Kubernetes Security

From Supply Chain to Compliance

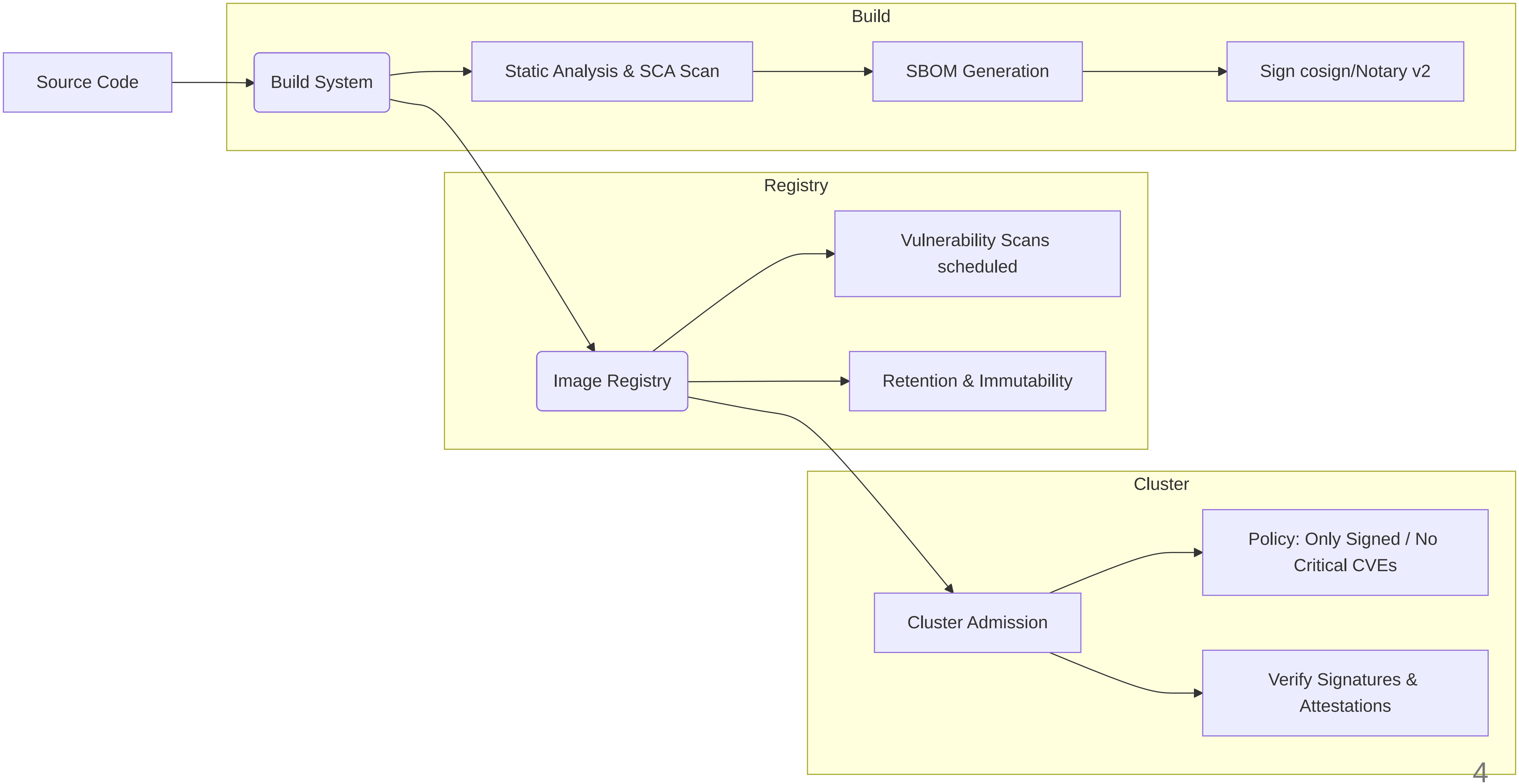
Agenda

1. Image Security (Scanning, Signing, SBOM)
2. Runtime Security (Falco, Sysdig)
3. Pod Sandboxing (gVisor, Kata)
4. Zero-Trust Networking in K8s
5. Auditing Kubernetes Clusters
6. Regulatory Compliance (HIPAA, GDPR, PCI)

