
KubeCon
CloudNativeCon
Europe 2023
TiKV

Automating Configuration and Permissions Testing for GitOps with OPA Conftest

Eve Ben Ezra & Mike Hume, The New York Times

Agenda

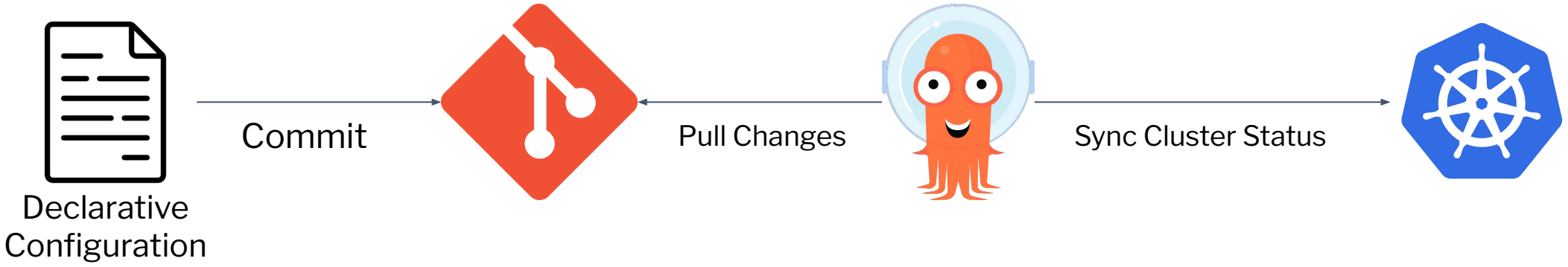
- Introduction
- Background and Definitions
- Internal Developer Platform
- Feedback and Developer Productivity
- OPA Conftest Overview and Demo
- Implementing Conftest
- CRD Validation with Kubeconform
- Summary and Questions

The New York Times

Background and Definitions

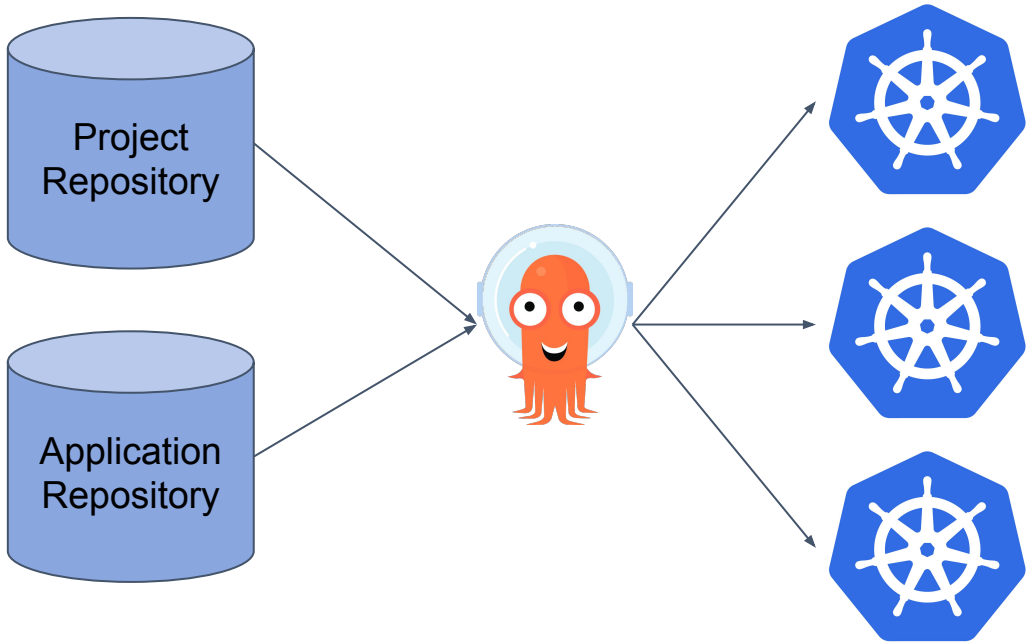
GitOps

An operational framework that uses Git as a single source of truth to manage infrastructure and application deployments, ensuring that the deployed environment always matches the desired state declaratively defined in Git.



Argo CD

A declarative, continuous delivery tool for Kubernetes that automates the deployment of applications to a K8s cluster. Argo CD relies on the GitOps operational framework.



Policy

A set of rules or guidelines that determine how a system should behave in specific situations. Policies generally define if certain actions are allowed, prohibited, or required under certain conditions.

Allow Policy

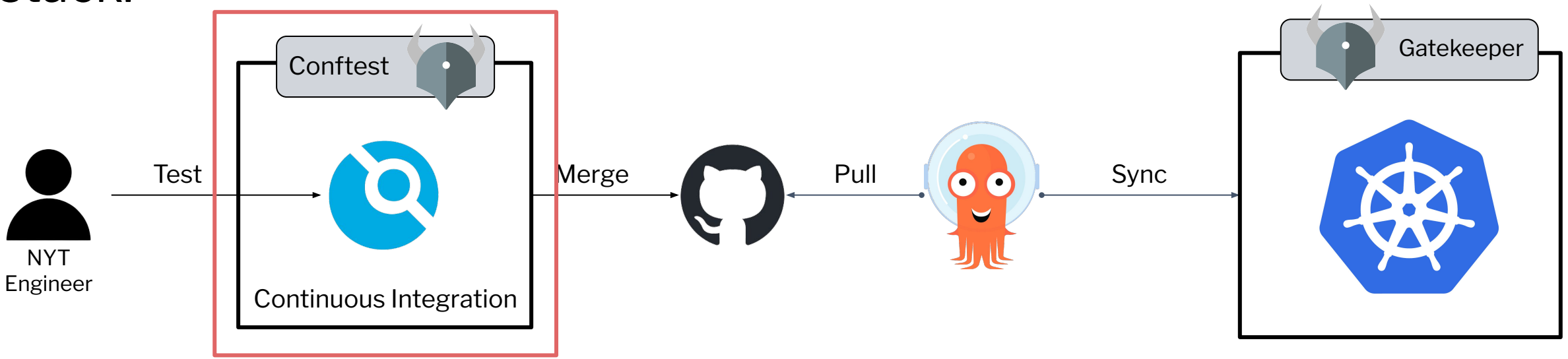
- Admin users can create or delete K8s resources

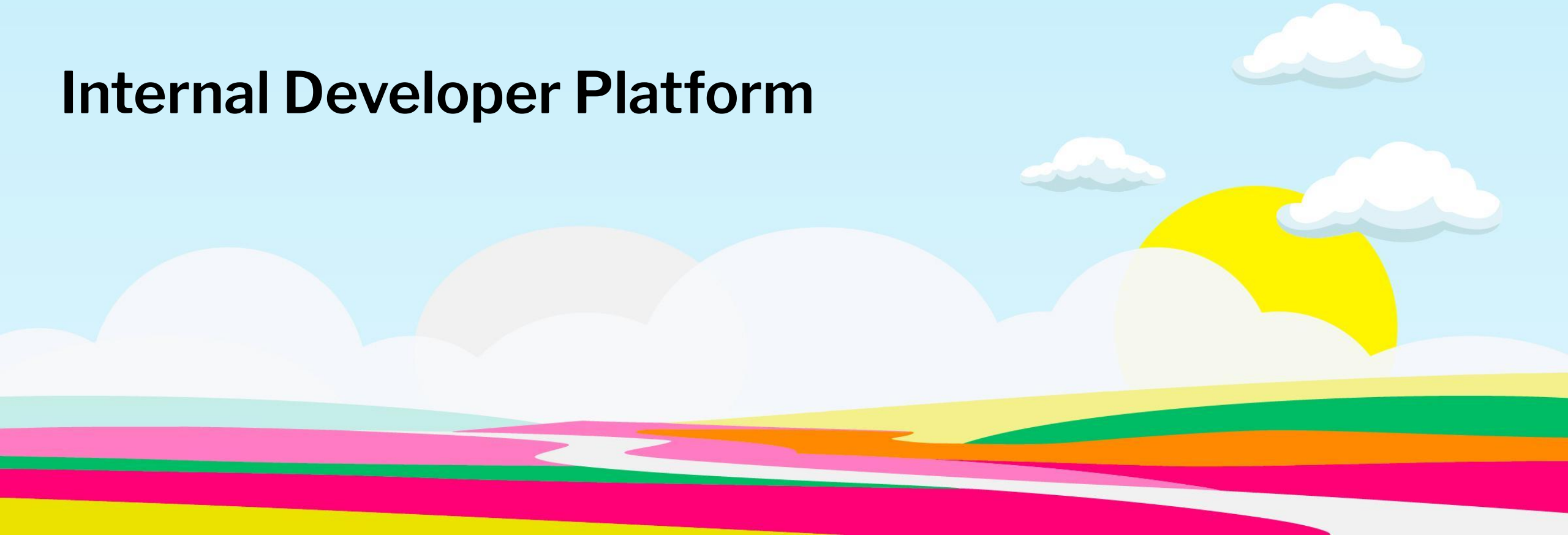
Deny Policy

- All incoming traffic from a specific IP address is denied
- Containers cannot run as root
- Images cannot be tagged :latest

