Securing the Invisible Hands: Policy and Guardrails for AI-Driven Infrastructure

Annie Talvasto

- Sr Manager, Product Marketing & Community at Upbound
- CNCF Ambassador
- Azure MVP

- CloudGossip podcast & TechCraft Show
- Previously: CloudNativeLive & Program Chair for Secure AI Summit

Karl Ots

- Head of Cloud Security at EPAM
- Microsoft Security MVP

- Author of Securing Microsoft Azure OpenAI (Wiley)
- Cloud Gossip podcast

@AnnieTalvasto

@KarlOts.com

How do we secure infrastructure that AI agents create without human oversight?

AI-Driven Infrastructure Risks

No Human Gatekeepers

AI makes rapid infra changes without manual review.

Misconfigurations or vulnerabilities can slip through.

Speed & Scale of Change

Autonomous changes happen too fast for human intervention if something goes wrong.

Propagating Impact

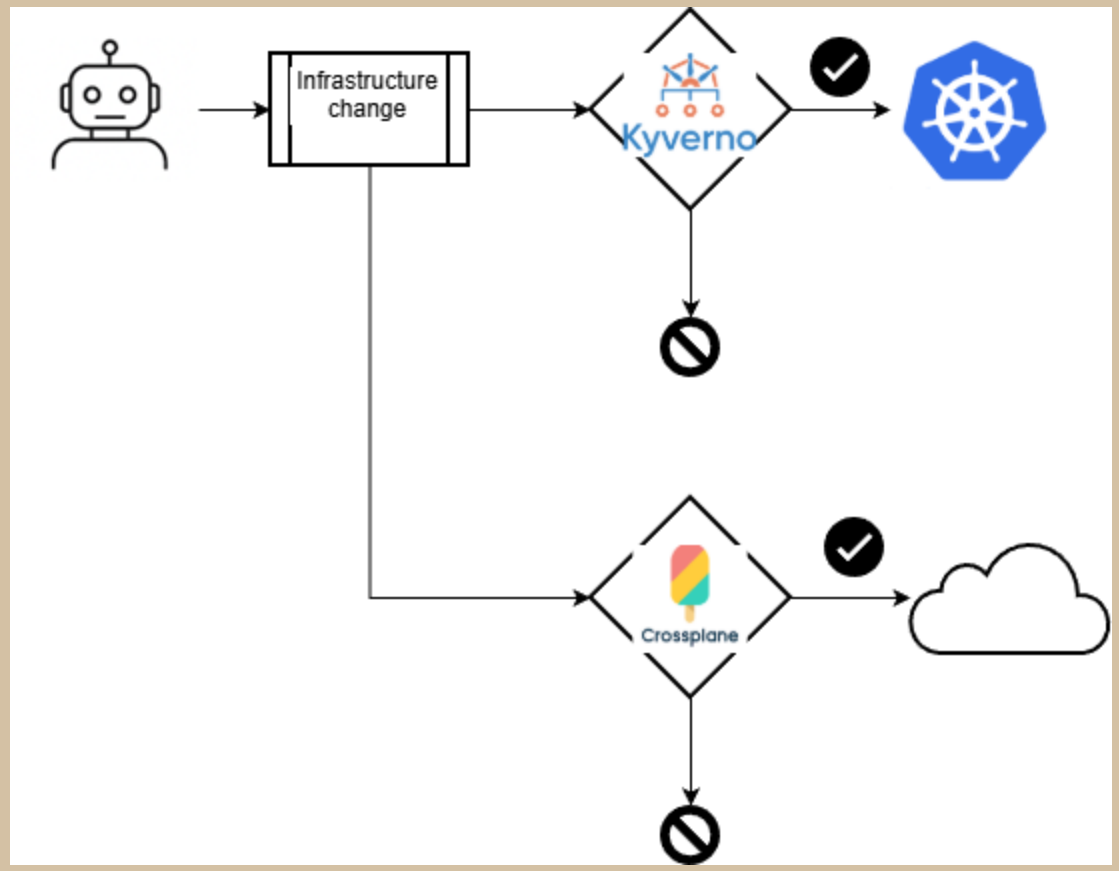
Misconfigurations or malicious actions by compromised agents can propagate widely before detection.

Need for guardrails

We require automated policies & monitors that work at machine speed to catch issues **in real-time.**

