** confirms that the system has been implemented in accordance with the defined Requirements (Inputs) (§5) and Technical Specifications (Outputs) (§6).
>
> **Validation** confirms that the system performs effectively under real-world operational conditions and withstands adversarial testing.

Core Verification Activities

- Confirm that all cloud security controls defined in §6 are implemented in the target environment—Landing Zone guardrails, segmentation and private endpoints, mTLS for east–west and administrative paths, API-gateway authentication and authorization, artifact signature and attestation checks at admission, encryption defaults with KMS, unified logging schema with authenticated time synchronization, posture gates (CSPM), and DR/BCP runbooks.

- Review and validate configuration baselines and organization-level policy constraints (for example, network boundaries, IAM role policies, key policies, DNS egress filters, tagging rules, quotas, and runtime limits) against recognized engineering and security benchmarks.
- Verify interoperability at integration points so that segmentation, Zero Trust enforcement, artifact verification, and egress controls do not introduce new vulnerabilities or disrupt business-critical services.
- Conduct peer review of architecture diagrams, Landing Zone checklist, segmentation maps, API exposure inventory and contracts, ICDs, and control mappings to ensure completeness and accuracy.
- Demonstrate fail-closed CI/CD promotion gates for critical checks (for example, posture findings, artifact verification), with pipeline evidence recorded.

Core Validation Activities

- Perform adversarial testing—penetration testing, red teaming, and BAS-informed emulation—focusing on lateral-movement resistance across accounts or projects, boundary control effectiveness, identity escalation paths, Zero Trust enforcement, and egress governance.
- Validate posture using automated and manual methods across providers to confirm resilience under realistic threat models; include elasticity events (scale-up/scale-out) to verify re-admission and integrity checks at runtime.
- Test operational resilience—region/zone failover of critical services, disaster-recovery routing, and incident response tied to cloud events—against defined service-level objectives.
- Measure control performance against metrics such as Mean Time to Detect (MTTD), Mean Time to Contain (MTTC), recovery objectives, segmentation block-rate, gate fail-closed rate, and logging immutability.

Required Deliverables

1. Test Plans and Procedures – Detailed scope, tools, environments, and methods for verification and validation phases.
2. Validation Reports – Results with pass/fail status, residual-risk ranking, and remediation priorities.
3. Evidence Artifacts – Logs, configuration exports, policy diffs, pipeline records, alerts, and drill outputs proving test execution and results—each labeled with an Evidence Pack ID (EP-02.x) and referenced in Table B-6.
4. Corrective Action Plans – Documented remediation steps for findings that require resolution before system acceptance.

Common Pitfalls to Avoid

- Treating penetration testing as a check-the-box exercise rather than a rigorous, adversary-informed assessment.
- Failing to document validation activities, creating gaps in audit trails and lessons learned.
- Neglecting continuous validation in dynamic or high-risk workloads or during scale events.
- Overlooking integration points between cloud-native and third-party controls.

Table B-6. Traceability Matrix: Requirements (§5) → Verification/Validation (§12) → Technical Specifications (§6):

Requirement ID	Requirement (summary)	Verification (build-correct)	Validation (works-right)	Related Technical Specs
5.1	Zero Trust Cloud Security	• MFA enabled for all privileged accounts; least-privilege policies present • Dynamic authorization policies configured in the identity platform	• Phishing/token-theft simulations require step-up for privileged actions • Lateral-movement attempts blocked by policy	§6.1 Identity & Access; §6.2 Network & Segmentation
5.2	Shared responsibility alignment	• Responsibility matrix documented per provider • Procedures to monitor provider vs organization controls	• Sample provider defaults (for example, storage encryption) and organization controls (for example, key rotation) confirmed effective	§6.3 Data Protection & Encryption; §6.5 Monitoring & IR
5.3	Automated security enforcement	• CSPM and IaC/PaC gates active across accounts/projects • Auto-remediation policies configured	• Safe misconfiguration in staging is blocked or auto-remediated within the target window	§6.5 Monitoring & IR; §6.4 API & Workload Security
5.4	Cloud network segmentation	• Logical and micro-segmentation policies deployed • Private endpoints configured for sensitive services	• BAS across trust boundaries; east–west movement attempts blocked	§6.2 Network & Segmentation
5.5	Cloud data encryption & compliance	• Encryption by default at rest and in transit • Centralized KMS with rotation; data classified and tagged	• Encrypted restore drill succeeds; transport scans meet policy; key/cert hygiene checks pass	§6.3 Data Protection & Encryption
5.6	Landing Zone baseline	• Guardrail checklist completed (identity, network,	• Attempted promotion with one guardrail disabled fails closed;	§6.2 Network & Segmentation