Open Policy Agent

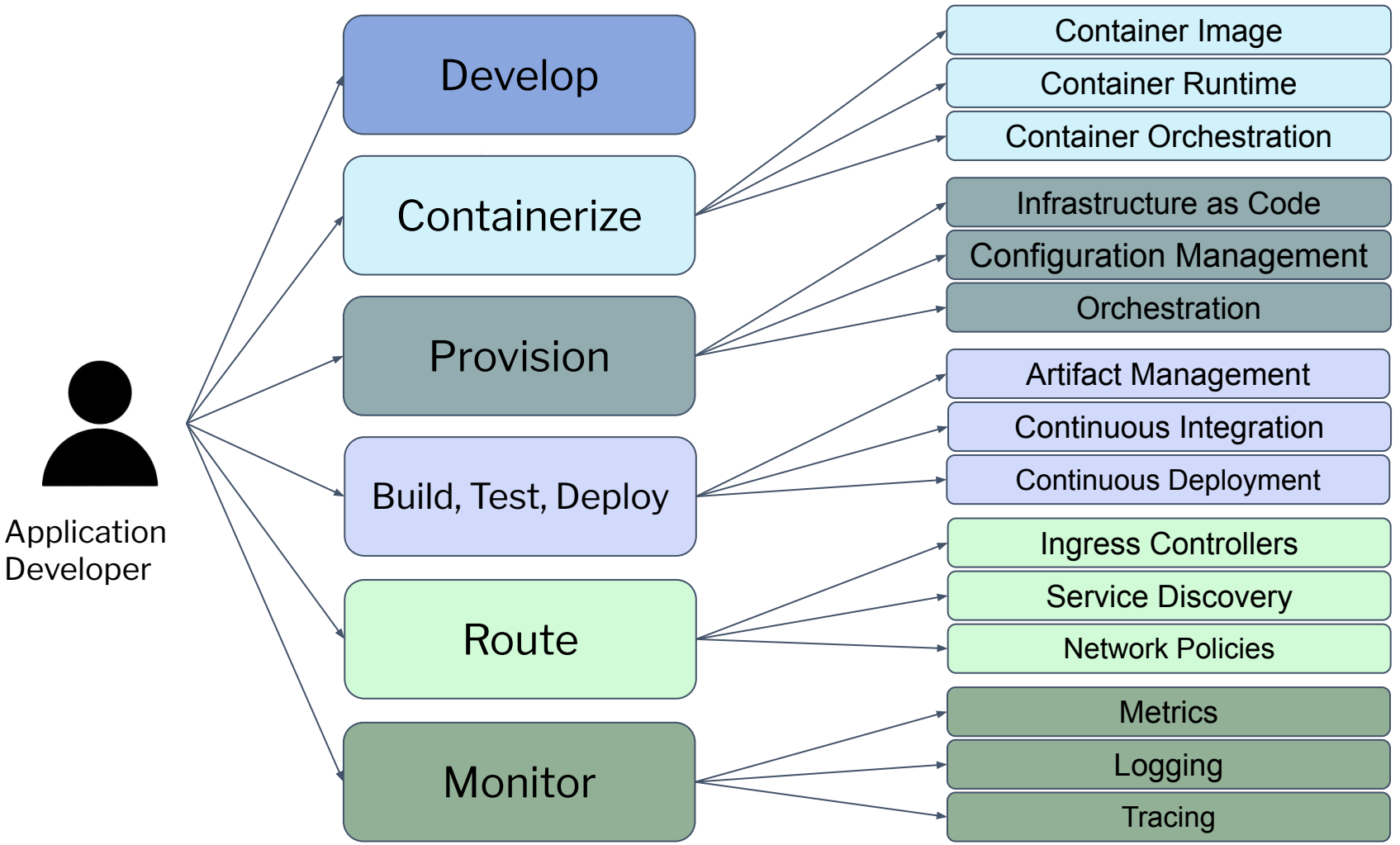
An open-source, general-purpose policy engine that unifies policy enforcement across the cloud-native stack.