1 Image Security

Threat Model & Objectives

- Threats: Malicious layers, embedded secrets, vulnerable libraries, tampering in transit, provenance spoofing
- Objectives: Integrity, provenance, minimal attack surface, repeatability, transparency
- Controls: Scanning (pre-/post-build), signing/verification, SBOMs, policy-as-code, hermetic builds, provenance attestations


Source Code
Build
Build System
Static Analysis & SCA Scan
SBOM Generation
Sign cosign/Notary v2
Registry
Image Registry
Vulnerability Scans scheduled
Retention & Immutability
Cluster
Cluster Admission
Policy: Only Signed / No Critical CVEs
Verify Signatures & Attestations

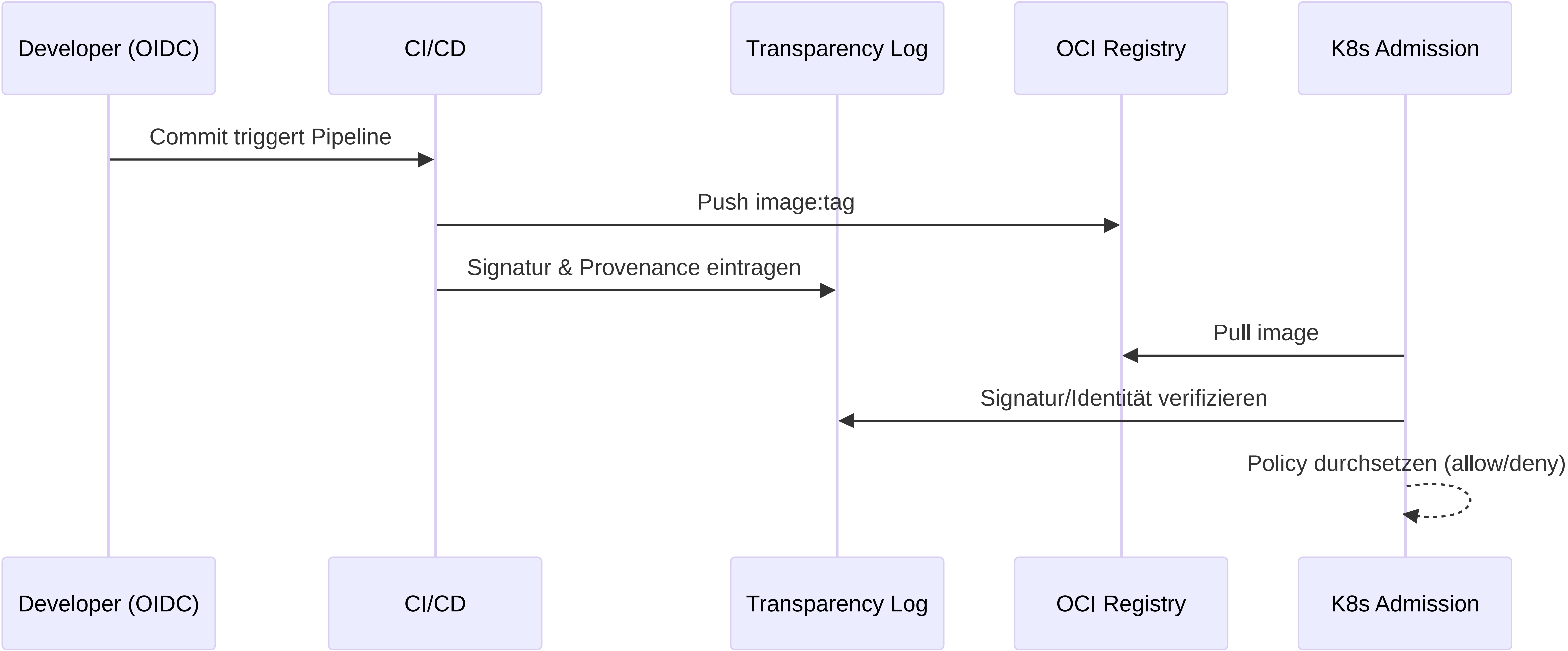
Scanning (SCA & Secrets)