Requirement ID	Requirement (summary)	Verification (build-correct)	Validation (works-right)	Related Technical Specs
		logging, tagging, baseline policies); promotion gate in place	spoke contract tests pass/deny as expected	
5.7	Unified telemetry and evidence readiness	• Unified logging schema enabled; authenticated time synchronization verified; Evidence Pack repo created	• Inject test events and confirm central visibility; tamper-evident retention verified	§6.5 Monitoring & IR
5.8	Artifact integrity and approved sources	• Admission/pipeline policy enforces signature/attestation; allowlist of registries/namespaces; mutable tags denied	• Unsigned or unapproved-registry image is denied in stage; scale event re-checks integrity	§6.4 API & Workload Security
5.9	Interface contracts and private endpoints	• ICDs exist per exposed interface; private endpoints configured where feasible; contract tests defined	• Positive/negative synthetic tests pass; no public exposure without time-bounded exception	§6.2 Network & Segmentation; §6.4 API & Workload Security
5.10	Multi-tenant isolation and quotas	• Per-tenant quotas and runtime limits configured; micro-segmentation applied	• Noisy-neighbor simulation shows isolation; quota/limit enforcement observed in telemetry	§6.2 Network & Segmentation

Evidence guidance

- Attach plans and procedures, approved diagrams, policy-as-code repositories, pipeline logs, scan reports, posture findings, API gateway exports, segmentation maps, transport scan results, KMS rotation logs, restore-drill outputs, and dated sign-offs. Include authenticated time-synchronization evidence and proof of immutable log retention.
- Store artifacts in a secure repository and reference each row with an EP-02.x ID in this matrix.

How to use this matrix

- During planning: confirm each §5 requirement has at least one verification and one validation activity scheduled.
- During execution: record the EP-02.x ID for each row when completed.
- During review: when a requirement or control changes, update its linked activities and §6 references to keep the chain intact.

Practitioner Guidance:

Treat §12 as a continuous engineering function, not a one-time event.

- Map every §5 requirement to one verification and one validation in Table B-6, each with a unique EP-02.x ID.
- Exercise BAS techniques that match your architecture; track MTTD/MTTC against targets and adjust controls.
- Validate management-plane isolation, admission checks, and egress default-deny during every major change window.

Quick Win Playbook:

Title: Establish V&V Traceability for a Single Requirement (§12)

Objective: Create an end-to-end, auditable chain from one §5 requirement to verification and validation tests, with artifacts recorded under EP-02.2.

Target: Stand up V&V traceability and an Evidence Pack entry for one requirement (for example, §5.4 Cloud network segmentation).

Component/System: Traceability matrix (Table B-6), CI/CD pipeline job (platform test runner), synthetic/BAS tester, centralized telemetry and evidence repository.

Protects: Verification gaps and audit failure by proving each requirement has defined tests, owners, cadence, and stored artifacts.

Stops/Detects: Untested changes, missing evidence links, stale test schedules, segmentation or egress regressions.

Action: Add a new Table B-6 row linking the chosen §5 requirement to one verification and one validation test and the related §6 outputs; implement a CI job that 1) pulls current configs/policies for verification, 2) executes a negative and a positive synthetic path test for validation (for example, deny east–west across zones, deny unallowlisted egress, allow a documented contract), 3) fails closed on critical findings, and 4) uploads all artifacts to EP-02.2.

Proof: Completed Table B-6 row, test plan and procedure, CI job logs, policy/config exports used for verification, synthetic/BAS outputs, matching SIEM/XDR events with timestamps; attach to Evidence Pack ID EP-02.2.

Metric: Both tests execute with recorded pass/fail results; block events appear in centralized telemetry within the target MTTD; artifacts are stored under EP-02.2; next scheduled run and owner are documented.

Rollback: Revert the Table B-6 row to the prior version if needed; disable or roll back the CI job; archive all newly created artifacts under EP-02.2 as superseded.