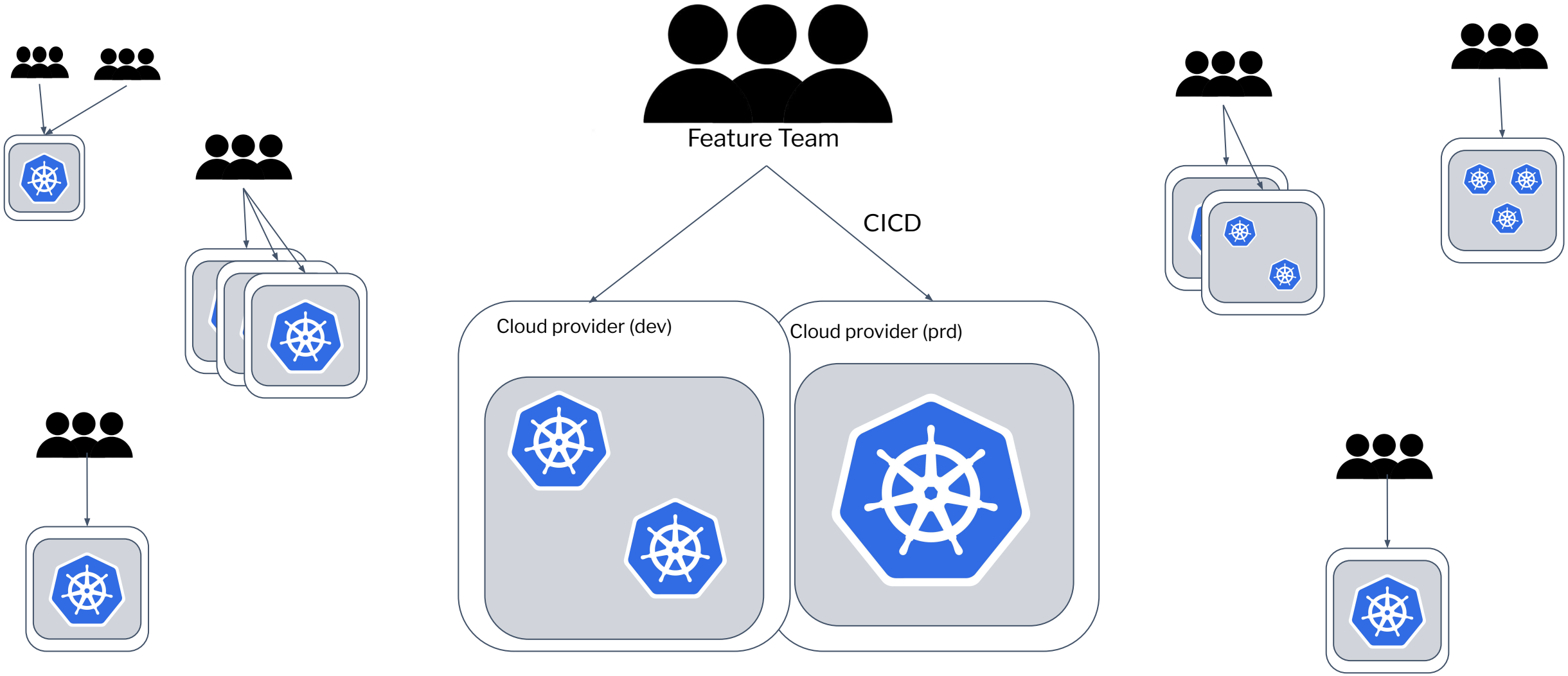


Internal Developer Platform

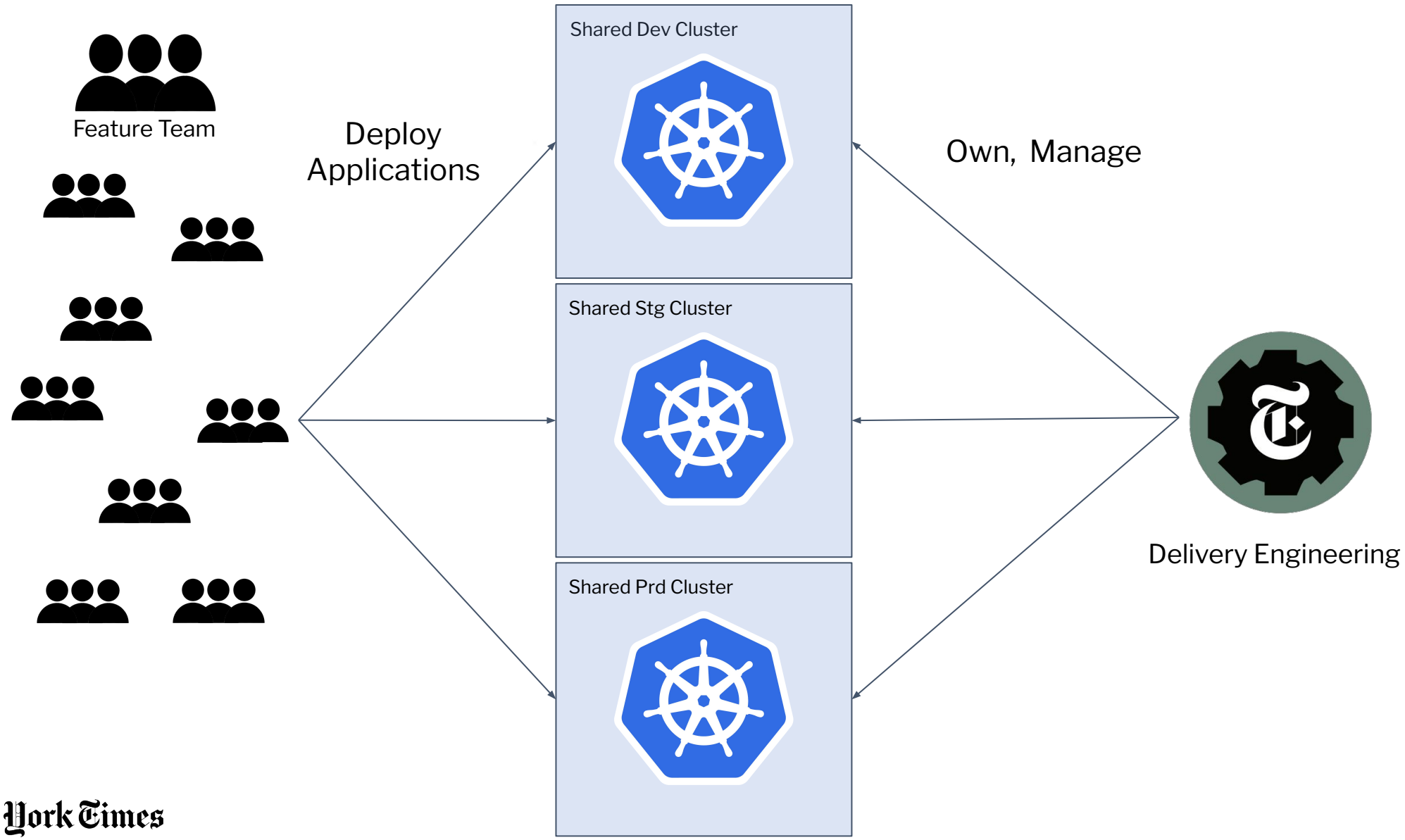
Internal Developer Platform



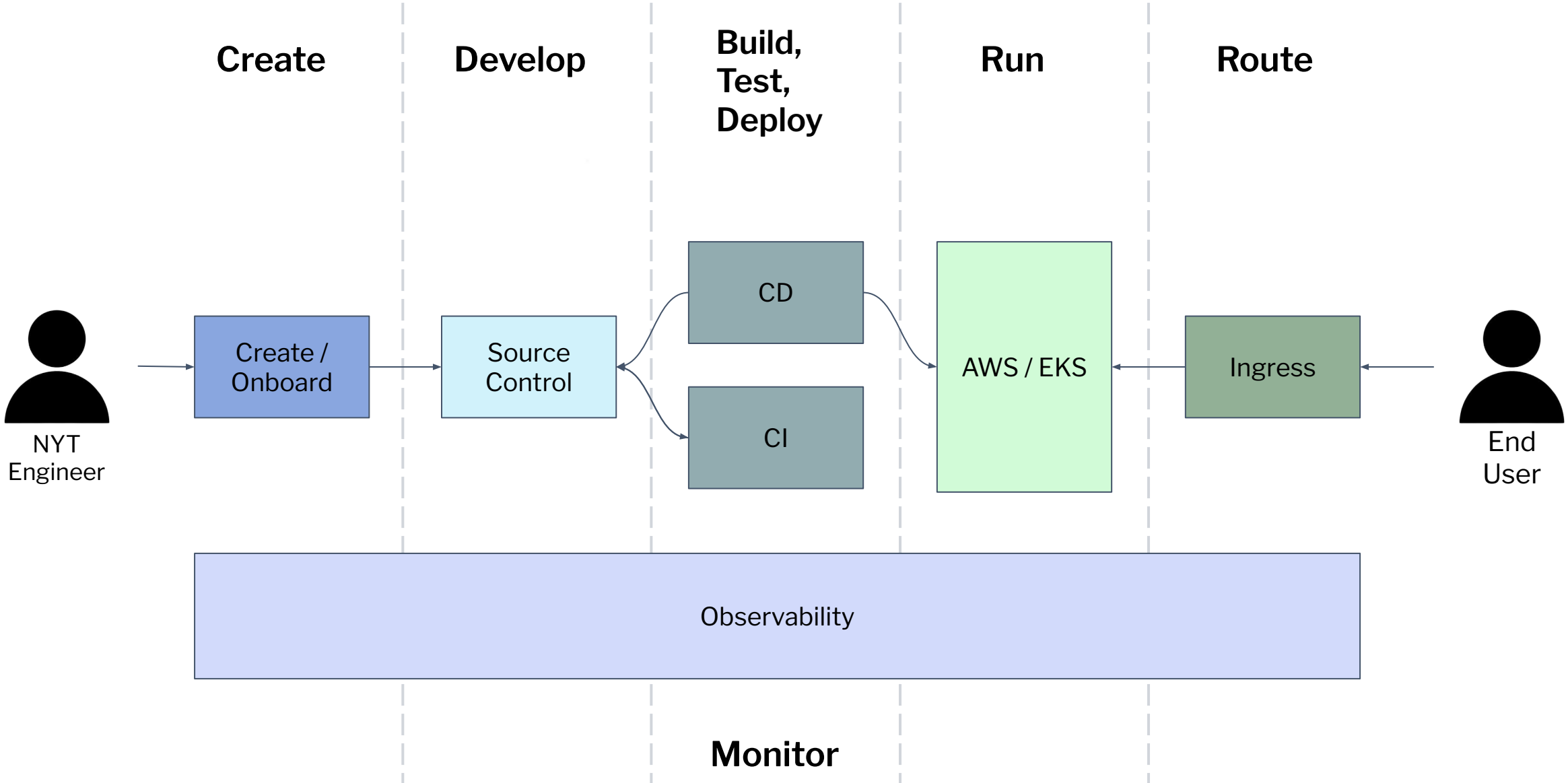
Internal Developer Platform



Internal Developer Platform

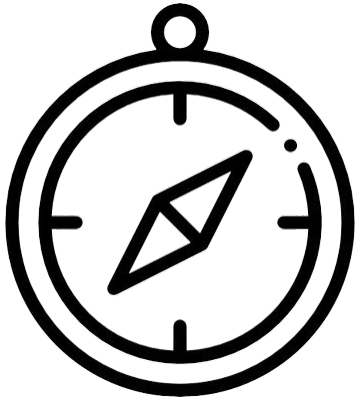


Internal Developer Platform

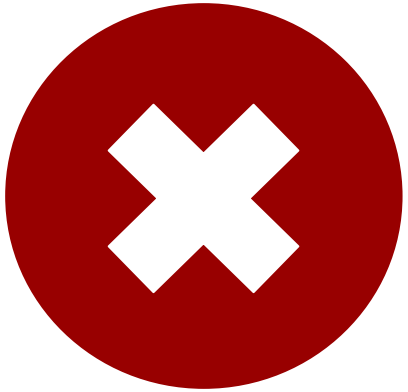


Shared Infrastructure Considerations:

Security



Developer Autonomy



Feedback

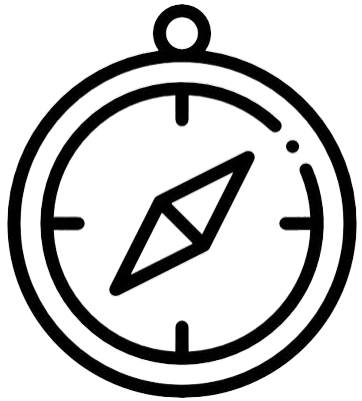
Shared Infrastructure Considerations:

Security

- Need for robust policy
- Need to ensure compliance across tenants

Shared Infrastructure Considerations:

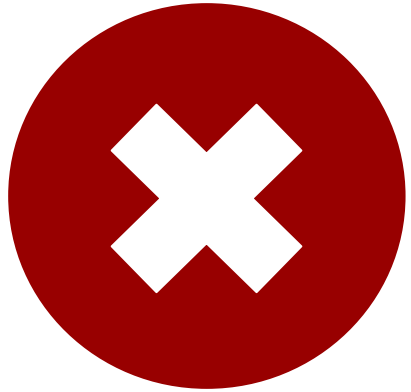
- Reduced freedom with respect to infrastructure
- Risk of increased frustration



Developer Autonomy

Shared Infrastructure Considerations:

- Late feedback decreases productivity and increases Time To Deployment (TTD)
- Requires clear communication about policies



Feedback

Feedback and Developer Productivity

Feedback and Developer Productivity

Question: When is the ideal time for developers to begin getting feedback in a GitOps operational framework?

A. During PR process, before merge to main branch
B. During automated testing in a CI process
C. After merge, during sync process
D. While developing locally
E. After any user or stakeholder testing
F. As frequently as possible throughout the development lifecycle

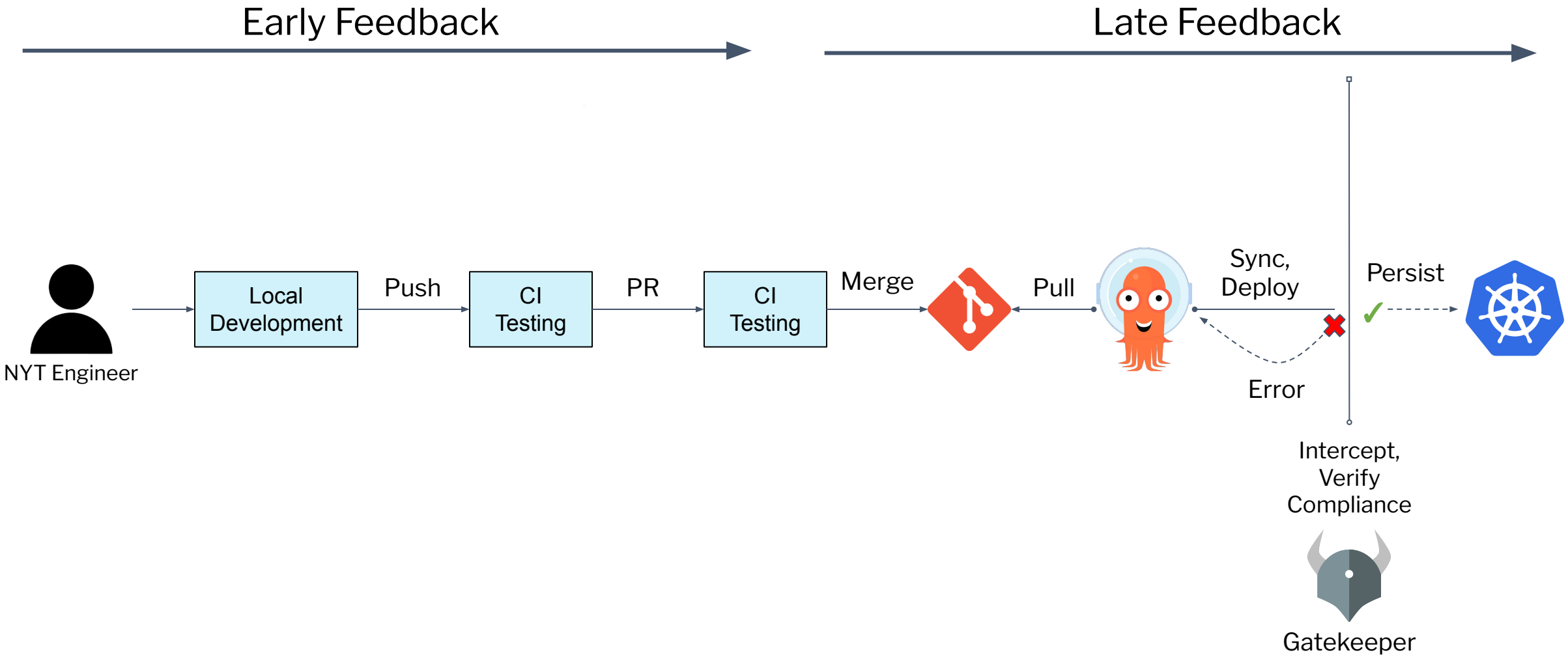
Feedback and Developer Productivity

Question: When is the ideal time for developers to begin getting feedback in a GitOps operational framework?