- SCA: Detect CVEs across OS and application layers
- Depth: OS packages (apk/apt/yum), language deps (npm/pip/Go), base images
- Secrets & Misconfig: API keys, SSH keys, weak config (root, CAP_SYS_ADMIN)

> Principle: Fail closed for critical severities; waiver with justification only.

Signing & Verification

- Artifact signing: Sigstore/cosign, Notary v2 — detached signatures in OCI registry
- Keyless patterns: OIDC identity → transparency log (Rekor)
- Verify at admission: Reject unsigned or untrusted identities
- Attestations: Build provenance (SLSA/in-toto) proves how it was built


Developer (OIDC)
CI/CD
Transparency Log
OCI Registry
K8s Admission
Commit triggert Pipeline
Push image:tag
Signatur & Provenance eintragen
Pull image
Signatur/Identität verifizieren
Policy durchsetzen (allow/deny)
Developer (OIDC)
CI/CD
Transparency Log
OCI Registry
K8s Admission

SBOMs (Software Bill of Materials)

- Standards: SPDX, CycloneDX
- Content: Packages, versions, licenses, dependency graph
- Use: Risk assessment, license compliance, vulnerability mapping, change control
- Lifecycle: Generate in CI → store with image → verify presence at deploy

> Pair SBOM with provenance attestation; SBOM without provenance is weaker.

Policy-as-Code at the Gate

- Admission (OPA Gatekeeper / Kyverno) enforces:
  - Only signed images, approved base images, no latest, block high CVEs
  - Required labels/annotations (owner, data class), SBOM required
- Registry controls: Immutability, retention, quarantine on failure
- Runtime defaults: Minimal capabilities & read-only filesystem from the outset

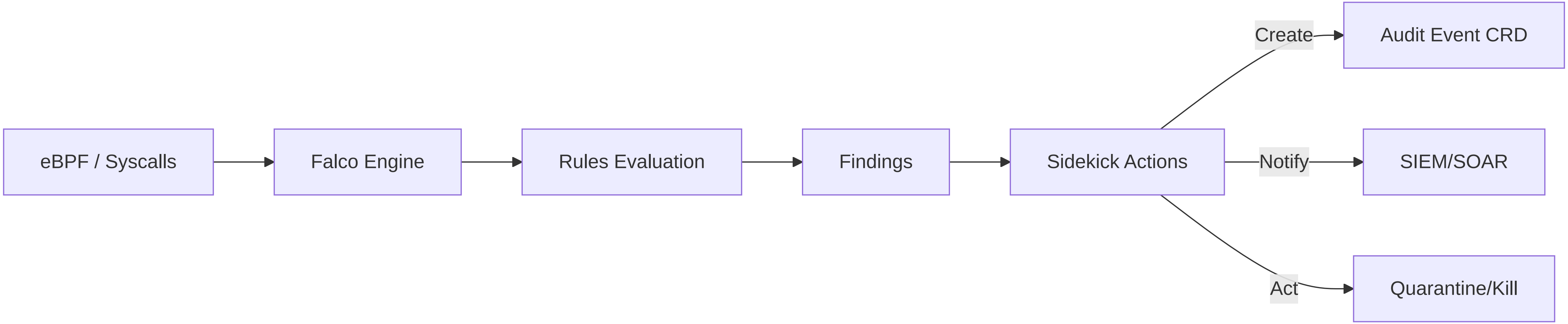
2 Runtime Security

Conceptual Model

- Observability-first: Syscall-, kernel-, and network-level signals (often via eBPF)
- Detect & Respond: Suspicious behaviors (crypto-mining, breakout attempts)
- Context: K8s metadata (pod, namespace, labels) + cloud/host signals
- Controls: Detection rules, automated responses (kill, isolate, notify)