By embedding these practices into the Parent Standard, organizations ensure that their cloud security architecture is not only compliant and auditable but also practical, resilient, and defensible—regardless of whether controls utilize cloud-native features or third-party products. This approach establishes a foundation for future Sub-Standards and supports consistent, engineering-grade validation across the cloud security lifecycle.

Section 13. Implementation Guidelines

This section does not prescribe vendor-specific tactics. Parent Standards are stable, long-lived architectural foundations. Here, we define how sub-standards and delivery teams must translate the Parent's intent into operational behaviors that are testable, automatable, and auditable.

Purpose of This Section in Sub-Standards

Sub-standards must use Implementation Guidelines to:

- Translate architectural expectations from the Parent Standard into enforceable run-time and pipeline behaviors.
- Provide platform-agnostic practices that improve adoption, avoid failure, and align with ISAUnited's defensible design philosophy.
- Highlight common failure modes and how to prevent them with measurable gates and checks.
- Offer repeatable patterns (as code) that enforce controls, trust models, and engineering discipline.

Open Season Guidance for Contributors

Contributors developing sub-standards Must:

- Align all guidance with the strategic posture in this Parent Standard.
- Avoid vendor or product terms; express controls as requirements, tests, and evidence.
- Include lessons learned (what fails, why, and how the test proves it).
- Focus on repeatable engineering patterns, not one-offs.

• Provide a minimal Standards Mapping (Spec/Control → NIST/ISO clause from §8 → Evidence Pack ID).

Technical Guidance

A. Organizing Principles (normative)

1. **Everything as code** – Policies, configs, network intents, pipelines, runbooks, and tests Must be version-controlled, peer-reviewed, and promoted through environments on protected branches.
2. **Gated change** – Every merge and deployment Must pass non-bypassable security gates tied to quantitative acceptance criteria (see §6 and §12).
3. **Immutable, reproducible releases** – No manual device or policy changes post-build; releases Must be reproducible from source and verified at deploy.
4. **Least privilege & JIT** – Pipeline identities, automation runners, and administrators Must use scoped permissions with time-bound elevation; break-glass Must be exceptional and fully audited.
5. **Environment parity** – Staging Must mirror production controls (authn/z, egress, TLS/mTLS, logging schema) so test results are predictive; drift Must be monitored and reconciled.

B. Guardrails by Pipeline Stage (normative)

1. **Pre-commit / local**
   • Secrets scanning and commit signing required.
   • Pre-commit hooks Should run linters and policy checks for network/IaC definitions.
2. **Pull request (PR) / code review**
   • Code owner approval required; Threat-Model Delta recorded in the PR template for significant change.
   • IaC policy-as-code gate (or equivalent) for segmentation, identity, cryptography, logging, and egress rules; Critical = 0.
   • Require evidence pointers in the PR (planned tests and Evidence Pack ID stubs).
3. **Build & package**
   • Deterministic artifacts (pinned versions; no ad hoc fetch at deploy).
   • Artifacts signed; integrity verified prior to promotion.
   • Transitive dependency review for automation and pipeline components.
4. **Pre-deploy / release**
   • Config drift detection against approved baselines; change approval as code.
   • Progressive rollout (staged/canary) for network policies; define health SLOs and automatic rollback.
   • Negative/positive traffic contract tests for inter-zone flows; egress allowlist tests.
5. **Deploy & runtime**
   • TLS 1.3 at edges; mTLS for service-to-service/admin paths where required;

certificates managed via PKI/KMS with rotation.
• Egress allowlists per zone/workload; runners and automation are isolated with restricted outbound.
• Unified logging schema (timestamp, actor, action, resource, result, trace_id, control_id, env); logs to immutable store with authenticated time sync.
• Management-plane isolation with bastion, MFA/JIT, and full session recording.

6. **Post-deploy validation & operations**
   • Continuous validation (BAS/adversary-emulation scenarios) scheduled; failover/DR routing drills.
   • Security SLOs tracked: target MTTD/MTTC per §12; segmentation block-rate goals; egress violations = 0 in sensitive zones.
   • Auto-generate an Evidence Pack per release (configs, policy diffs, validation results, logs, drift reports, ADR links).