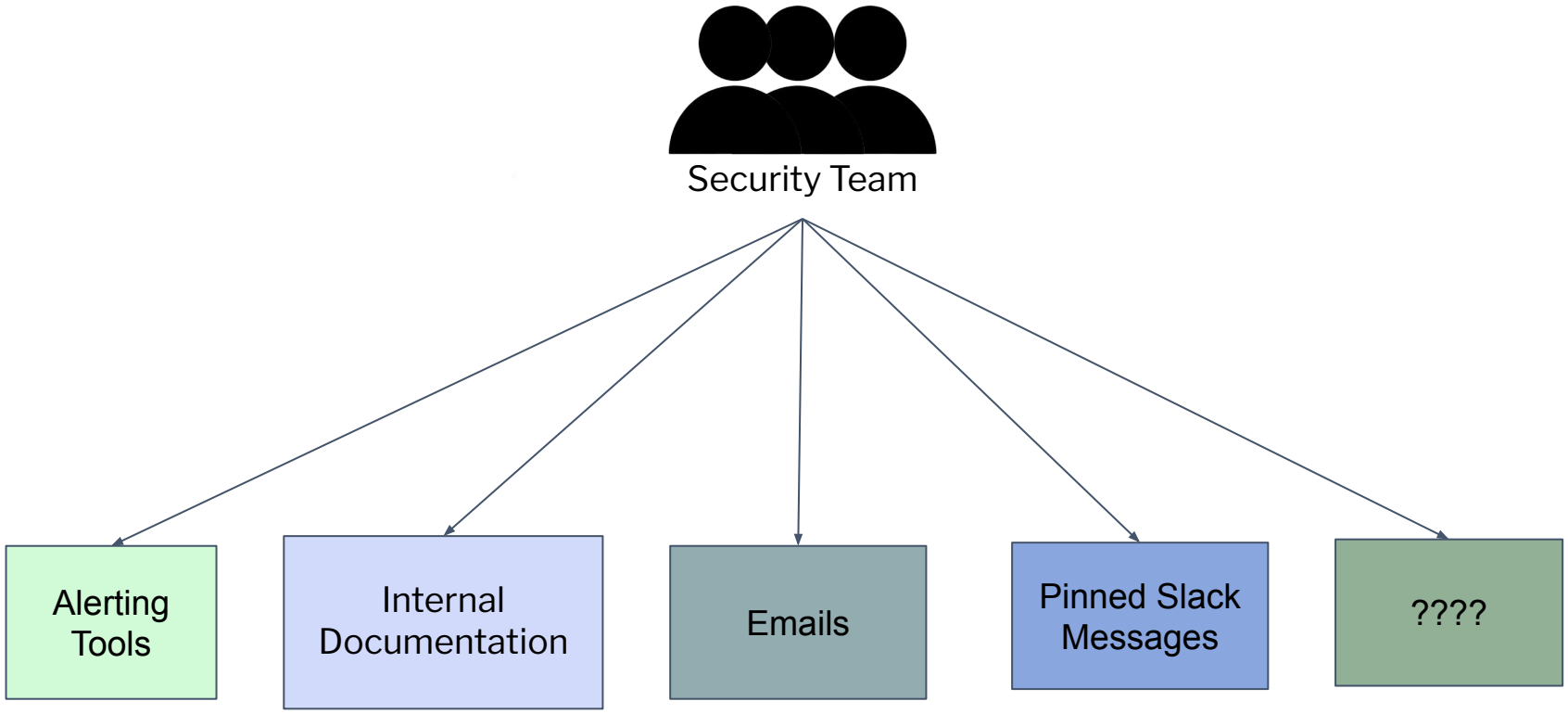
D. While developing locally

F. As frequently as possible throughout the development lifecycle

Feedback and Developer Productivity



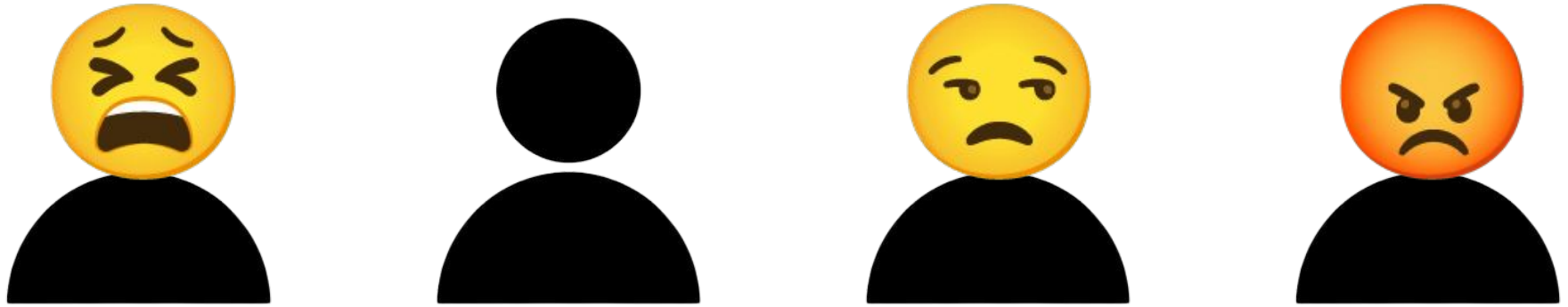
Feedback and Developer Productivity



The New York Times

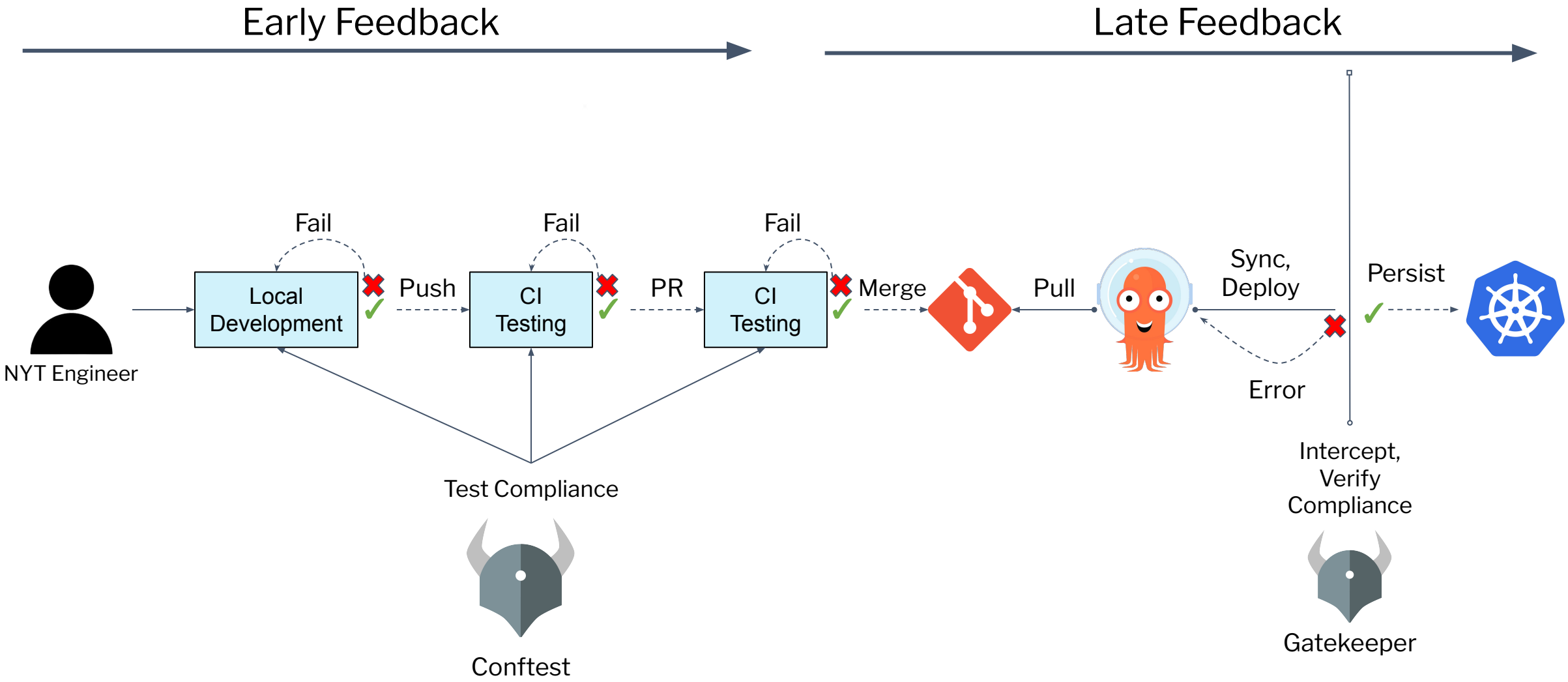
Feedback and Developer Productivity

Late Feedback, Hard-To-Find Policies = Frustrated Devs, Lowered Productivity

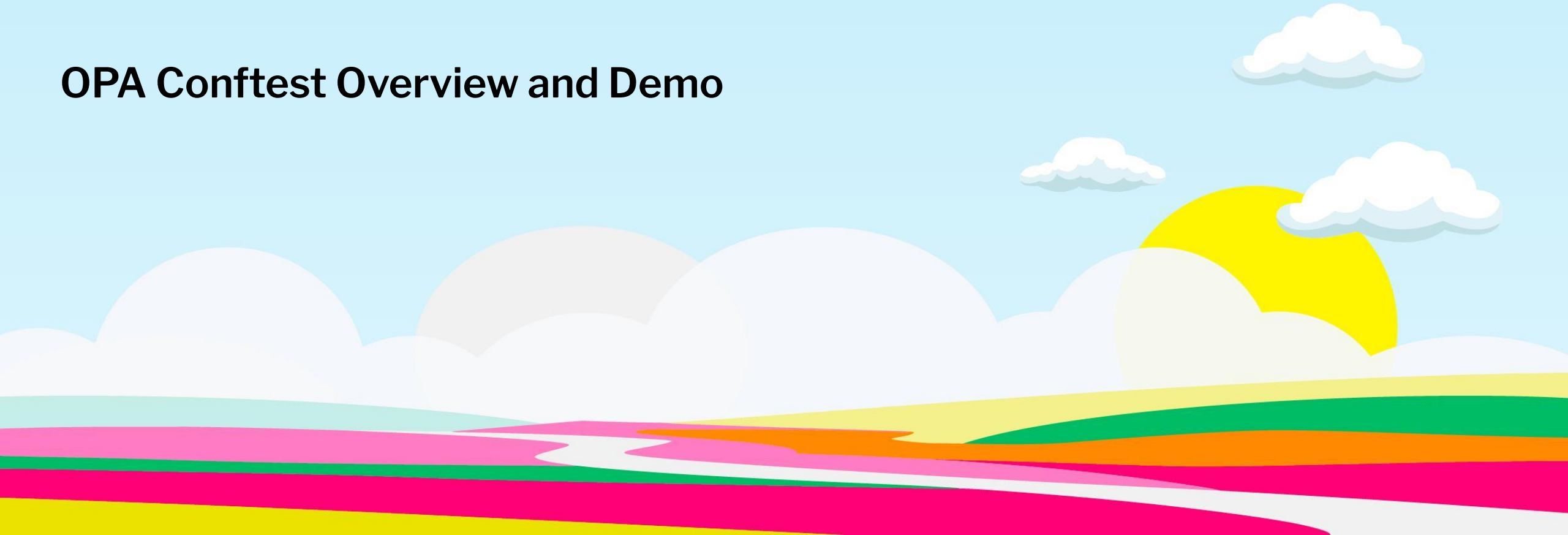


The New York Times

Feedback and Developer Productivity



The New York Times



OPA Conftest Overview and Demo

Policy Demo

KubeCon CloudNativeCon Europe 2023

ecbenezra/kubecon-eu-2023-conftest

```
conftest pull git::https://github.com/ecbenezra/kubecon-eu-2023-conftest.git//kubecon-examples/
```

The New York Times

Learning Curve

Easy to read, write, and understand?