Falco & Sysdig (theory)

- Falco: CNCF project; rule engine on syscalls/eBPF; Kubernetes-aware outputs
- Rules: Human-readable conditions (e.g., write to /etc from a container)
- Responders: Falco Sidekick → SIEM, Slack, K8s actions
- Sysdig (platform concept): Adds posture, forensics, compliance packs


eBPF / Syscalls
Falco Engine
Rules Evaluation
Findings
Sidekick Actions
Create
Audit Event CRD
Notify
SIEM/SOAR
Act
Quarantine/Kill

Runtime Baselines & Hardening

- Least privilege: Drop Linux caps, seccomp, AppArmor/SELinux profiles
- FS & process: Read-only rootfs; no package managers in prod images
- Network: Default-restricted egress
- Drift control: Alert on execution of unknown binaries or shell spawns

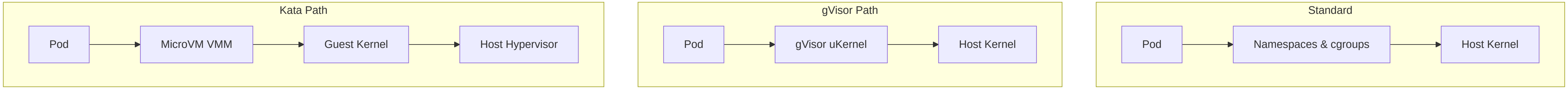
3 Pod Sandboxing

Why Sandbox?

- Threats: Kernel escape, noisy neighbor, untrusted multi-tenancy
- Goal: Stronger isolation boundary than namespaces/cgroups alone
- Trade-off: Compatibility & performance vs. isolation strength

gVisor vs. Kata (models)

- gVisor: User-space kernel intercepts syscalls; reduces host-kernel exposure
- Kata Containers: Lightweight VMs (hardware virtualization) per pod
- Decision drivers: Latency/throughput, syscall coverage, GPU/TPM needs, node density, workload type


Kata Path
Pod
MicroVM VMM
Guest Kernel
Host Hypervisor
gVisor Path
Pod
gVisor uKernel
Host Kernel
Standard
Pod
Namespaces & cgroups
Host Kernel

Integration (theory)

- RuntimeClass: Map workloads to runtimes (high-risk → Kata; balanced → gVisor)
- Node pools: Taint/label nodes for sandbox workloads
- Security posture: Combine with minimal caps, seccomp, and strong network policies