C. Identity, Secrets, and Keys (normative alignment to §6)

- Use KMS for key storage; define certificate issuance, rotation, and revocation; maintain service identity inventories.
- Use short-lived credentials for pipelines and bastions; scope secrets to job/environment; redact in logs.
- No secrets in repositories or device images; inject at runtime; full auditability of access.

D. Supply-Chain Integrity (normative)

- Only deploy signed, verified configurations and images from trusted sources; restrict registries/repositories.
- Quarantine and verify third-party artifacts (scripts, modules); enforce license and integrity checks.
- Separate build and deploy identities; forbid production writes from build jobs.

E. Measurement & Acceptance (aligned to §6 and §12)

- mTLS coverage for designated paths meets target; certificate inventory current with no expirations inside policy window.
- Zone egress: default-deny enforced; allowlisted destinations only; exceptions time-bounded with approvals.
- Logging: authenticated time sync; required fields present; evidence retention immutable.
- Detection: MTTD/MTTC targets met for boundary and east–west anomalies; monthly review and tuning.
- Each change linked to an Evidence Pack ID tying artifacts to §5 → §6 → §12.

Common Pitfalls (and the engineered countermeasure)

1. Pipelines as suggestions → Enforce non-bypassable gates; block merges and releases on fails; store failing artifacts as proof.
2. One-time scanning → Treat checks as gates with thresholds; require coverage for changed items.
3. Manual hot-fixes/drift → Detect and reconcile drift; forbid out-of-band edits; require Architecture Decision Records.
4. Open egress / shared runners → Isolate runners; restrict outbound; allowlist per zone/workload.
5. Management plane exposure → Bastion-only with MFA/JIT; block direct access from production subnets.
6. Weak crypto / stale certs → Enforce TLS 1.3/mTLS where required; rotate and monitor via PKI/KMS.
7. Incomplete logging/time → Enforce unified schema, authenticated time sync, and immutable retention.
8. No evidence → Every release Must have an Evidence Pack ID with linked tests and results.

ISAUnited encourages organizations to utilize these guidelines as foundational references for continuous improvement. Although detailed technical instructions and controls will be elaborated upon in subsequent sub-standards, consistently applying these guidelines will significantly enhance the cloud security posture and ensure operational resilience.

Practitioner Guidance:

- Treat these guidelines as operational defaults; exceptions require written justification and time-boxed compensating controls.
- Map each practice to a §5 readiness input, a §6 output, and a §12 test; assign an Evidence Pack ID (EP-02.*) for traceability.
- Maintain a single source of truth (diagrams, policies, repositories) to reduce drift; review quarterly or after major architectural change.
- Enforce fail-closed CI/CD gates on missing MFA, segmentation policies, encryption settings, or PaC checks.
- Record owners and approvers for every change; require two-person review for privileged changes.
- Capture before/after diffs and attach them to the Evidence Pack to support verification and audits.

Quick Win Playbook:

Title: Change-Control Gate for Cloud Configuration-as-Code (§13)

Objective: Enforce a non-bypassable gate that requires an Evidence Pack ID and two-person approval before cloud configuration changes are promoted.

Target: Establish a protected-branch gate for cloud configuration-as-code (for example, IAM policies, network rules, egress allowlists, KMS settings, logging/telemetry baselines).

Component/System: Config/policy repository, pipeline job (policy gate), branch protection rules, approval workflow.

Protects: Production cloud posture from unreviewed or unevidenced changes to identity, network, encryption, and logging controls; maintains auditability.
Stops/Detects: Direct-to-main commits, single-approver merges, missing Evidence Pack ID, skipped policy tests, undeclared drift.

Action: Add a mandatory gate that checks for
1) a valid EP-02.* reference in the pull-request template,
2) two-person approval by designated code owners, and
3) passing policy tests;
configure the pipeline to fail closed when any condition is missing; run one negative test PR to confirm rejection, then a compliant PR to confirm pass.

Proof: Branch-protection configuration, gate/pipeline configuration diff, failed run log (gate rejection), approved run log (gate pass); attach to Evidence Pack ID EP-02.3.

Metric: 100 % of configuration pull requests include an Evidence Pack ID and two-person approval; 0 direct-to-main merges; gate pass/fail rate is recorded for review.

Rollback: Revert the gate configuration and branch-protection settings to the previous version; redeploy the prior commit; archive the new artifacts under EP-02.3 as superseded.