Conftest Policies

- Warning
  - Use the *warn* rule
- Failure
  - Use the *deny* rule
    - Returns string
  - Use the *violation* rule
    - Returns structured data error

- Name rules *warn, deny, violation*
  - *Optional*: suffix rules
    - warn_rule_name
    - deny_rule_name
    - violation_rule_name

Policy Example: Latest Tags

Policy: No latest tags in staging or production

```yaml
spec:
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

Security Engineer

Hacker

The New York Times

Policy Example: Latest Tags

Deployment Spec

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:1.3.2
        ports:
        - containerPort: 80
      - name: other-container
        image: my-image:latest
        ports:
        - containerPort: 8080
```

Path: spec.template.spec.containers

Rego Policy

```
deny[msg] {
    image := input.spec.template.spec.containers[_].image
    endswith(image,"latest")
    msg := sprintf("image %v cannot be tagged latest %v/%v",[image,input.kind,input.metadata.name])
}
```

Policy Example: Latest Tags


```
⊗ $ conftest test example-yamls/
  FAIL - example-yamls/deployment.yaml - main - image my-image:latest cannot be tagged latest Deployment/nginx-deployment

  1 test, 0 passed, 0 warnings, 1 failure, 0 exceptions
```

Policy Example: Latest Tags

Dev Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-dev
  labels:
    app: nginx
    env: dev
spec:
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

Stg Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-stg
  labels:
    app: nginx
    env: stg
spec:
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
```

Policy Example: Latest Tags

Enforce
`metadata.label.env`

```
deny_no_env_label[msg] {
    labelKinds[input.kind]
    not input.metadata.labels.env
    msg := sprintf("%v/%v does not contain env
    label", [input.kind, input.metadata.name])
}
```

Enforce
Environment Value

```
allowedEnvs := {"dev", "stg", "prd"}

deny_invalid_env_label[msg] {
    labelKinds[input.kind]
    not allowedEnvs[input.metadata.labels.env]
    msg := sprintf("%v/%v metadata label must be
    equal to dev, stg, or prd", [input.kind,
    input.metadata.name])
}
```

Policy Example: Latest Tags


```
is_dev {
    input.metadata.labels.env == "dev"
}
```

Policy Example: Latest Tags


```
deny_latest_tags[msg] {
    not is_dev
    containerImage := input.spec.template.spec.containers[_].image
    endswith(containerImage, "latest")
    msg := sprintf("image %v cannot be tagged latest %v/%v",
    [containerImage, input.kind, input.metadata.name])
}
```

Policy Example: Latest Tags


```
warn_latest_tags[msg] {
    is_dev
    containerImage := input.spec.template.spec.containers[_].image
    endswith(containerImage, "latest")
    msg := sprintf("dev image is tagged latest %v/%v. Be aware latest tags are not
    permitted in stg or prd", [input.kind, input.metadata.name])
}
```

Policy Example: Latest Tags


```
package main

# deny K8s objects without an environment label
deny_no_env_label[msg] {
    not input.metadata.labels.env
    msg := sprintf("%v/%v does not contain env label", [input.kind, input.metadata.
    name])
}

is_dev {
    input.metadata.labels.env == "dev"
}

deny_latest_tags[msg] {
    not is_dev
    containerImage := input.spec.template.spec.containers[_].image
    endswith(containerImage, "latest")
    msg := sprintf("image %v cannot be tagged latest %v/%v", [containerImage, input.
    kind, input.metadata.name])
}

warn_latest_tags[msg] {
    is_dev
    containerImage := input.spec.template.spec.containers[_].image
    endswith(containerImage, "latest")
    msg := sprintf("dev image is tagged latest %v/%v. Be aware latest tags are not
    permitted in stg or prd", [input.kind, input.metadata.name])
}
```

Policy Example: Latest Tags

KubeCon CloudNativeCon Europe 2023

```
$ conftest test -p policy/ example-yamls/
WARN - example-yamls/deployment.yaml - main - dev image nginx:latest is tagged latest Deployment/nginx-dev. Be aware that latest tags are not permitted in stg or prd envs
FAIL - example-yamls/deployment.yaml - main - image nginx:latest cannot be tagged latest Deployment/nginx-stg
```

The New York Times

Policy Example: Policy Exceptions

Deployment Spec

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: vital-deployment-stg
  labels:
    app: nginx
    env: stg
spec:
  replicas: 3
  selector:
    matchLabels:
      app: vital-deployment
  template:
    metadata:
      labels:
        app: vital-deployment
    spec:
      containers:
      - name: busybox
        image: busybox:latest
        ports:
        - containerPort: 80
      - name: important-image
        image: vital-image:latest
        ports:
        - containerPort: 80
```

Rego Policy

```
allowedImages := {"busybox:latest", "vital-image:latest"}

exception[rules] {
    allowedImages[input.spec.template.spec.containers[_].
    image]
    rules := ["latest_tags"]
}
```

```
deny_latest_tags[msg] {
    not is_dev
    containerImage := input.spec.template.spec.containers[_].image
    endswith(containerImage, "latest")
    msg := sprintf("image %v cannot be tagged latest %v/%v",
    [containerImage, input.kind, input.metadata.name])
}
```

Policy Example: Understanding Policy Logic


```
# deny K8s objects without an environment label
deny_no_env_label[msg] {
    not input.metadata.labels.env
    msg := sprintf("%v/%v does not contain env label",[input.kind,input.metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for pods
    containerImage := input.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for deployments and replicasets
    containerImage := input.spec.template.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}
```

Pods

Deployments / ReplicaSets

Policy Example: Logical AND

All evaluate to true

```
# deny K8s objects without an environment label
deny_no_env_label[msg] {
    not input.metadata.labels.env
    msg := sprintf("%v/%v does not contain env label",[input.kind,input.metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for pods
    containerImage := input.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for deployments and replicasets
    containerImage := input.spec.template.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}
```

Policy Example: Logical OR

Same policy name

```
# deny K8s objects without an environment label
deny_no_env_label[msg] {
    not input.metadata.labels.env
    msg := sprintf("%v/%v does not contain env label",[input.kind,input.metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for pods
    containerImage := input.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}

deny_latest_tags[msg] {
    not is_dev
    # container image path for deployments and replicasets
    containerImage := input.spec.template.spec.containers[_].image
    image_latest_tag(containerImage)
    msg := sprintf("image %v cannot be tagged latest %v/%v",[containerImage,input.kind,input.
    metadata.name])
}
```

Policy Example: Unit Testing

Test

```
test_deny_stg_deployment {
    deny_latest_tags with input as stg_deployment_with_latest_tag
}
```

Testing Input

```
stg_deployment_with_latest_tag := {
    "apiVersion": "apps/v1",
    "kind": "Deployment",
    "metadata": {
        "name": "nginx-stg",
        "labels": {
            "app": "nginx",
            "env": "stg",
        },
    },
    "spec": {"template": {"spec": {"containers": [{
        "name": "nginx",
        "image": "nginx:latest",
        "ports": [{"containerPort": 80}],
    }]}}},
}
```

Policy Example: Code Coverage


```
$ opa test kubecon-examples/ --coverage
```

```
        {
          "start": {
            "row": 57
          },
          "end": {
            "row": 57
          }
        }
      ],
      "covered_lines": 5,
      "coverage": 100
    }
  },
  "covered_lines": 41,
  "not_covered_lines": 3,
  "coverage": 93.2
```

Rego Versatility

Use Conftest for:

- Kubernetes
- Dockerfile
- Terraform
- Kustomize
- Typescript
- Jsonnet
- XML

etc

The New York Times

Example: Drone CI


```
package main

import future.keywords

image_latest_tag(imageName) if {
    endswith(imageName, "latest")
}

deny_no_image_tag contains msg if {
    input.kind == "pipeline"
    not every_image_contains_tag
    msg = sprintf("%v/%v contains images with no tag", [input.kind, input.name])
}

deny_image_latest contains msg if {
    input.kind == "pipeline"
    image = input.steps[_].image
    image_latest_tag(image)
    msg = sprintf("%v/%v contains images with latest tag: %v", [input.kind, input.name, image])
}

every_image_contains_tag if {
    every step in input.steps {
        contains(step.image, ":")
    }
}
```

Example: ArgoCD AppProject

AppProject CRD

```yaml
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: my-project
  namespace: argocd
spec:
  namespaceResourceWhitelist:
  - group: 'apps'
    kind: Deployment
  - group: 'apps'
    kind: StatefulSet
  - group: 'policy'
    kind: AnotherKnownPolicy
  - group: 'networking.istio.io'
    kind: Gateway
  - group: 'networking.istio.io'
    kind: DestinationRule
```

Rego Policy

```
allowedNamespaceResources := {
    "apps": {"Deployment", "ReplicaSet", "StatefulSet"},
    "policy": {"AnotherKnownPolicy"},
    "networking.istio.io": {"Gateway", "DestinationRule"},
}
```

```
deny_namespace_groups[msg] {
    group := input.spec.namespaceResourceWhitelist[_].group
    not allowedNamespaceResources[group]
    msg = sprintf("namespaceResourceWhitelist group %v not
    permitted", [group])
}
```

```
deny_namespace_kinds[msg] {
    some i # since order matters, define explicit iterator
    group := input.spec.namespaceResourceWhitelist[i].group
    kind := input.spec.namespaceResourceWhitelist[i].kind
    allowedKinds := allowedNamespaceResources[group]
    not allowedKinds[kind]
    msg = sprintf("namespaceResourceWhitelist kind %v not
    allowed for group %v", [kind, group])
}
```