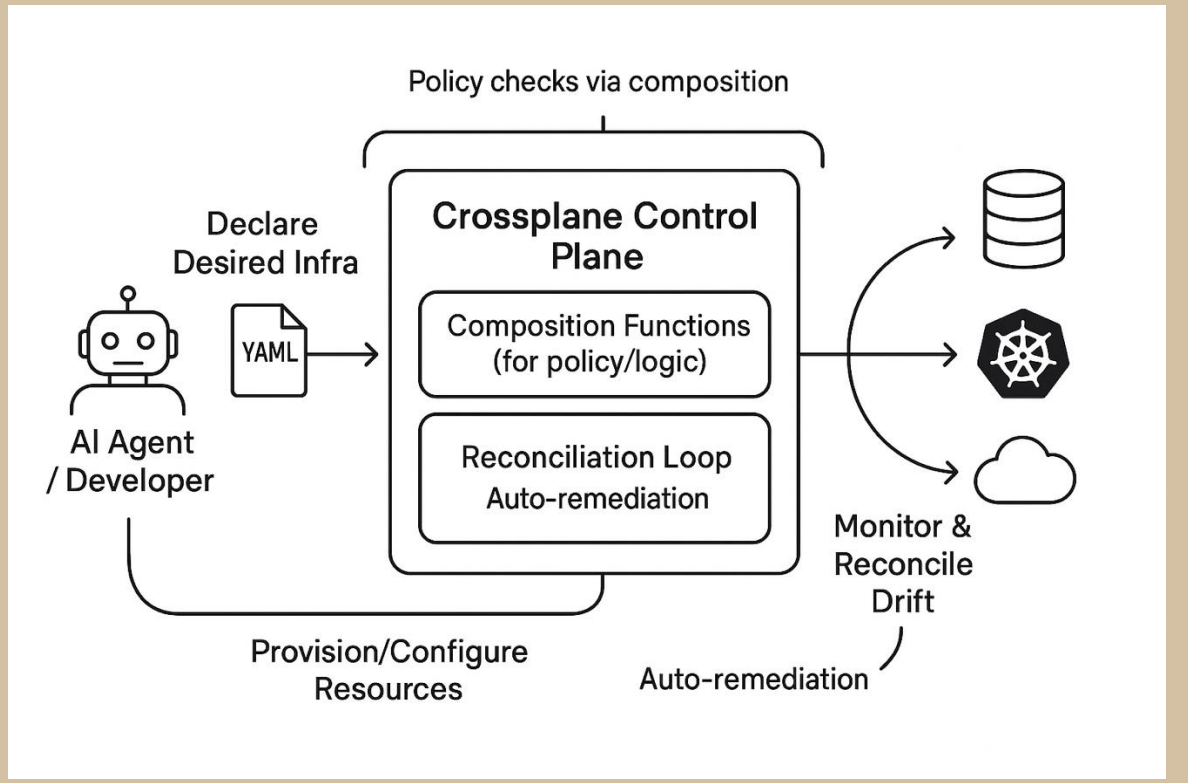
Policy as Code guardrails

- Policy as Code: Define security rules as declarative policies, preventing misconfigurations or non-compliant resources from ever being created
- Kubernetes Policies: e.g. Kyverno or OPA enforce rules on cluster changes before they take effect.
- Crossplane Composition Policies: extend policy scope from k8s cluster to cloud resources.



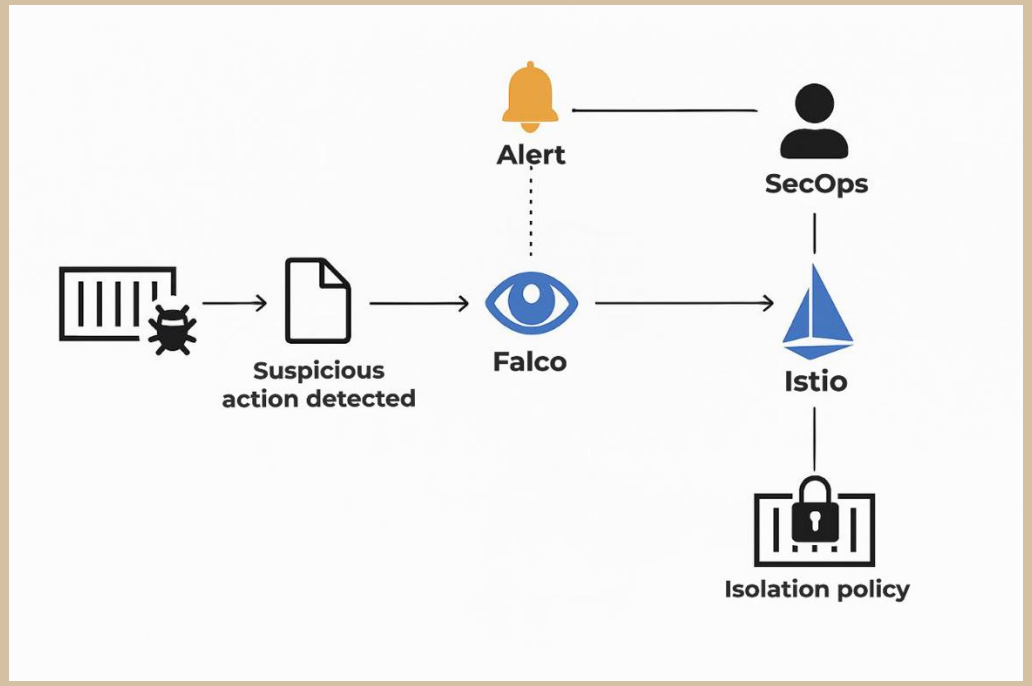
Crossplane 2.0: Declarative Control Plane for Safe Automation

- **Crossplane**: Open-source control plane (Kubernetes-based) to manage cloud infrastructure as code (YAML) across providers.
- **Composition Functions** (New in 2.0): Insert custom logic/policies into provisioning. (E.g., auto-attach mandatory security configs, or require approval for certain requests.)
- **Continuous Reconciliation**: Crossplane's controllers constantly compare declared state vs. actual state. If an AI-driven change causes drift (deviation), Crossplane detects & fixes it within seconds (auto-remediation).



Real-Time Monitoring and Response

- **Continuous Monitoring:** Even with good policies, runtime issues can occur. Use tools like **Falco** to watch running resources for suspicious behavior.
- **Automated Response:** Integrate alerts with actions: e.g., **Istio/Network Policies** to isolate a compromised service instantly.
- **Drift Detection:** Monitor config drift continuously. If any config deviates (e.g., AI agent changes something out-of-band), flag it or auto-roll back.



