Kubescape

Kubernetes security scanner

CONCEPTS

Scan manifests before deploy and the live cluster afterwards: the same controls run on both.

Target cluster, files, Helm, image

↓ load

Framework NSA, MITRE, CIS: sets of controls

↓ evaluate

Controls C-xxxx checks, exceptions applied

↓ score

Report table, JSON, SARIF, HTML

SCAN TARGETS

`kubescape scan`
Scan the cluster in your kubeconfig

`kubescape scan` *`<path>`*
Scan local YAML files or a directory

`kubescape scan` *`<chart-dir>`*
Scan a Helm chart directory

`kubescape scan` *`<kustomize-dir>`*
Scan a Kustomize directory

`kubescape scan image` *`<image>`*
Scan an image for vulnerabilities

`--include-namespaces` *`<list>`*
Scan only these namespaces

`--exclude-namespaces` *`<list>`*
Skip these namespaces

FRAMEWORKS AND CONTROLS

`kubescape scan framework` *`<name>`*
Run all controls of a framework

`kubescape scan framework nsa`
NSA-CISA hardening guide

`kubescape scan framework mitre`
MITRE ATT&CK for Kubernetes

`kubescape scan control` *`<id>`*
Run a single control, e.g. C-0017

FRAMEWORKS AND CONTROLS (cont.)

`--controls-config` *`<file>`*
Scan with edited control parameters

VIEWS AND THRESHOLDS

`--verbose`
Show each failing resource and the fix

`--compliance-threshold` *`<pct>`*
Exit 1 if compliance is below pct

`--severity-threshold` *`<level>`*
Fail on controls at this severity or above

Gate CI on --compliance-threshold: it is the documented, non-deprecated pass/fail switch.

OUTPUT

`--format json --output` *`<file>`*
Machine-readable report

`--format sarif --output` *`<file>`*
SARIF for code scanning

`--format junit --output` *`<file>`*
JUnit XML for CI test views

EXCEPTIONS

`--exceptions` *`<file>`*
Apply accepted-risk exceptions

`posturePolicies`
Controls or frameworks to exclude

`resources`
Match by kind, name, namespace or labels

FIX, PATCH AND POLICY

`kubescape fix` *`<report.json>`*
Auto-fix manifests from a JSON report

`kubescape fix` *`<report.json>`* **`--dry-run`**
Preview fixes without writing