Example: ArgoCD AppProject

KubeCon CloudNativeCon Europe 2023

Using `some i`

```
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: apps; kind: Deployment; allowedKinds: {"Deployment", "ReplicaSet", "StatefulSet"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: apps; kind: StatefulSet; allowedKinds: {"Deployment", "ReplicaSet", "StatefulSet"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: policy; kind: AnUnknownPolicy; allowedKinds: {"AnotherKnownPolicy"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: apps; kind: Deployment; allowedKinds: {"Deployment", "ReplicaSet", "StatefulSet"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: apps; kind: StatefulSet; allowedKinds: {"Deployment", "ReplicaSet", "StatefulSet"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: policy; kind: AnotherKnownPolicy; allowedKinds: {"AnotherKnownPolicy"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: networking.istio.io; kind: Gateway; allowedKinds: {"DestinationRule", "Gateway"}
PRNT   kubecon-examples/argo-policies/argo.rego:21: group: networking.istio.io; kind: DestinationRule; allowedKinds: {"DestinationRule", "Gateway"}

FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist group aThirdSneakyGroup not permitted
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind AnUnknownPolicy not allowed for group policy

4 tests, 2 passed, 0 warnings, 2 failures, 0 exceptions
```

The New York Times

Example: ArgoCD AppProject

Using _

```
PRNT   kubecon-examples/argo-policies/argo.rego:20: group: networking.istio.io; kind: Deployment; allowedKinds: {"DestinationRule", "Gateway"}

PRNT   kubecon-examples/argo-policies/argo.rego:20: group: networking.istio.io; kind: StatefulSet; allowedKinds: {"DestinationRule", "Gateway"}

PRNT   kubecon-examples/argo-policies/argo.rego:20: group: networking.istio.io; kind: AnotherKnownPolicy; allowedKinds: {"DestinationRule", "Gateway"}

PRNT   kubecon-examples/argo-policies/argo.rego:20: group: networking.istio.io; kind: Gateway; allowedKinds: {"DestinationRule", "Gateway"}

PRNT   kubecon-examples/argo-policies/argo.rego:20: group: networking.istio.io; kind: DestinationRule; allowedKinds: {"DestinationRule", "Gateway"}

FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist group aThirdSneakyGroup not permitted
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind AnUnknownPolicy not allowed for group apps
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind AnUnknownPolicy not allowed for group policy
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind Deployment not allowed for group policy
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind Gateway not allowed for group apps
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind Gateway not allowed for group policy
FAIL - example-configs/argo/failing-project.yaml - main - namespaceResourceWhitelist kind StatefulSet not allowed for group policy
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind AnotherKnownPolicy not allowed for group apps
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind AnotherKnownPolicy not allowed for group networking.istio.io
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind Deployment not allowed for group networking.istio.io
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind Deployment not allowed for group policy
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind DestinationRule not allowed for group apps
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind DestinationRule not allowed for group policy
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind Gateway not allowed for group apps
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind Gateway not allowed for group policy
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind StatefulSet not allowed for group networking.istio.io
FAIL - example-configs/argo/project.yaml - main - namespaceResourceWhitelist kind StatefulSet not allowed for group policy

18 tests, 1 passed, 0 warnings, 17 failures, 0 exceptions
```

Example: Traces

KubeCon CloudNativeCon Europe 2023

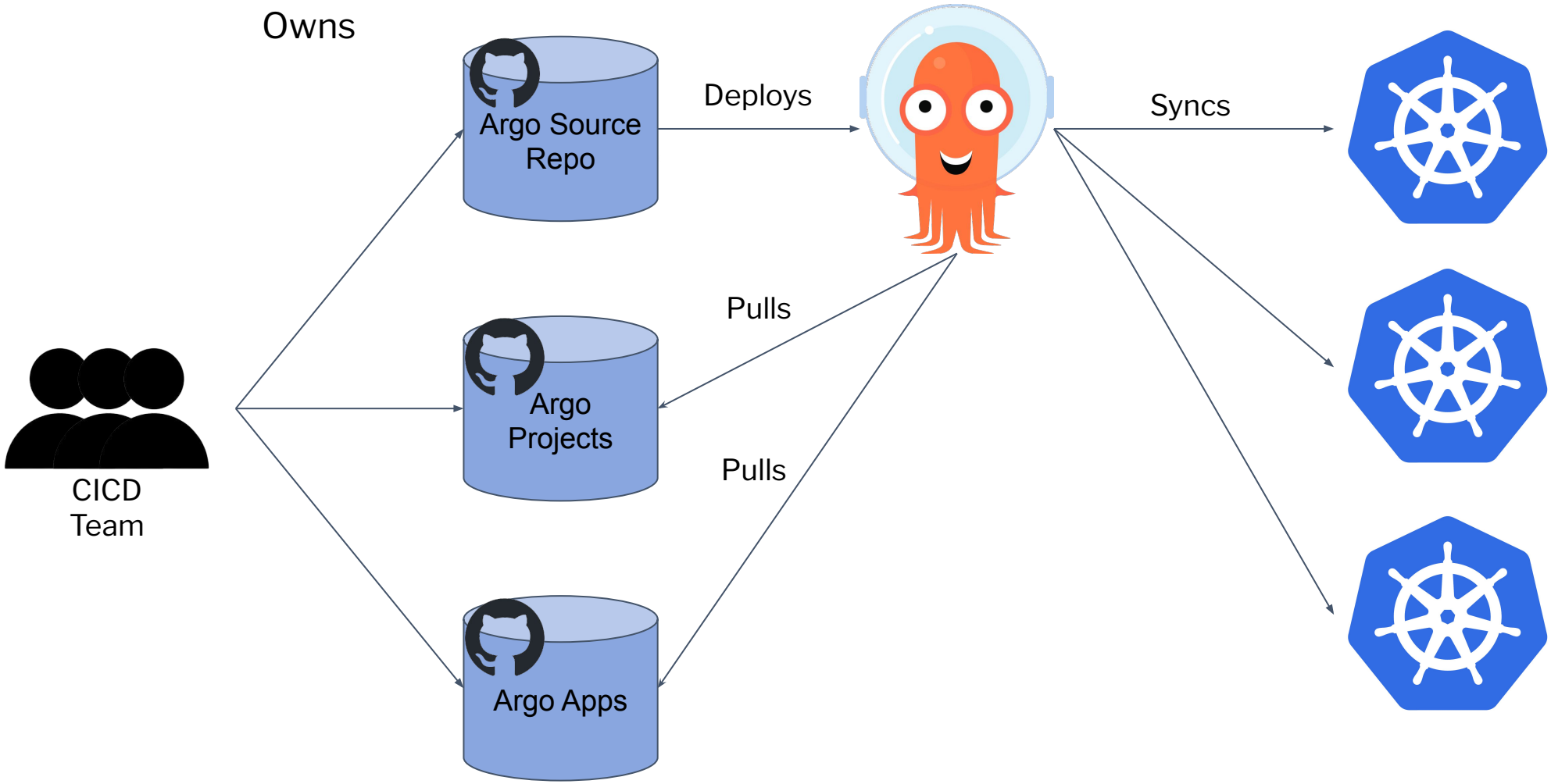
```
$ conftest test -p policy-dir/ config-dir/ --trace
```

```
file: example-configs/kubernetes/deployment.yaml | query: data.main.warn_latest_tags
TRAC   Enter data.main.warn_latest_tags = _
TRAC    | Eval data.main.warn_latest_tags = _
TRAC    | Index data.main.warn_latest_tags (matched 2 rules)
TRAC    | Enter data.main.warn_latest_tags
TRAC    | | Eval data.main.is_dev
TRAC    | | Index data.main.is_dev (matched 0 rules)
TRAC    | | Fail data.main.is_dev
TRAC    | Enter data.main.warn_latest_tags
TRAC    | | Eval data.main.is_dev
TRAC    | | Index data.main.is_dev (matched 0 rules)
TRAC    | | Fail data.main.is_dev
TRAC    | Exit data.main.warn_latest_tags = _
TRAC   Redo data.main.warn_latest_tags = _
TRAC    | Redo data.main.warn_latest_tags = _
```