Architectural Patterns for AI-Native Security

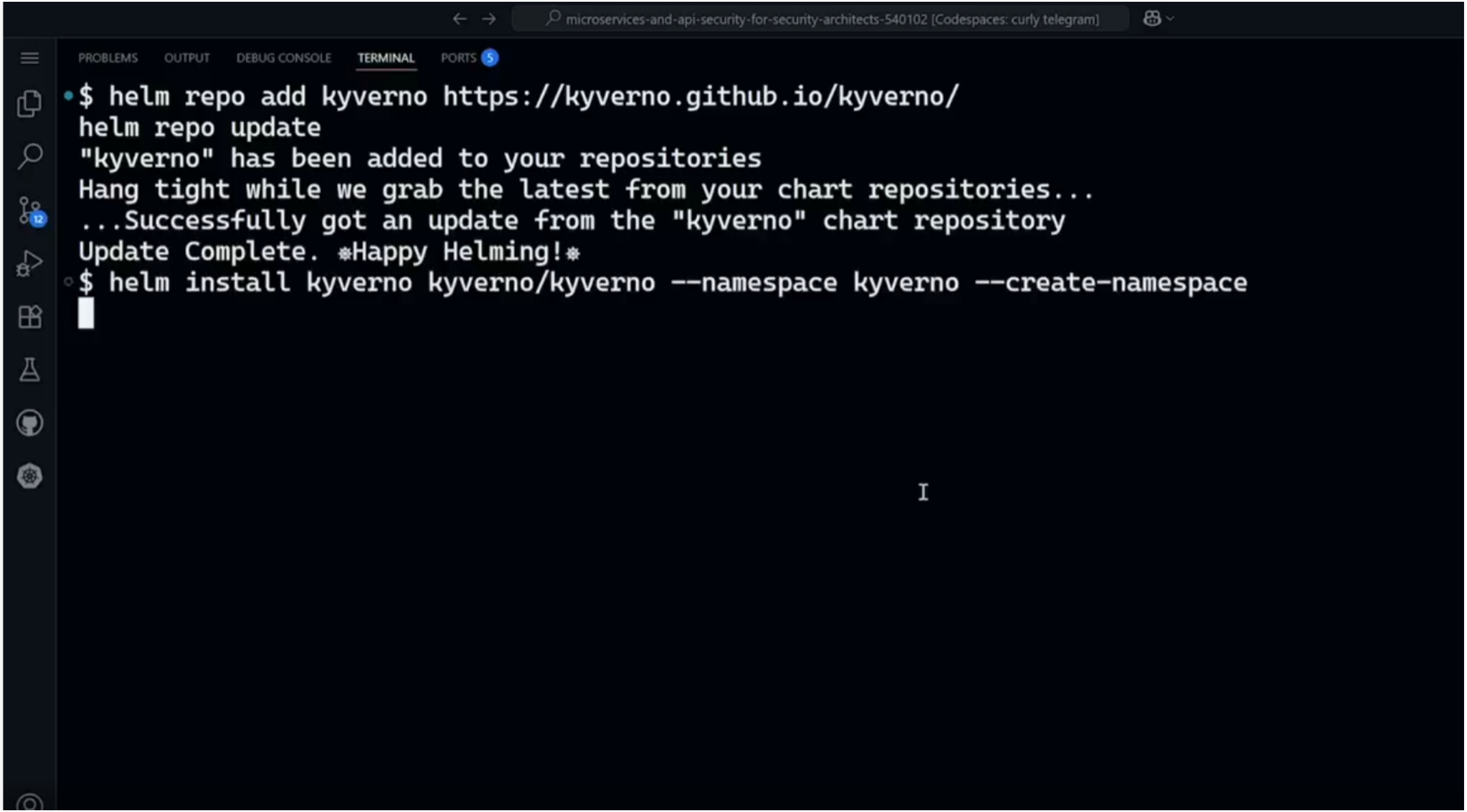
Shift security left

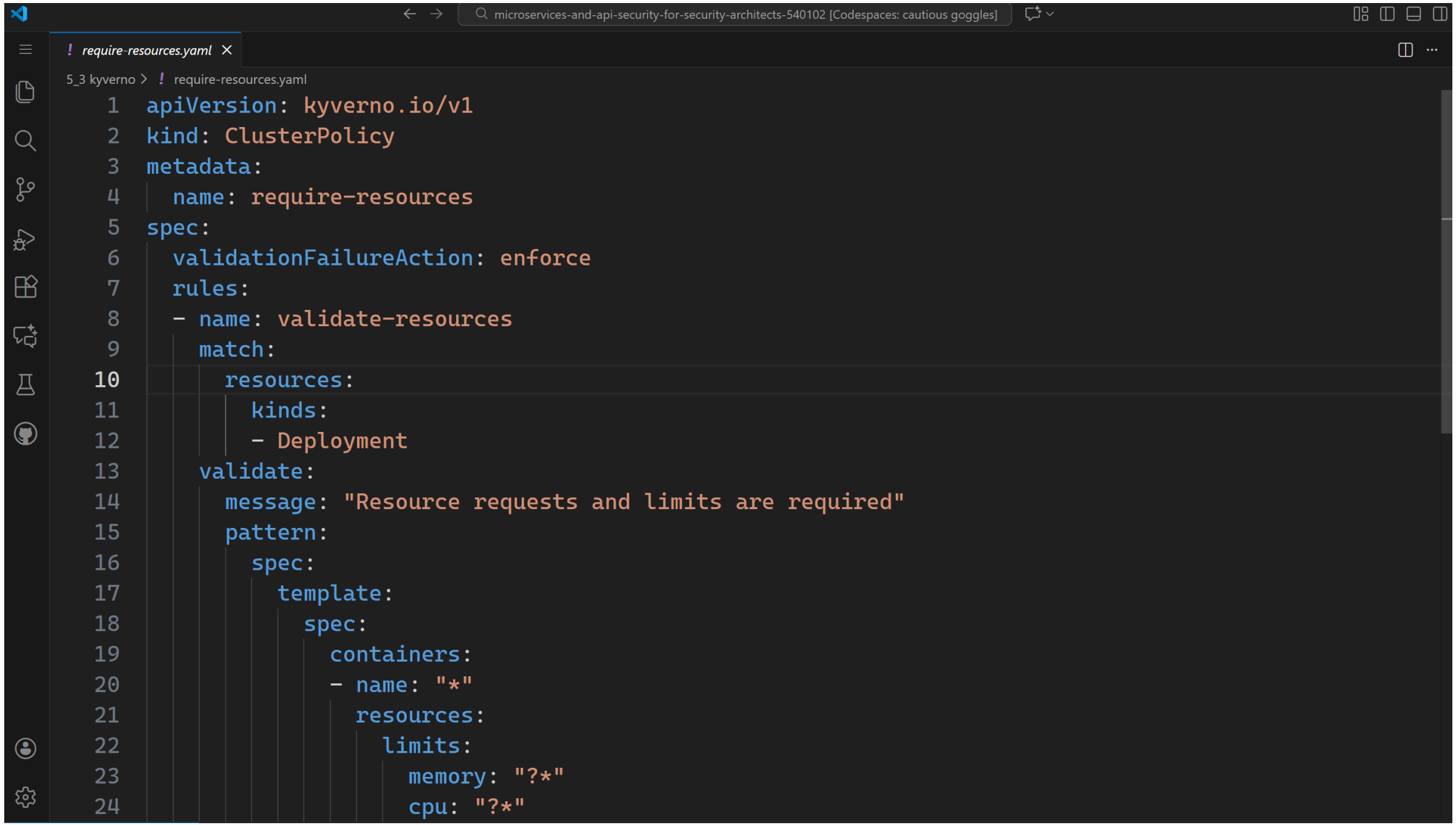
GitOps for the win

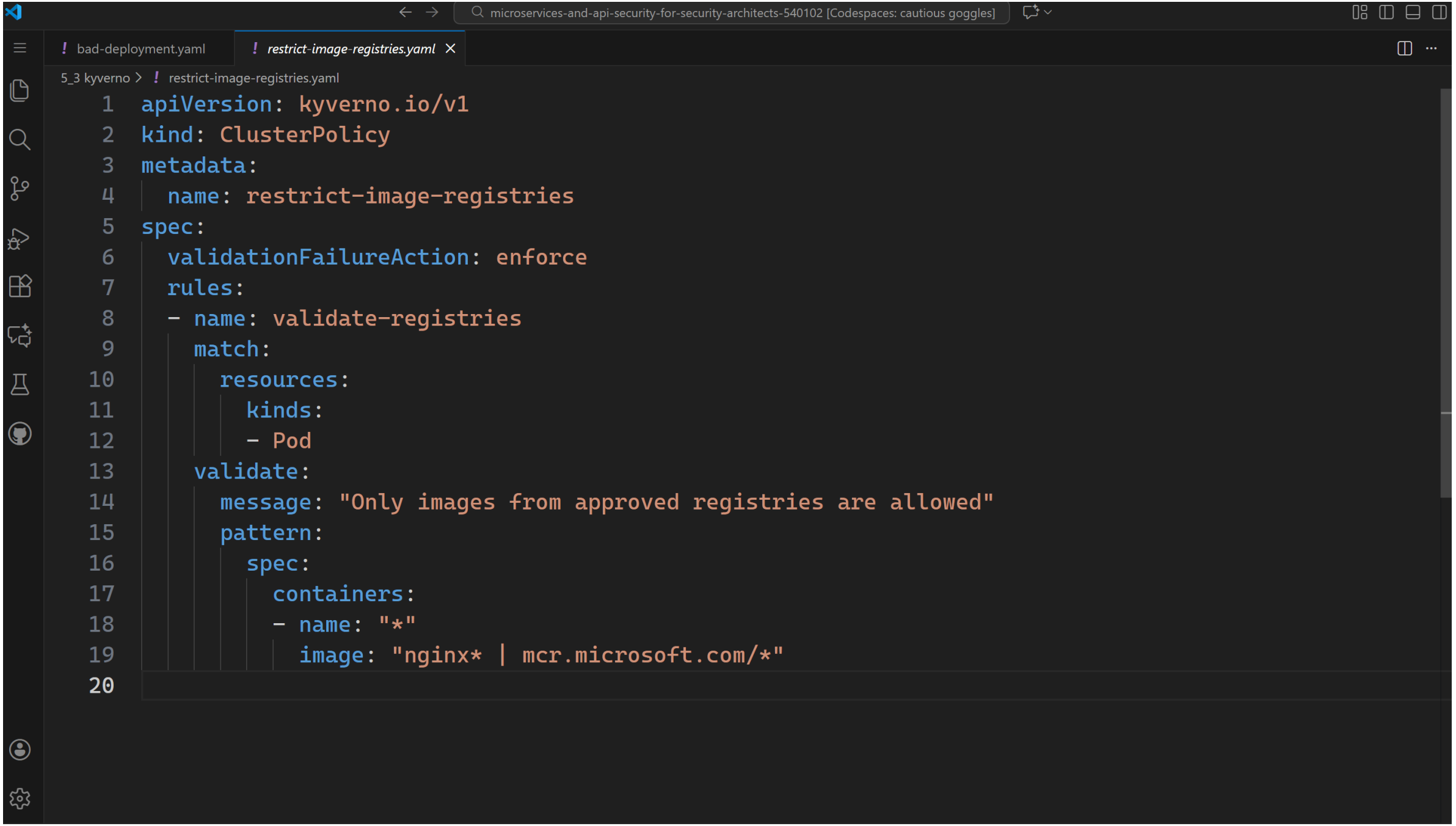
Minimize agentic IAM