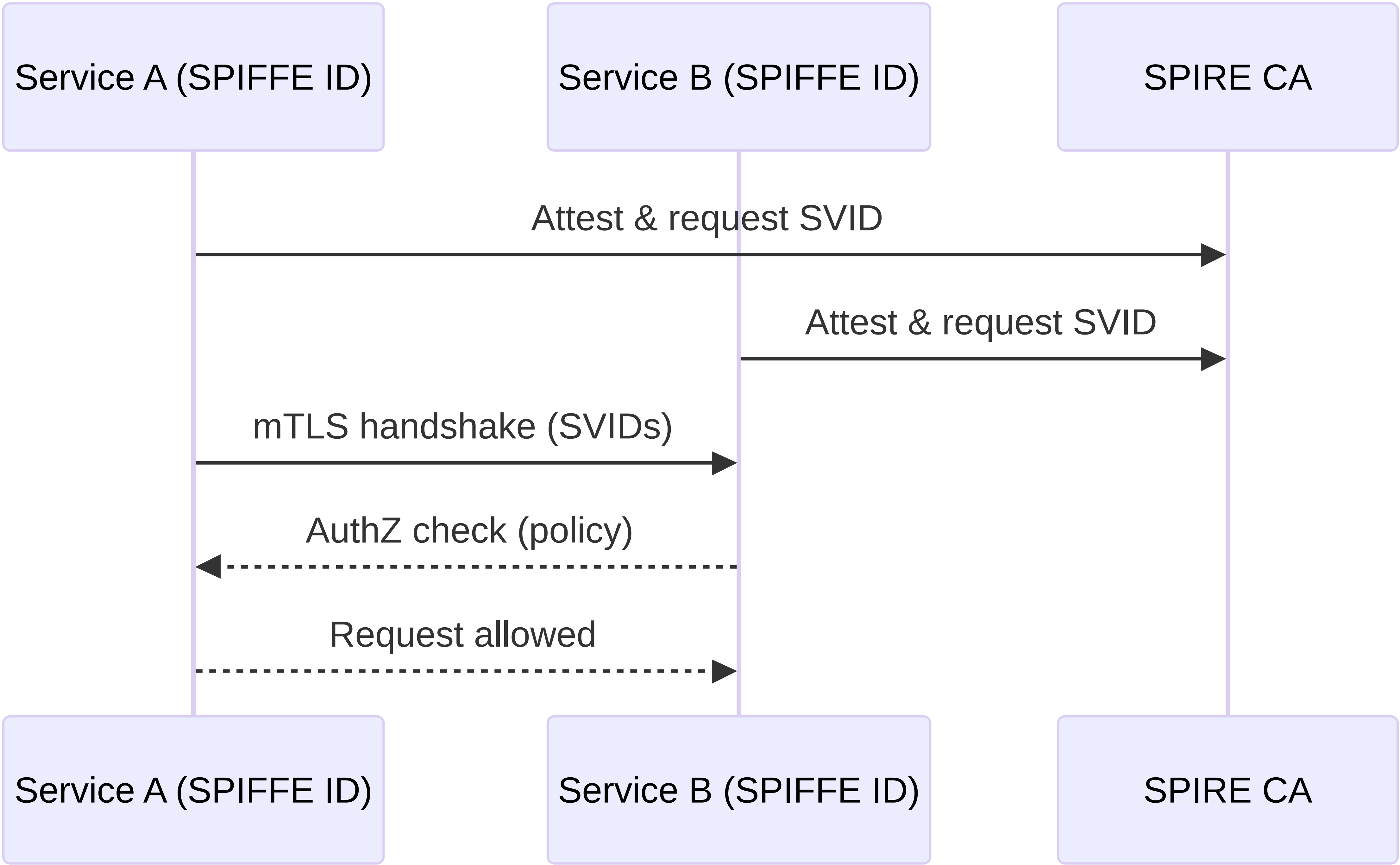
4 Zero-Trust Networking in K8s

Principles

- Never trust network location; verify identity per request
- Strong identities: SPIFFE IDs (per workload), short-lived certificates
- AuthZ everywhere: Policies by identity & intent (Layer 7 where possible)
- Least-privilege paths: Deny-by-default east–west; controlled egress

Identity & mTLS

- Identity issuance: SPIRE attests node/workload → SVID
- mTLS: Automatic cert rotation; workload-to-workload encryption
- Service meshes: Istio/Linkerd enforce mTLS, provide policy & telemetry


Service A (SPIFFE ID)
Service B (SPIFFE ID)
SPIRE CA
Attest & request SVID
Attest & request SVID
mTLS handshake (SVIDs)
AuthZ check (policy)
Request allowed
Service A (SPIFFE ID)
Service B (SPIFFE ID)
SPIRE CA