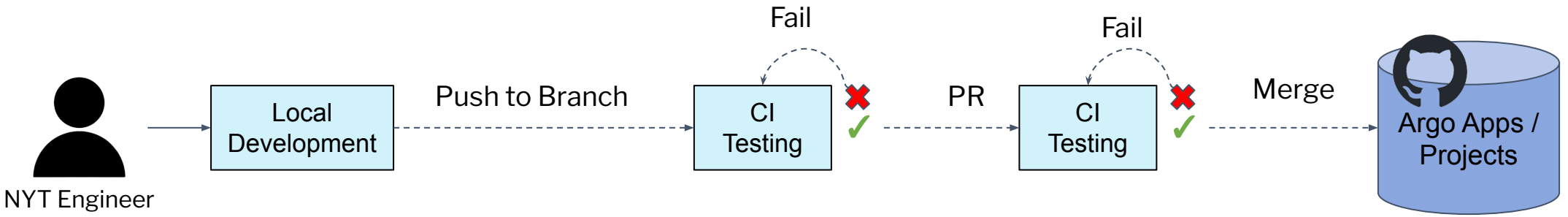
The New York Times

Implementing Conftest

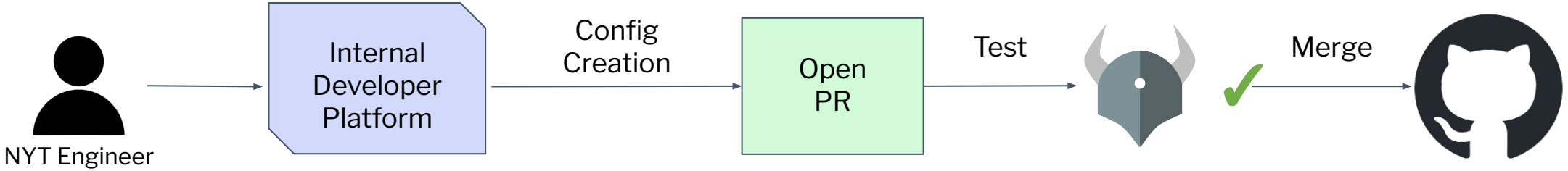
Implementing Conftest



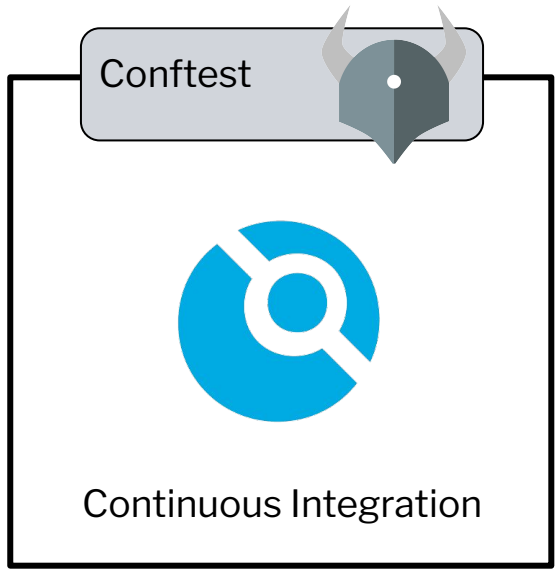
Implementing Conftest



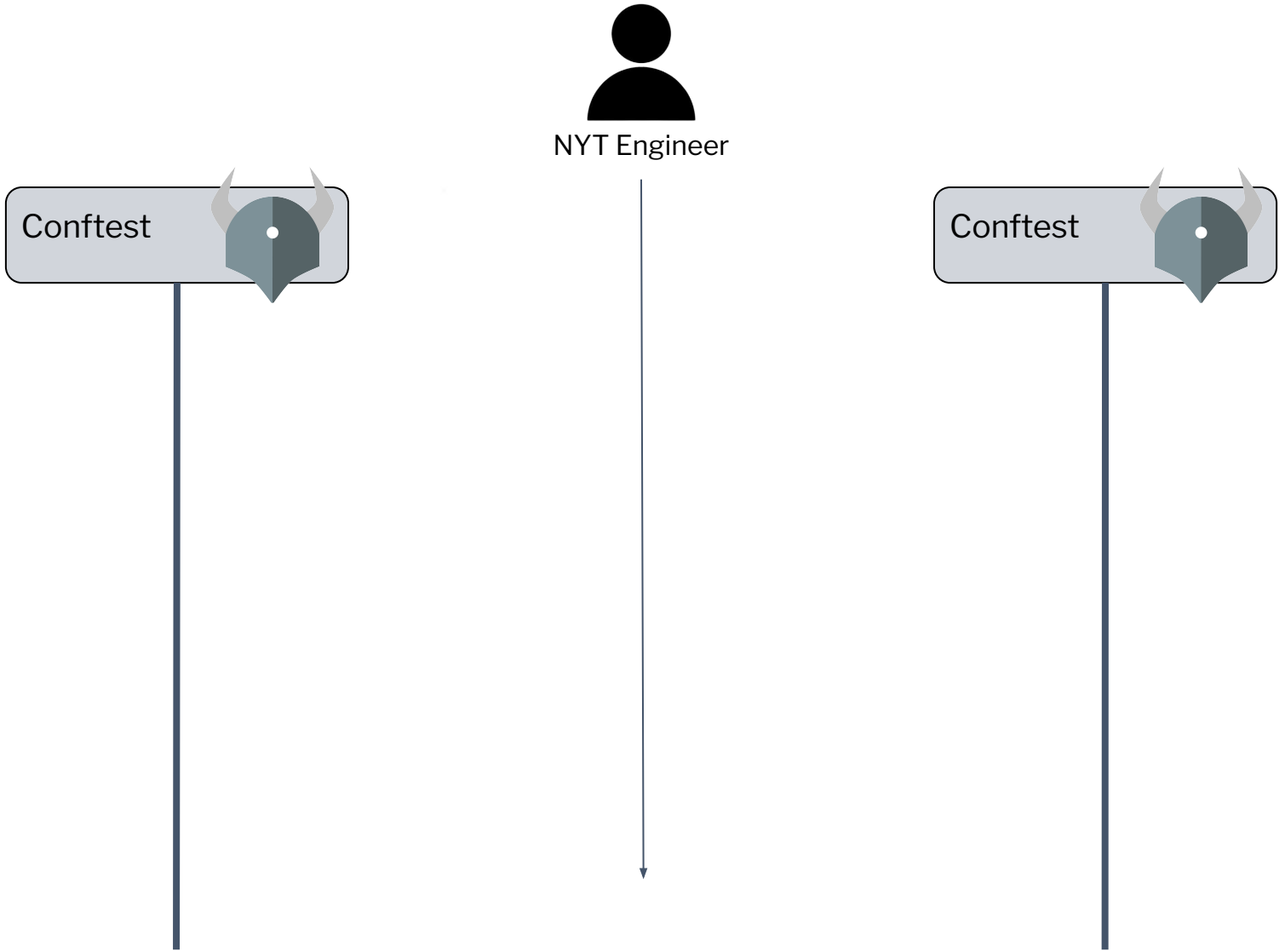
Implementing Conftest



Implementing Conftest



Implementing Conftest



Implementing Conftest

Conftest 

Robust policy implementation starts locally

Feature Engineers

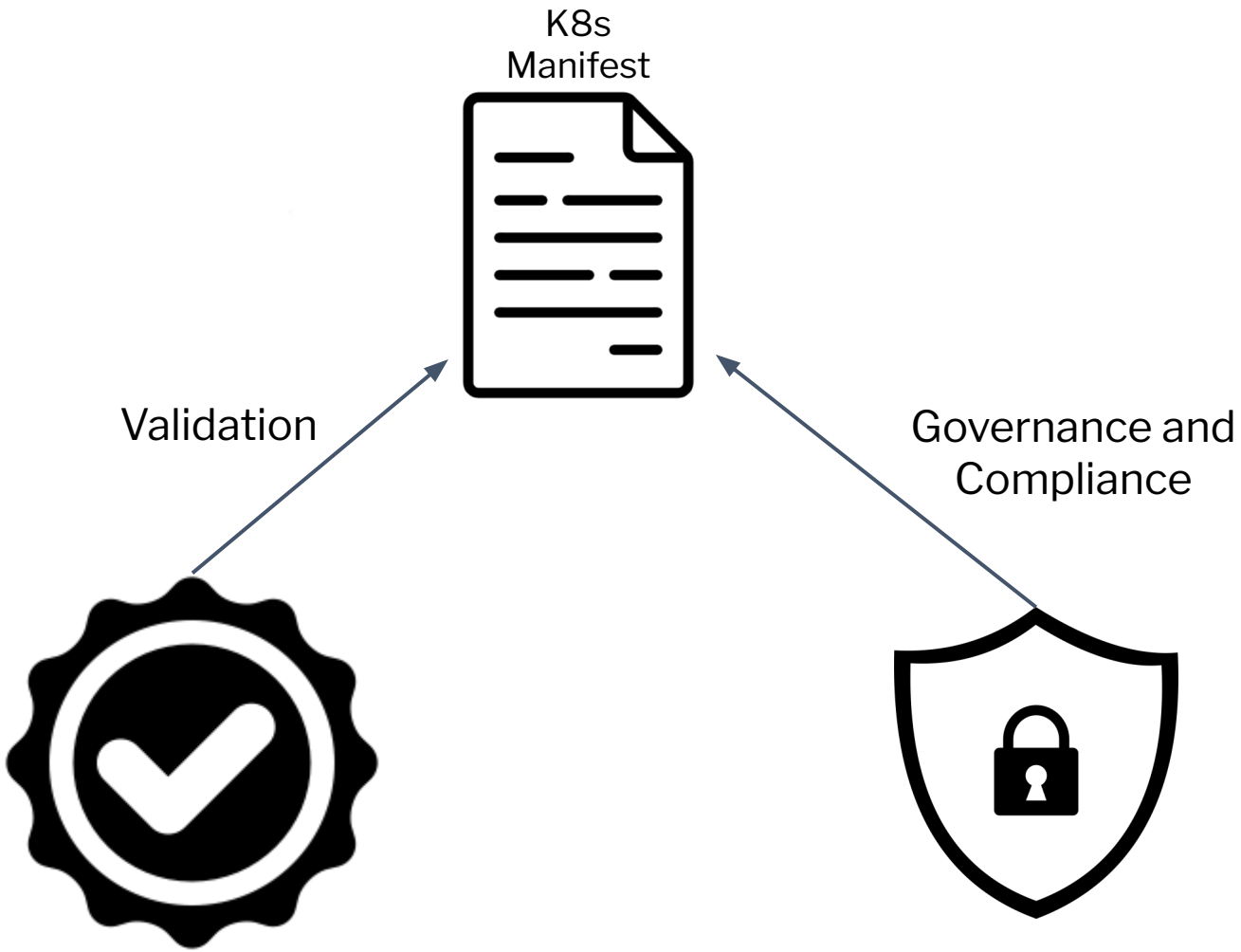
Security Engineers

Operations Engineers

The New York Times

Schema Validation with Kubeconform

Validation And Governance



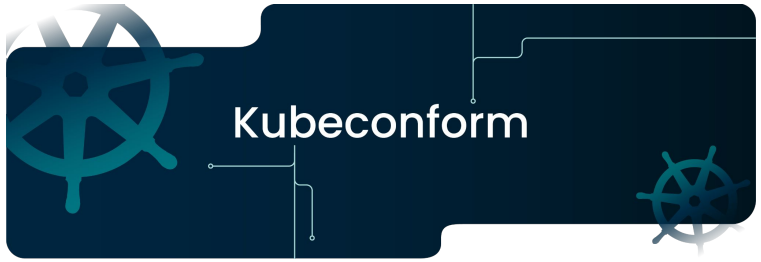
The New York Times

Why Manifest Validation

Misconfigured manifests (yaml) leads to unintended behaviors

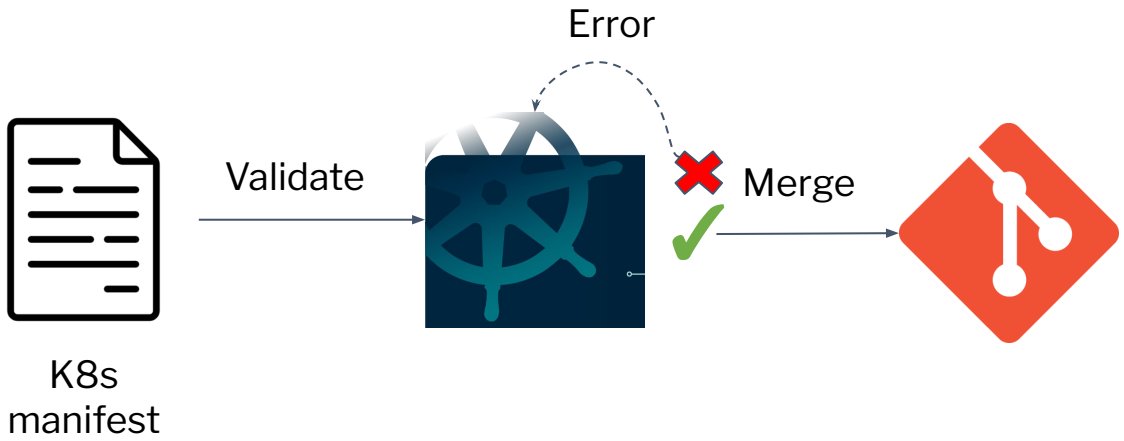
The New York Times

Kubeconform



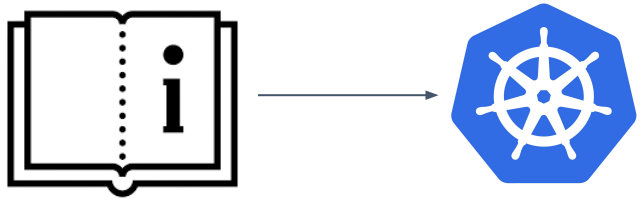
A validation tool for Kubernetes manifests and custom resources that can be incorporated into CI or used locally.

https://github.com/yannh/kubeconform



Custom Resource Definition

A Kubernetes API extension that enables users to define their own resource types and controllers. CRDs allow for the creation of custom resources, like Argo Applications or Projects, and controllers to manage them, extending Kubernetes' core capabilities to better suit specific use cases.



Kubernetes Resource Example

Deprecated Version

```yaml
# v1beta manifest
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: example
spec:
  backend:
    serviceName: default-backend
    servicePort: 80
  rules:
  - http:
      paths:
      - path: /testpath
        backend:
          serviceName: test
          servicePort: 80
```

Stable Version

```yaml
# v1 manifest
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example
spec:
  defaultBackend:
    service:
      name: default-backend
      port:
        number: 80
  rules:
  - http:
      paths:
      - path: /testpath
        pathType: ImplementationSpecific
        backend:
          service:
            name: test
            port:
              number: 80
```

Kubernetes Resource Example

Updating `apiVersion` alone

```yaml
# v1beta manifest
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example
spec:
  backend:
    serviceName: default-backend
    servicePort: 80
  rules:
  - http:
      paths:
      - path: /testpath
        backend:
          serviceName: test
          servicePort: 80
```

```
$ kubeconform \
--kubernetes-version=1.24.0 \
-schema-location default \
v1beta1-to-v1-ingress.yaml
```

```
v1beta1-to-v1-ingress.yaml - Ingress example is invalid: problem validating schema.
Check JSON formatting: jsonschema: '/spec/rules/0/http/paths/0' does not validate wi