Appendices

Appendix A. Engineering Traceability Matrix:

Req ID	Requirement (Inputs) (§5)	Technical Specifications (Outputs) (§6)	Core Principles (§7)	Control Mappings (§9)	Verification (Build-Correct) (§12)	Validation (Works-Right) (§12)	Evidence Pack ID
5.1	Zero Trust Cloud Security	§6.1 Identity & Access Security; §6.2 Network & Segmentation	RP-02 Zero Trust; RP-01 Least Privilege	CSA CCM IAM-09; CIS 5.1	MFA enabled; least-privilege IAM review; dynamic authorization policies verified	Phishing/token-theft simulations require step-up; lateral-movement attempts blocked	EP-02.11
5.2	Shared Responsibility Model Alignment	§6.3 Data Protection & Encryption; §6.5 Monitoring & Incident Response	RP-05 Secure by Design; RP-15 Evidence Production	CSA CCM DSI-03; CIS 14.4	Shared-responsibility matrix documented; provider/org control mapping reviewed	Provider defaults (for example, storage encryption) and organization controls (for example, key rotation) confirmed effective	EP-02.12
5.3	Automated Security Enforcement	§6.5 Monitoring & Incident Response; §6.4 API & Workload Security	RP-10 Secure Defaults; RP-15 Evidence Production	CIS 13.1; CSA CCM IVS-09	CSPM and IaC/PaC gates active; auto-remediation configured	Misconfiguration in staging blocked or auto-remediated within the target window; posture gates enforced	EP-02.13

Req ID	Requirement (Inputs) (§5)	Technical Specifications (Outputs) (§6)	Core Principles (§7)	Control Mappings (§9)	Verification (Build-Correct) (§12)	Validation (Works-Right) (§12)	Evidence Pack ID
5.4	Cloud Network Segmentation	§6.2 Network & Segmentation	RP-04 Defense in Depth; RP-06 Minimize Attack Surface	CSA CCM IVS-09; CIS 13.1	Logical segmentation; private endpoints configured	BAS east–west tests blocked; hub-and-spoke contract tests pass; mTLS validated.	EP-02.14
5.5	Cloud Data Encryption & Compliance	§6.3 Data Protection & Encryption	RP-18 Protect Confidentiality; RP-10 Secure Defaults	CSA CCM DSI-03; CIS 14.4	Encryption defaults validated; KMS rotation policies reviewed	Encrypted restore drill succeeds; transport scans meet policy; key/cert hygiene passes.	EP-02.15
5.6	Landing Zone Baseline	§6.2 Network & Segmentation	RP-10 Secure Defaults; RP-14 Resilience & Recovery	CSA CCM IVS-09; CIS 13.1	Guardrail checklist complete (identity, network, logging, tagging, baseline policies)	Promotion with a missing guardrail fails closed; spoke contract tests pass.	EP-02.16
5.7	Unified Telemetry & Evidence Readiness	§6.5 Monitoring & Incident Response	RP-15 Evidence Production; RP-20 Protect Availability	CIS 14.4; CSA CCM DSI-03	Unified logging schema validated; authenticated time synchronization verified; EP repository created	Injected test events appear in SIEM/XDR within the MTTD target; retention is tamper-evident.	EP-02.17

Req ID	Requirement (Inputs) (§5)	Technical Specifications (Outputs) (§6)	Core Principles (§7)	Control Mappings (§9)	Verification (Build-Correct) (§12)	Validation (Works-Right) (§12)	Evidence Pack ID
5.8	Artifact Integrity & Approved Sources	§6.4 API & Workload Security	RP-05 Secure by Design; RP-10 Secure Defaults	CSA CCM IVS-09; CIS 5.1	Admission policy enforces signature/attestation; mutable tags denied; approved registries/namespaces allowlisted	Unsigned or unapproved-registry image denied; runtime integrity checks enforced.	EP-02.18
5.9	Interface Contracts & Private Endpoints	§6.2 Network & Segmentation; §6.4 API & Workload Security	RP-06 Minimize Attack Surface; RP-02 Zero Trust	CSA CCM IVS-09; OWASP API2	ICDs created per exposed interface; private endpoints validated; contract tests defined	Synthetic positive and negative tests pass; no public exposure without a time-bounded exception.	EP-02.19
5.10	Multi-Tenant Isolation & Quotas	§6.2 Network & Segmentation	RP-20 Protect Availability; RP-01 Least Privilege	CSA CCM IVS-09; CIS 13.1	Tenant quotas and runtime limits configured; micro-segmentation applied	Noisy-neighbor simulation confirms isolation; limit enforcement observed in telemetry.	EP-02.20