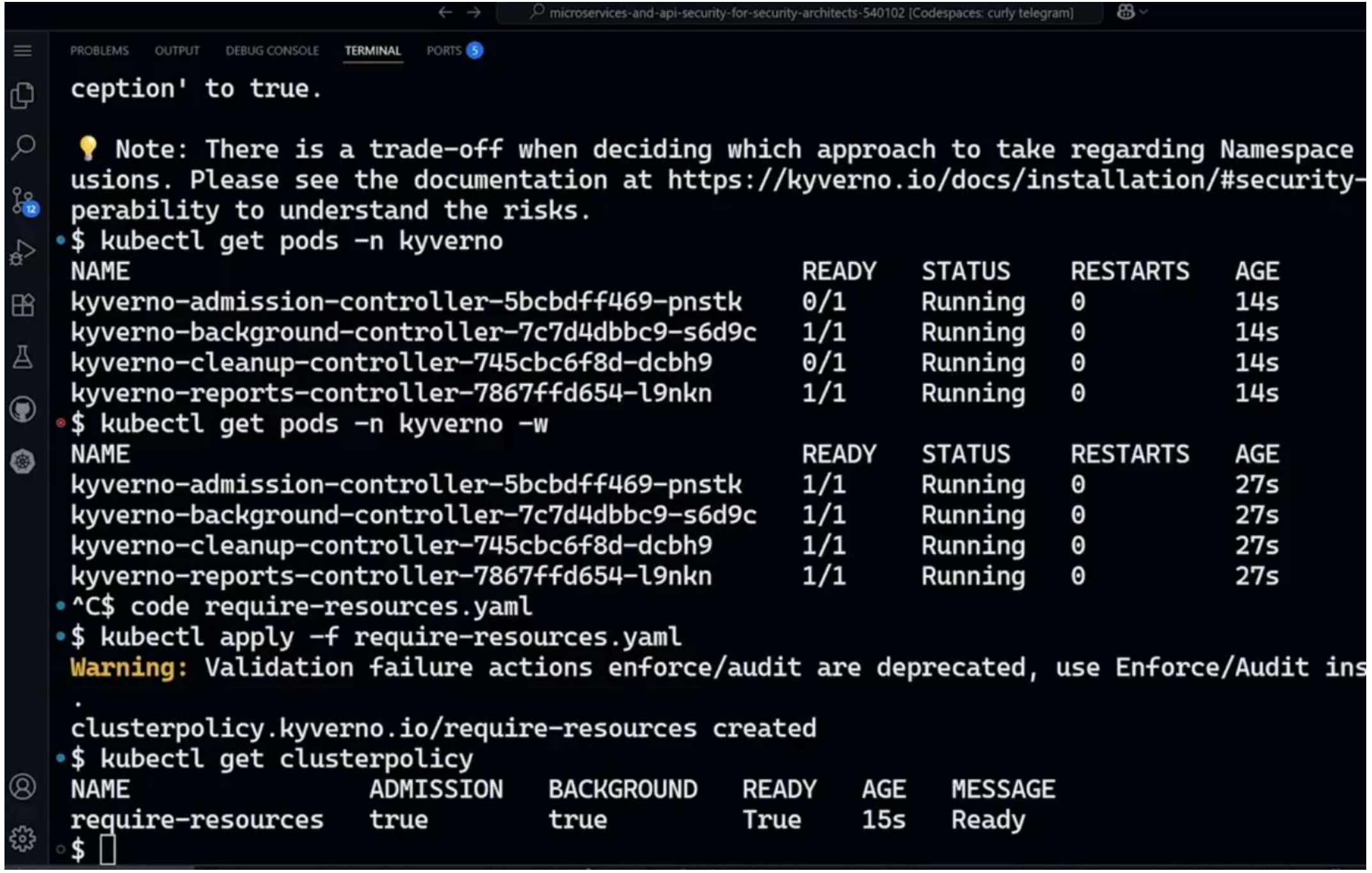
Defense in Depth

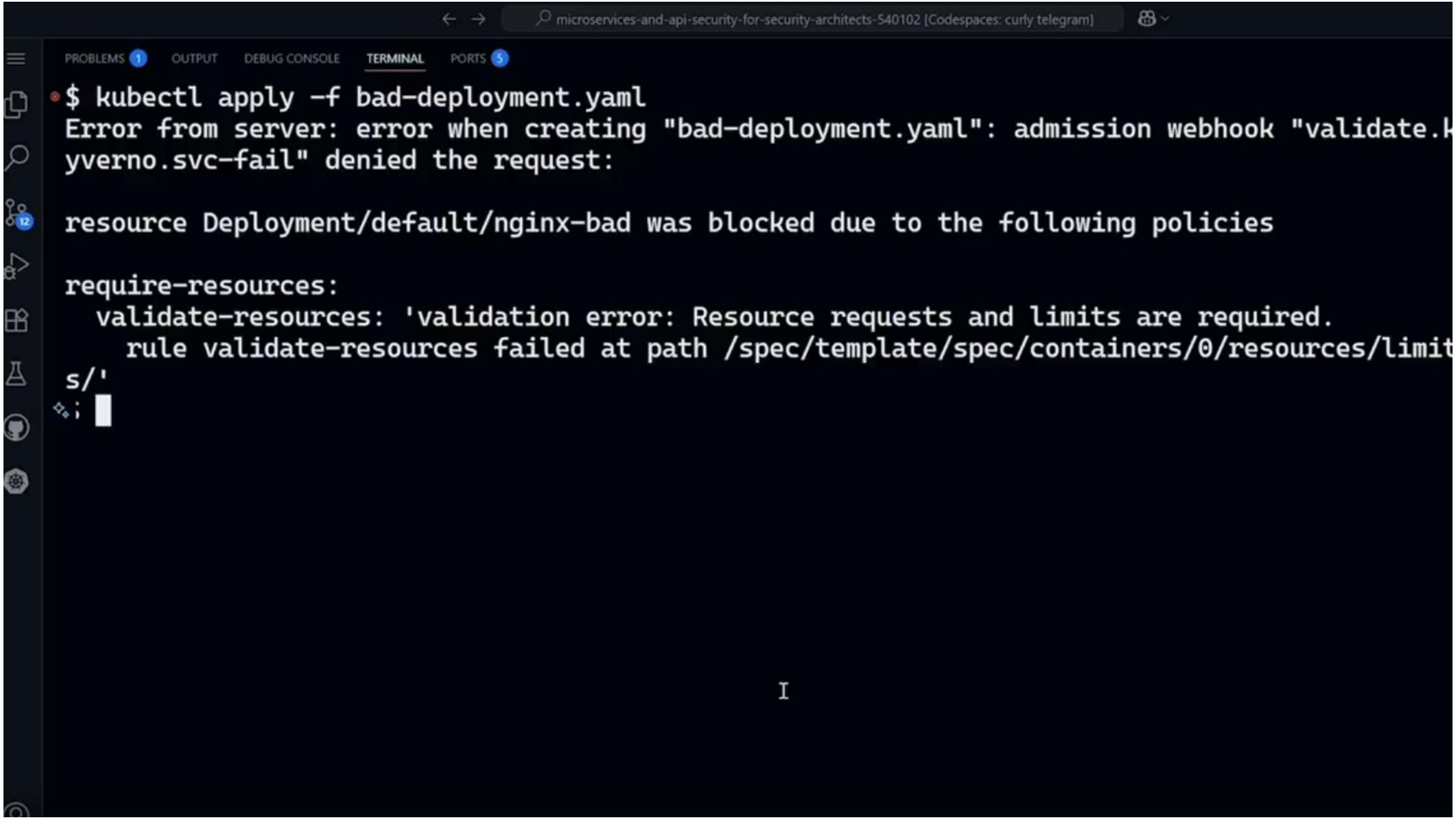
Demo: Guardrails and Policy as Code in action


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
$ helm repo add kyverno https://kyverno.github.io/kyverno/
helm repo update
"kyverno" has been added to your repositories
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "kyverno" chart repository
Update Complete. ⎈Happy Helming!⎈
$ helm install kyverno kyverno/kyverno --namespace kyverno --create-namespace