Network Policies & Egress

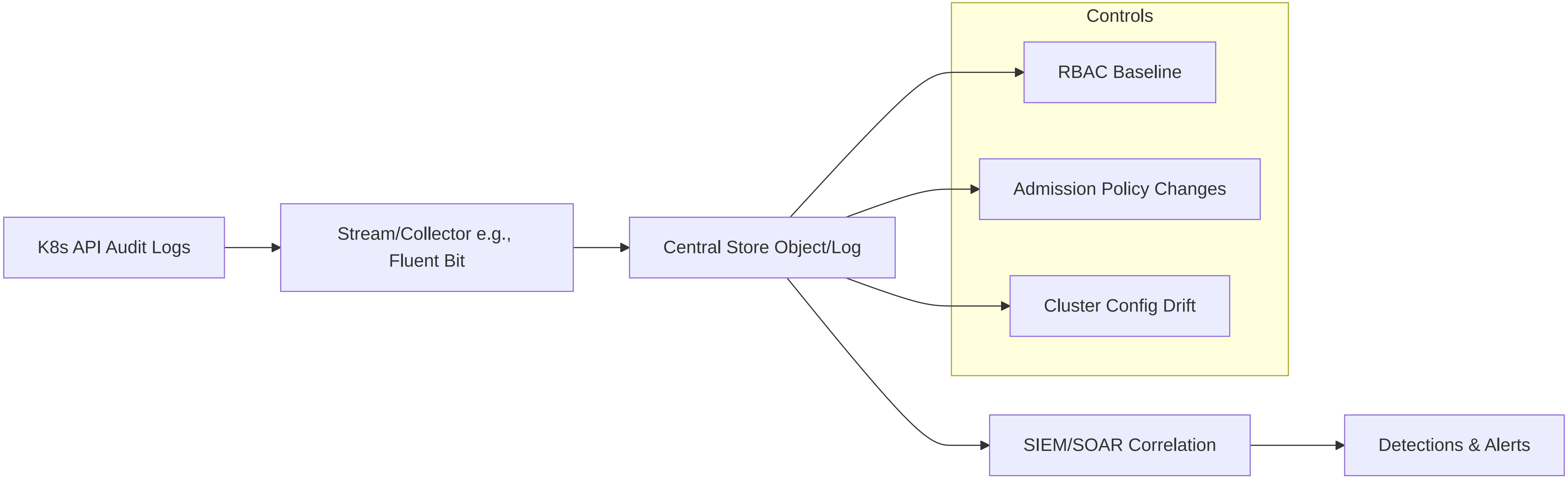
- K8s NetworkPolicy: Namespaced L3/L4 allow-lists; deny by default
- CNI extensions (e.g., Cilium): L7 policies, DNS-aware egress, identity-based rules
- Egress controls: NAT gateways, egress policies, proxy enforcement; DNS policy as choke point

> Combine mesh AuthN/Z with CNI L3/L7 for defense-in-depth.

5 Auditing Kubernetes Clusters

What to Audit & Why

- API server activity: who did what, where, when
- Control plane config: policy changes, RBAC, admission configs
- Node & workload events: crashes, restarts, image pulls, privilege changes
- Objective: Forensics, incident response, compliance evidence


K8s API Audit Logs
Stream/Collector e.g., Fluent Bit
Central Store Object/Log
Controls
RBAC Baseline
Admission Policy Changes
Cluster Config Drift
SIEM/SOAR Correlation
Detections & Alerts

Audit Policy Design

- Stages: RequestReceived, ResponseStarted, ResponseComplete, Panic
- Selectors: By user/group, namespaces, resources, verbs
- Retention & immutability: WORM storage, tamper-evident logs
- PII minimization: Redact sensitive objects; balance forensics vs. privacy