Appendix B. EP-02 Summary Matrix – Evidence Pack Overview:

Layer	EP Identifier	Purpose	Evidence Categories Included
Parent EP	EP-02	Master Evidence Pack for the D02 Parent Standard. Stores global cloud architecture evidence, Landing Zone artifacts, identity models, invariants, and cross-provider V&V artifacts supporting §§5, 6, 10, and 12.	• Cloud architecture diagrams • Landing Zone guardrail verification (identity, network, telemetry, tagging) • Trust boundaries (accounts, subscriptions, projects) • Identity models (IAM roles, ABAC/RBAC policies) • Invariants register • Interface Control Documents (ICDs) • Cross-cloud segmentation maps • Parent-level V&V evidence (Table B-6) • Provider and multi-cloud logs, scan results, configs
Sub-EP (Quick Win)	EP-02.1	§6 Quick Win – API Security & Gateway Authentication. Stores boundary enforcement evidence for one Internet-exposed API.	• API gateway configurations • OAuth/OIDC/JWT validation logs • Rate-limit test results • Allow/deny logs and exposure inventory • Secrets-in-vault evidence
Sub-EP (Quick Win)	EP-02.2	§12 Quick Win – V&V Traceability for one requirement. Stores the new Table B-6 row, tests, and artifacts.	• Completed Table B-6 row (PDF or screenshot) • Test plan/procedure • Platform test (synthetic/BAS) outputs • CI job/promotion-gate logs • Centralized telemetry entries (alerts, denials)
Sub-EP (Quick Win)	EP-02.3	§13 Quick Win – Change-Control Gate for configuration-as-code.	• Branch protection and gate config • Gate/pipeline diffs • Failed/approved run logs • Owner/approver records

Layer	EP Identifier	Purpose	Evidence Categories Included
Sub-EP	EP-02.4	Landing Zone Baseline (Parent).	• Guardrail checklist and evidence • Org-policy constraints • Tagging rules and tests • Initial posture snapshot
Sub-EP	EP-02.5	Supports future Sub-Standard ISAU-DS-CS-2001: Cloud IAM & Access Security.	• IAM role/permission exports • Access-review logs • MFA/JIT proof • Session-recording evidence • Identity-drift detection • Privilege-escalation test results
Sub-EP	EP-02.6	Supports future Sub-Standard ISAU-DS-CS-2002: Network Segmentation & East–West Control.	• VPC/VNet segmentation diagrams • Private-endpoint configs • Hub-and-spoke enforcement logs • mTLS enforcement evidence • BAS lateral-movement tests • Egress allowlist results
Sub-EP	EP-02.7	Supports future Sub-Standard ISAU-DS-CS-2003: Egress Control & Allowlisting.	• Egress policy-as-code • Allowed domain/IP lists • Negative egress test evidence • DNS egress filter logs • Usage/egress telemetry
Sub-EP	EP-02.8	Supports future Sub-Standard ISAU-DS-CS-2004: Data Protection & Key Management.	• AES-256 and TLS 1.3 enforcement logs • KMS rotation and key-use audit • Data classification/tagging proof • DLP egress policy evidence • Encrypted restore drills

Layer	EP Identifier	Purpose	Evidence Categories Included
Sub-EP	EP-02.9	Supports future Sub-Standard ISAU-DS-CS-2008: Centralized Logging, Telemetry & Evidence Production.	• Unified logging schema outputs • Authenticated time-sync evidence • Immutable retention configuration • SIEM/XDR correlation logs

Change Log and Revision History

Review Date	Changes	Committee	Action	Status
December 2025	Standards Revision	Standards Committee	Publication	Pending
November 2025	Standards Submitted	Technical Fellow Society	Peer review	Pending
October 2025	Standards Revision	Task Group ISAU-TG39-2024	Draft submitted	Complete
December 2024	Standards Development (Parent D01)	Task Group ISAU-TG39-2024	Draft complete	Complete