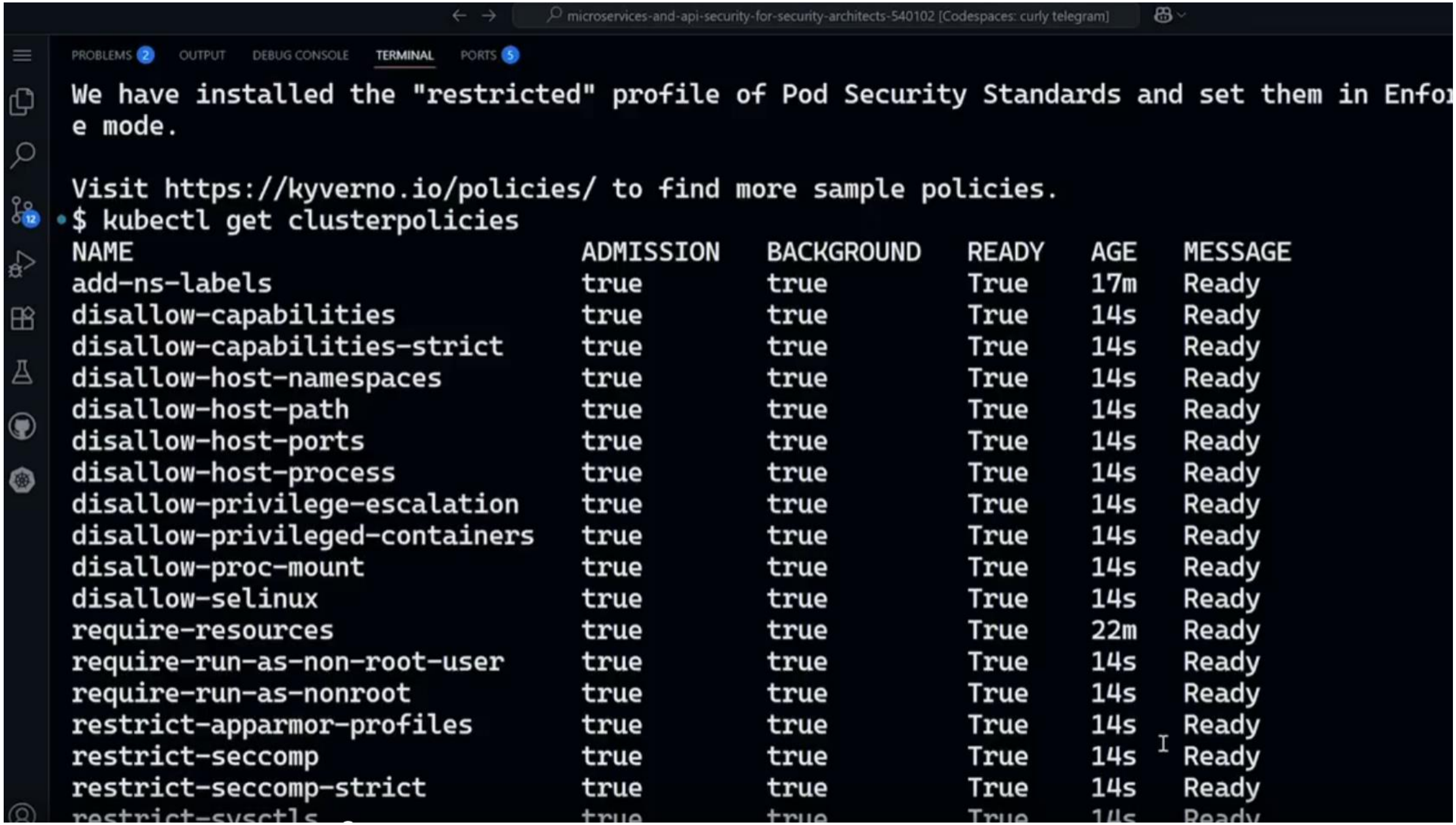

microservices-and-api-security-for-security-architects-540102 [Codespaces: cautious goggles]
require-resources.yaml
5_3 kyverno > require-resources.yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
name: require-resources
spec:
validationFailureAction: enforce
rules:
- name: validate-resources
match:
resources:
kinds:
- Deployment
validate:
message: "Resource requests and limits are required"
pattern:
spec:
template:
spec:
containers:
- name: "*"
resources:
limits:
memory: "?*"
cpu: "?*"