6 Regulatory Compliance in Kubernetes

Shared Responsibility

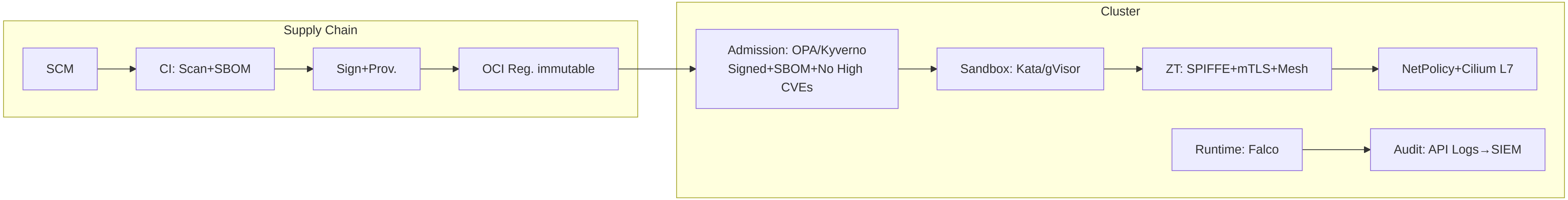
- Regimes: HIPAA (health), GDPR (privacy), PCI DSS (payments)
- Layers: Cloud/IaaS, Kubernetes platform, workloads, data/processes
- Approach: Map controls → implement technical guardrails → collect evidence

Control Area	HIPAA	GDPR	PCI DSS	K8s/Cloud Mechanisms (theory)
Access Control	Unique user IDs, authN	Lawful basis, access rights	Strong auth, least privilege	RBAC, SSO/OIDC, short-lived tokens
Transmission Security	Encrypt in transit	Art. 32 security	Encrypt cardholder data	mTLS (mesh), TLS to ingress/egress
Data at Rest	Encryption & key mgmt	Privacy by design	Protect stored PAN	KMS-backed PV encryption, secret managers

Control Area	HIPAA	GDPR	PCI DSS	K8s/Cloud Mechanisms (theory)
Isolation	Segregate ePHI	Data minimization	Segment CDE	Namespaces, network policies, sandbox runtimes
Change Mgmt	Config mgmt	DPIA for risk	Change control	GitOps, policy-as-code, provenance attestations
Incident Response	Breach notif.	72h notification	IR procedures	Falco detections, playbooks, forensics-ready

Data Protection & Privacy by Design

- Data classification: Tag workloads & resources (owner, data class)
- Minimization: Only necessary data in pods; use ephemeral storage where possible
- Residency & locality: Node/zone pinning, regional services
- Data subject rights (GDPR): Discoverability, deletion workflows, logging of access


Supply Chain
SCM
CI: Scan+SBOM
Sign+Prov.
OCI Reg. immutable
Cluster
Admission: OPA/Kyverno Signed+SBOM+No High CVEs
Sandbox: Kata/gVisor
ZT: SPIFFE+mTLS+Mesh
NetPolicy+Cilium L7
Runtime: Falco
Audit: API Logs→SIEM

Maturity Roadmap (high level)

- Foundational: SBOMs, signing, basic NetworkPolicies, RBAC hygiene
- Intermediate: Admission enforcement, mTLS via mesh, runtime baselining
- Advanced: Sandbox runtimes, identity-based L7 policies, provenance (SLSA), automated evidence collection

Key Takeaways

- Shift-left + enforce-right: Supply chain integrity + strict admission
- Assume breach: Runtime detection & sandboxing reduce blast radius
- Identity > IP: Zero-Trust with SPIFFE mTLS & L7 AuthZ
- Prove it: Audit design, immutable logs, policy-as-code for compliance

References & Further Reading

- Kubernetes Docs: Security, Admission, Audit
- CNCF Projects: Falco, SPIFFE/SPIRE, Notary, OPA, Kyverno
- CIS Kubernetes Benchmark
- SLSA Framework & in-toto attestations
- SPDX / CycloneDX SBOM standards
- Cilium (L3–L7 networking & Hubble observability)