Kyverno

Flight Kyverno Five - Boarding to Production

Taming Kubernetes chaos with the Kyverno Five

Neha Jaju
Senior Software Engineer, Nirmata

Clusters are like busy airports

Thousands of “workloads” arrive and depart every minute. Without a crew coordinating it, things go wrong — fast:

Unsigned images slip in

anyone's container can board

Missing labels & defaults

nothing is tagged or owned

New namespaces, no guardrails

tenants land with nothing in place

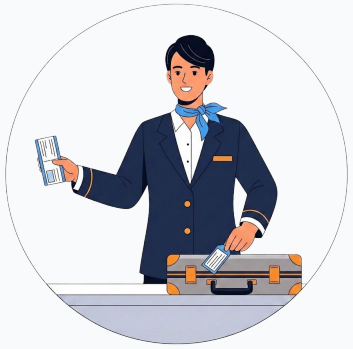
Stale jobs & configs pile up

the gates never get cleared

Kyverno: Your specialized ground crew for Kubernetes safety.

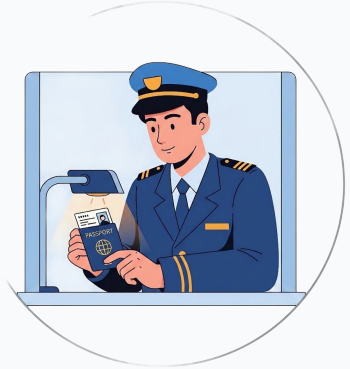
An airport runs on specialist crews

No single hero. A brigade — each owning one job, handed off in sequence from check-in to take-off and back.



Check-in

tag & complete

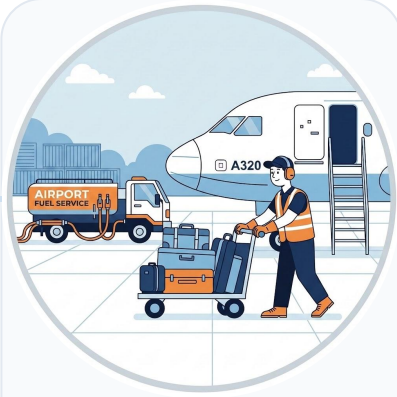


Security

verify identity

Boarding gate

allow or deny



Ground crew

prep every flight

Turnaround

clear the gate

Each of these crews maps to exactly one Kyverno policy type. Let's walk the journey.

STEP 1 · CHECK-IN

MutatingPolicy

Kyverno

Check-in Desk

GROUND HANDLING

Tags your bags, prints your boarding pass, assigns your seat — you leave the desk flight-ready.

Shapes each resource on the way in — adds labels, defaults and sidecars.

add-default-labels.yaml

```yaml
apiVersion: policies.kyverno.io/v1
kind: MutatingPolicy
metadata:
  name: add-default-labels
spec:
  matchConstraints:
    resourceRules:
    - resources: ["deployments"]
      operations: ["CREATE","UPDATE"]
  mutations: # add what's missing
  - patchType: JSONPatch
    jsonPatch:
      expression: >-
        [JSONPatch{op:"add",
          path:"/metadata/labels/team", value:"pay"}]
```

↳ **Fix it for them — don't reject them.**

STEP 2 · SECURITY

`ImageValidatingPolicy`

Kyverno

Security & Passport

IDENTITY & SCREENING

Checks your passport is genuine and you are who you say you are. No valid ID, no entry.

Verifies image signatures and provenance before anything runs.

verify-signatures.yaml

```yaml
apiVersion: policies.kyverno.io/v1
kind: ImageValidatingPolicy
metadata:
  name: verify-signatures
spec:
  matchImageReferences: # which images
  - glob: "ghcr.io/myorg/*"
  attestors: # who we trust
  - name: cosign
    cosign:
      keyless:
        identities:
        - issuer: "https://token.actions..."
validations: # must be signed
  - expression: >-
      images.containers.map(i,
        verifyImageSignatures(i,[attestors.cosign])>0)
      .all(v, v)
```

↳ **Prove the image is who it claims to be.**

STEP 3 · BOARDING GATE

ValidatingPolicy

Kyverno

Boarding Gate

THE FINAL CHECK

Valid boarding pass for this flight? You're on. If not, you're turned away at the door.

The gatekeeper — allow or deny the resource against the rules.

require-labels.yaml

```yaml
apiVersion: policies.kyverno.io/v1
kind: ValidatingPolicy
metadata:
name: require-labels
spec:
matchConstraints:
resourceRules:
- resources: ["pods"]
operations: ["CREATE","UPDATE"]
validations: # the gate
- expression: >-
has(object.metadata.labels.app) &&
has(object.metadata.labels.version)
message: "Pods need app + version labels"
```

↳ **The guardrail at the door.**

STEP 4 · GROUND CREW

GeneratingPolicy

Kyverno

Ground Crew

RAMP & FLIGHT PREP

Every new flight gets fuel, catering and baggage carts staged — automatically, before it's needed.