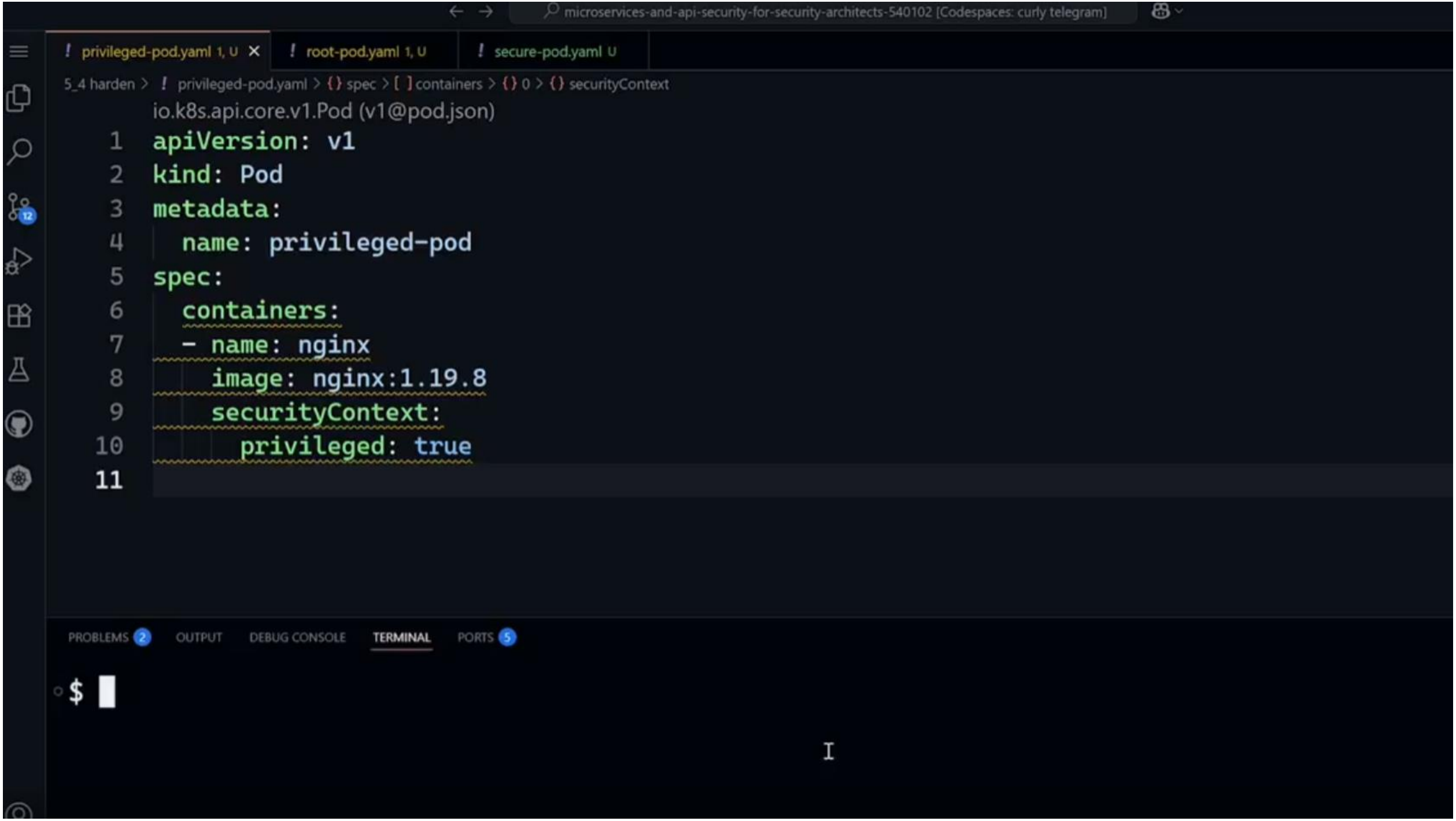

microservices-and-api-security-for-security-architects-540102 [Codespaces: cautious goggles]
bad-deployment.yaml
restrict-image-registries.yaml
5_3 kyverno > restrict-image-registries.yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
name: restrict-image-registries
spec:
validationFailureAction: enforce
rules:
- name: validate-registries
match:
resources:
kinds:
- Pod
validate:
message: "Only images from approved registries are allowed"
pattern:
spec:
containers:
- name: "*"
image: "nginx* | mcr.microsoft.com/*"


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
ception' to true.
💡 Note: There is a trade-off when deciding which approach to take regarding Namespace
usions. Please see the documentation at https://kyverno.io/docs/installation/#security-
perability to understand the risks.
$ kubectl get pods -n kyverno
NAME READY STATUS RESTARTS AGE
kyverno-admission-controller-5bcbdff469-pnstk 0/1 Running 0 14s
kyverno-background-controller-7c7d4dbbc9-s6d9c 1/1 Running 0 14s
kyverno-cleanup-controller-745cbc6f8d-dcbh9 0/1 Running 0 14s
kyverno-reports-controller-7867ffd654-l9nkn 1/1 Running 0 14s
$ kubectl get pods -n kyverno -w
NAME READY STATUS RESTARTS AGE
kyverno-admission-controller-5bcbdff469-pnstk 1/1 Running 0 27s
kyverno-background-controller-7c7d4dbbc9-s6d9c 1/1 Running 0 27s
kyverno-cleanup-controller-745cbc6f8d-dcbh9 1/1 Running 0 27s
kyverno-reports-controller-7867ffd654-l9nkn 1/1 Running 0 27s
^C$ code require-resources.yaml
$ kubectl apply -f require-resources.yaml
Warning: Validation failure actions enforce/audit are deprecated, use Enforce/Audit ins
.
clusterpolicy.kyverno.io/require-resources created
$ kubectl get clusterpolicy
NAME ADMISSION BACKGROUND READY AGE MESSAGE
require-resources true true True 15s Ready
$


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
$ kubectl apply -f bad-deployment.yaml
Error from server: error when creating "bad-deployment.yaml": admission webhook "validate.k
yverno.svc-fail" denied the request:
resource Deployment/default/nginx-bad was blocked due to the following policies
require-resources:
validate-resources: 'validation error: Resource requests and limits are required.
rule validate-resources failed at path /spec/template/spec/containers/0/resources/limit
s/'


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
$ helm search repo kyverno-policies
NAME CHART VERSION APP VERSION DESCRIPTION
kyverno/kyverno-policies 3.5.1 v1.15.1 Kubernetes Pod Securit
ndards implemented a...
$ helm install kyverno-policies --namespace kyverno kyverno/kyverno-policies --set pod
ityStandard=restricted --set validationFailureAction=Enforce
NAME: kyverno-policies
LAST DEPLOYED: Wed Sep 10 19:47:39 2025
NAMESPACE: kyverno
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Thank you for installing kyverno-policies 3.5.1
We have installed the "restricted" profile of Pod Security Standards and set them in E
e mode.
Visit https://kyverno.io/policies/ to find more sample policies.
$


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
We have installed the "restricted" profile of Pod Security Standards and set them in Enfo
e mode.
Visit https://kyverno.io/policies/ to find more sample policies.
$ kubectl get clusterpolicies
NAME ADMISSION BACKGROUND READY AGE MESSAGE
add-ns-labels true true True 17m Ready
disallow-capabilities true true True 14s Ready
disallow-capabilities-strict true true True 14s Ready
disallow-host-namespaces true true True 14s Ready
disallow-host-path true true True 14s Ready
disallow-host-ports true true True 14s Ready
disallow-host-process true true True 14s Ready
disallow-privilege-escalation true true True 14s Ready
disallow-privileged-containers true true True 14s Ready
disallow-proc-mount true true True 14s Ready
disallow-selinux true true True 14s Ready
require-resources true true True 22m Ready
require-run-as-non-root-user true true True 14s Ready
require-run-as-nonroot true true True 14s Ready
restrict-apparmor-profiles true true True 14s Ready
restrict-seccomp true true True 14s Ready
restrict-seccomp-strict true true True 14s Ready


microservices-and-api-security-for-security-architects-540102 [Codespaces: curly telegram]
privileged-pod.yaml 1, U
root-pod.yaml 1, U
secure-pod.yaml U
5_4 harden > privileged-pod.yaml > {} spec > [ ] containers > {} 0 > {} securityContext
io.k8s.api.core.v1.Pod (v1@pod.json)
1 apiVersion: v1
2 kind: Pod
3 metadata:
4 name: privileged-pod
5 spec:
6 containers:
7 - name: nginx
8 image: nginx:1.19.8
9 securityContext:
10 privileged: true
11
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS 5
$