th https://raw.githubusercontent.com/yannh/kubernetes-json-schema/master/v1.24.0-sta
ndalone/ingress-networking-v1.json#/properties/spec/properties/rules/items/propertie
s/http/properties/paths/items/required: missing properties: 'pathType'
$
```

Custom Resource Definition


```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: argocd-dev
  namespace: default
  finalizers:
  - resources-finalizer.argocd.argoproj.io
spec:
  destination:
    namespace: default
    name: in-cluster
  project: default
  source:
    path: argo-cd/overlays/dev
    repoURL: https://github.com/org/repo
    targetRevision: HEAD
    syncPolicy:
      automated:
        prune: true
```

Can you find the bug?

Custom Resource Definition


```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: argocd-dev
  namespace: default
  finalizers:
  - resources-finalizer.argocd.argoproj.io
spec:
  destination:
    namespace: default
    name: in-cluster
  project: default
  source:
    path: argo-cd/overlays/dev
    repoURL: https://github.com/org/repo
    targetRevision: HEAD
    syncPolicy:
      automated:
        prune: true
```

```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: argocd-dev
  namespace: default
  finalizers:
  - resources-finalizer.argocd.argoproj.io
spec:
  destination:
    namespace: default
    name: in-cluster
  project: default
  source:
    path: argo-cd/overlays/dev
    repoURL: https://github.com/org/repo
    targetRevision: HEAD
  syncPolicy:
    automated:
      prune: true
```

Custom Resource Definitions

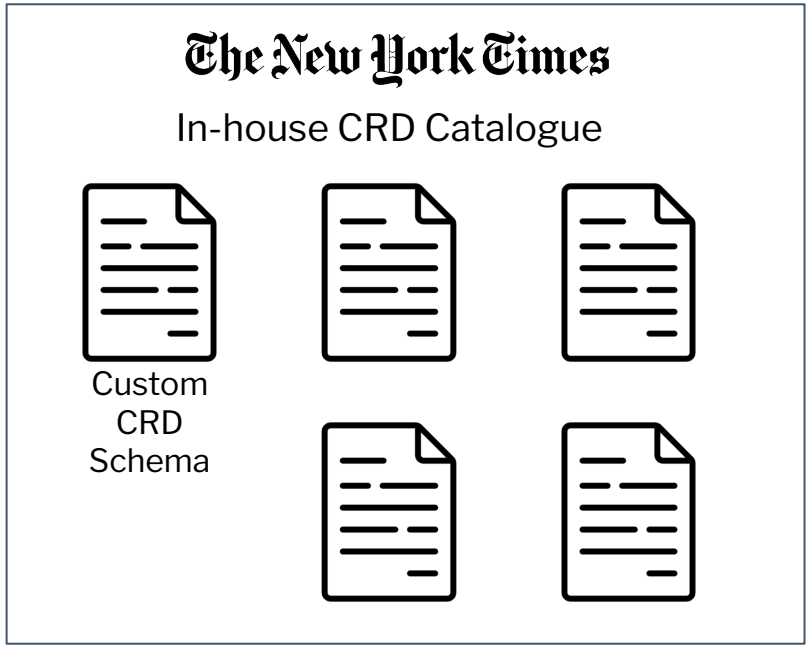
Over 300 popular Kubernetes CRDs (CustomResourceDefinition) in JSON schema format!

CRD Schema Catalog: https://github.com/datreeio/CRDs-catalog

```
# Example of referencing schemas from github repository
$ kubeconform -schema-location default -schema-location
'https://raw.githubusercontent.com/datreeio/CRDs-catalog/main/{{.Group}}/{{.ResourceKind}}_{
{.ResourceAPIVersion}}.json' [MANIFEST]
```

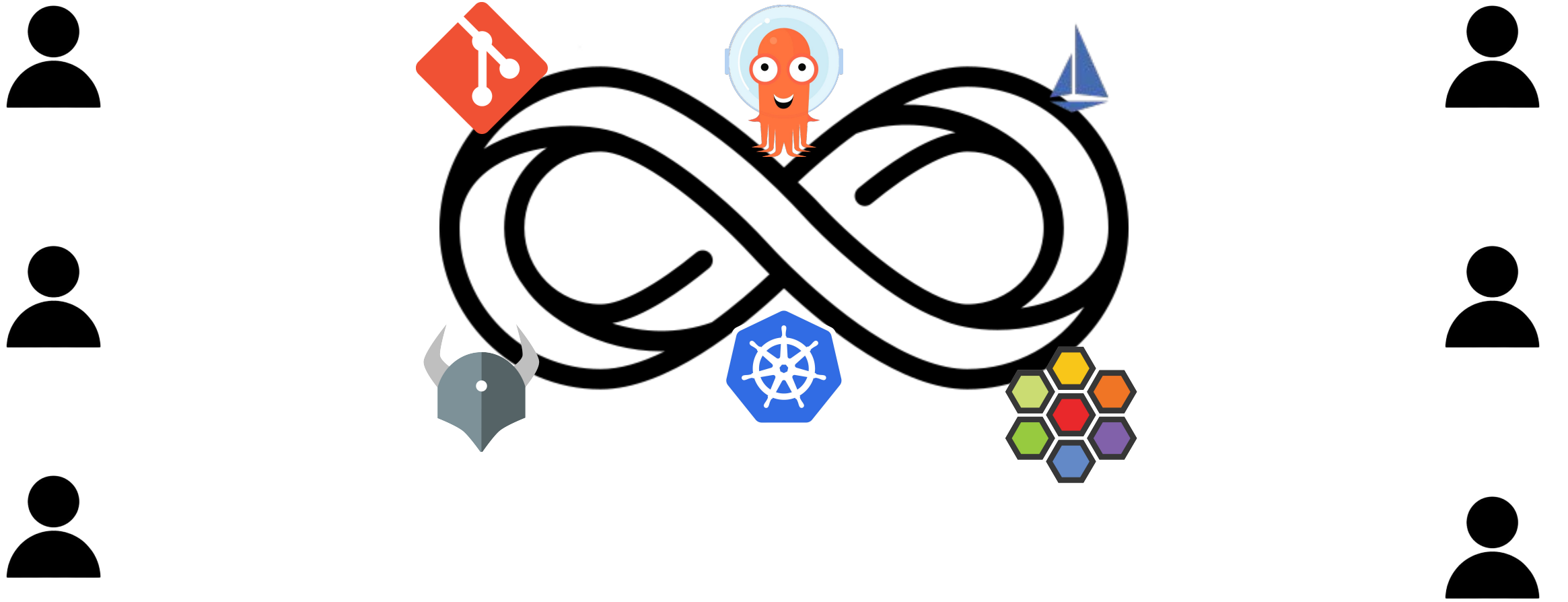
The New York Times

Custom Resource Definitions



Summary

Summary



The New York Times

THANK YOU

Interested in learning more about The New York Times' Internal Development Platform implementation?

- Scaling Argo Security and Multi-Tenancy in AWS EKS at the New York Times - David Grizzanti & Luke Philips, The New York Times
- Designing and Securing a Multi-Tenant Runtime Environment at the New York Times - Ahmed Bebars, The New York Times

Keep an eye out for recordings of these talks

Session QR Codes will be
sent via email before the event

Please scan the QR Code above
to leave feedback on this session