On each new namespace, provision its companions — secrets, quotas, NetworkPolicies.

clone-pull-secret.yaml

```yaml
apiVersion: policies.kyverno.io/v1
kind: GeneratingPolicy
metadata:
  name: clone-pull-secret
spec:
  matchConstraints: # trigger: new ns
    resourceRules:
    - resources: ["namespaces"]
      operations: ["CREATE"]
  generate: # provision it
  - expression: generator.Apply(variables.ns, [variables.src])
```

↳ Every new tenant born with guardrails.

STEP 5 · TURNAROUND CREW

`DeletingPolicy`

Turnaround Crew

CABIN & GATE RESET

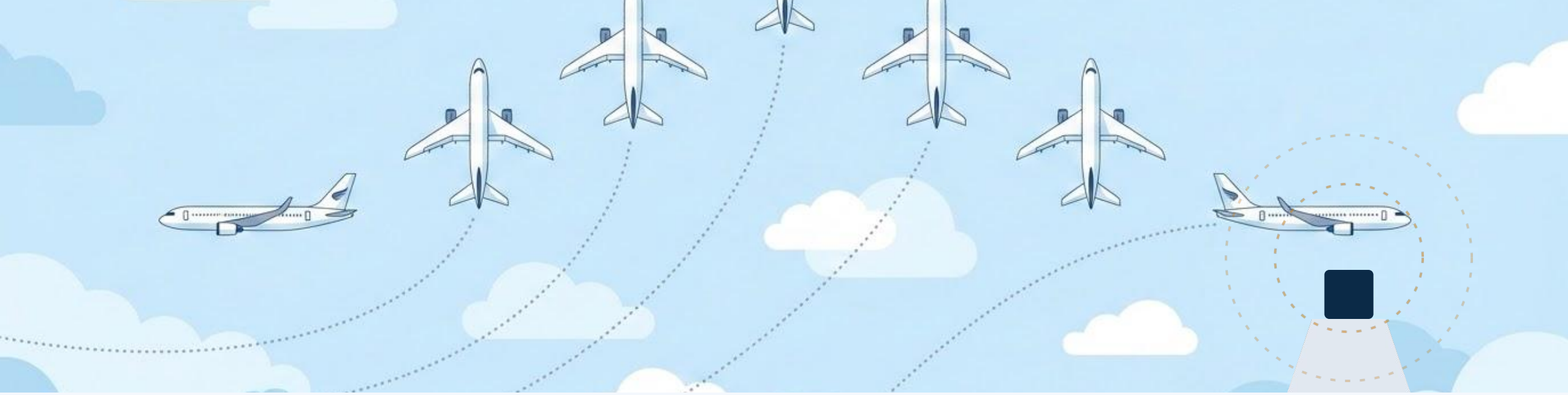
Once a flight lands, they clear the cabin and free the gate for the next departure.

On a schedule, retire what's done — old jobs, stale configs, expired clutter.

cleanup-completed-jobs.yaml

```yaml
apiVersion: policies.kyverno.io/v1
kind: DeletingPolicy
metadata:
  name: cleanup-completed-jobs
spec:
  schedule: "0 1 * * *" # nightly at 1 AM
  matchConstraints:
    resourceRules:
    - resources: ["jobs"]
      operations: ["*"]
  conditions: # only finished ones
  - name: isOld
    expression: >-
      time.now() - timestamp(
        object.metadata.creationTimestamp)
        > duration("72h")
```

↳ **Stop paying for zombies.**



IN HARMONY

Flying in Formation with Kubernetes

The Kyverno Five don't fight the platform — they fly in formation with it.

Kyverno

One language

Written in CEL, the language Kubernetes itself adopted.

Same runways

Validating & Mutating policies compile to native admission controllers.

No turf war

They work with RBAC and the API server, not around them.

From Check-in to Turnaround: Kyverno for the Full Resource Lifecycle

Mutating

Modifies resources to meet your standards.

Image Validating

Ensures only trusted images are deployed.

Validating

Checks and blocks non-compliant resources.

Generating

Automates creation of companion resources.

Deleting

Cleans up stale or unneeded resources.

"Kyverno is the airport operations team making sure every resource is trusted, compliant, correctly tagged, supported, and eventually cleared out."

Kyverno

Thank you

Thanks for flying with Kyverno.