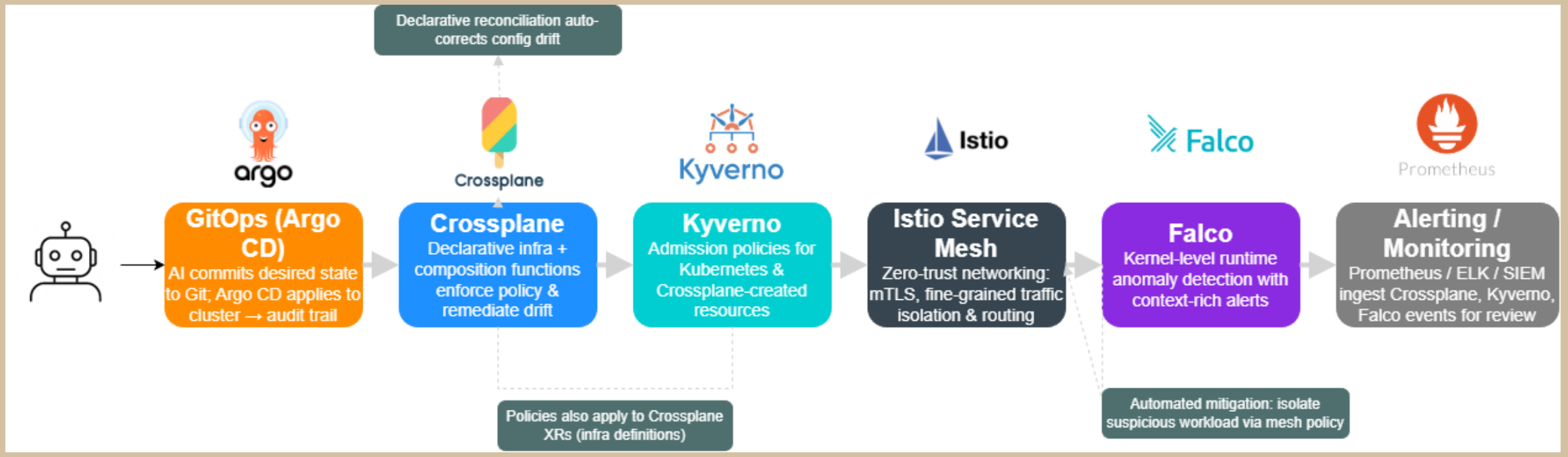
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS 5

```
$ kubectl apply -f privileged-pod.yaml
Error from server: error when creating "privileged-pod.yaml": admission webhook "validate.k
yverno.svc-fail" denied the request:

resource Pod/default/privileged-pod was blocked due to the following policies

disallow-capabilities-strict:
  require-drop-all: 'validation failure: Containers must drop `ALL` capabilities.'
disallow-privilege-escalation:
  privilege-escalation: 'validation error: Privilege escalation is disallowed. The
    fields spec.containers[*].securityContext.allowPrivilegeEscalation, spec.initContainers
[*].securityContext.allowPrivilegeEscalation,
    and spec.ephemeralContainers[*].securityContext.allowPrivilegeEscalation must
    be set to `false`. rule privilege-escalation failed at path /spec/containers/0/security
Context/allowPrivilegeEscalation/'
disallow-privileged-containers:
  privileged-containers: 'validation error: Privileged mode is disallowed. The fields
    spec.containers[*].securityContext.privileged and spec.initContainers[*].securityContex
t.privileged
    must be unset or set to `false`. rule privileged-containers failed at path /spec/contai
ners/0/securityContext/privileged/'
require-run-as-nonroot:
  run-as-non-root: 'validation error: Running as root is not allowed. Either the field
    spec.securityContext.runAsNonRoot must be set to `true`, or the fields spec.containers[
```

Demo: what did we see?

- AI agent submitted a Deployment without resource limit
- Kyverno validated this against the deployed ClusterPolicy
- Non-compliant Deployment was blocked

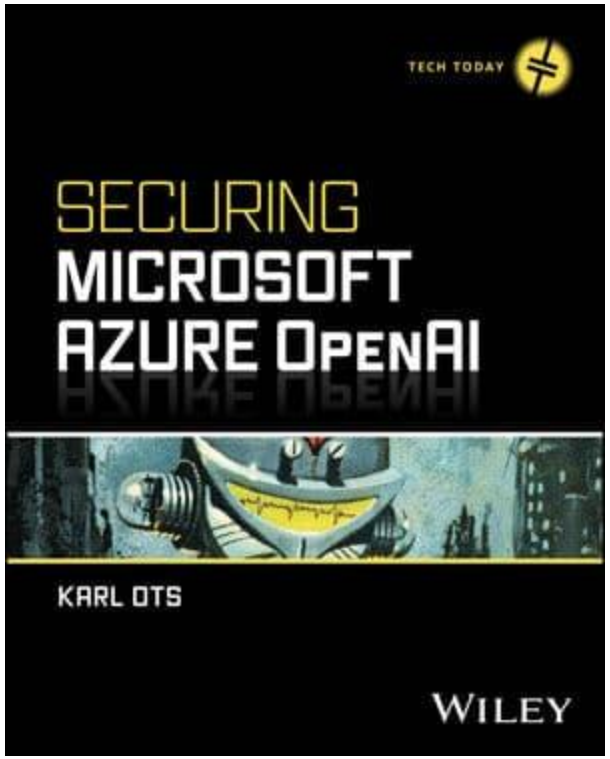
Bringing It All Together



Summary

- Autonomy != chaos
- Policy as Code for all
- Continuous compliance > periodic checks
- With the right design, we can enable AI confidently

upbound.io/intelligent-control-plane



a.co/d/iJ4YA3V

Course

Microservices and API Security for Security Architects: From Gateway Protection to Container Security

By: Karl Ots • 1 month ago

linkedin.com/learning/microservices-and-api-security-for-security-architects-from-gateway-protection-to-container-security/


Thank
you!