Kubescape Kubernetes security scanner

START HERE

`kubescape scan`
Scan the current cluster

`kubescape scan framework nsa`
Check against NSA hardening

`kubescape scan <path>`
Scan YAML, Helm or Kustomize

`kubescape scan image <image>`
Scan a container image

`kubescape scan control <id>`
Run one control

`--verbose`
Show failing resources and fixes

`--format json --output <file>`
Write a report file

`--compliance-threshold <n>`
Fail CI below this score

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

CONCEPTS

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

Exceptions show up as exclude, not fail, so accepted risk stays visible in the report.

The kubescape operator (in-cluster) adds continuous and vulnerability scanning; the CLI covers one-off scans.

SCAN TARGETS

`kubescape scan`
Scan the cluster in your kubeconfig

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

`kubescape scan <git-url>`
Scan a Git repository

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

`--values <file>`
Use a Helm values file when rendering

`--set <key=value>`
Override a Helm value

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

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

`kubescape scan image --username <u> --password <p> <image>`
Scan from a private registry

`--kubeconfig <file>`
Use another kubeconfig

`--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

FRAMEWORKS AND CONTROLS (cont.)

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

`kubescape scan framework nsa,mitre`
Run several frameworks at once

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

`kubescape scan control "<name>"`
Select a control by its name

`kubescape list controls`
List available controls

`kubescape list frameworks`
List available frameworks

`kubescape download controls-inputs`
Fetch the tunable control parameters

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

Framework names are matched case-insensitively; list them for the exact IDs, e.g. cis versions.

VIEWS AND THRESHOLDS

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

`--view security`
Default view when scanning files

`--view resource`
Group results by resource

`--view control`
Group results by control

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

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

`--fail-threshold <pct>`
Exit 1 if risk is above pct (deprecated)

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

Documented for local files and Git targets only.

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

`--format html --output <file>`
Shareable HTML report

`--format pdf --output <file>`
PDF report

`--format markdown --output <file>`
Markdown report

`--format csv --output <file>`
CSV report

`--format json,html --output <prefix>`
Several formats in one run

`--format-version v2`
Recommended JSON schema version

`--hide`
Pseudonymize sensitive metadata

EXCEPTIONS

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

`posturePolicies`
Controls or frameworks to exclude

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

Exception fields name, kind, namespace and controlID accept regex; resource labels do not.

Exclude a control in one namespace

```
[{"name": "allow-hostpath",
  "policyType":
"postureExceptionPolicy",
  "actions": ["alertOnly"],
  "resources":
[{"designatorType":
"Attributes",
    "attributes": {"namespace":
"monitoring"}}],
  "posturePolicies":
[{"controlID": "C-0060"}]}]
```

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

`kubescape fix <report.json> --no-confirm`
Apply all fixes without asking

`kubescape patch --image <image>`
Patch OS vulns in an image

Needs a running buildkitd.

`kubescape vap deploy-library`
Print the ValidatingAdmissionPolicy library

CONFIG AND OFFLINE

`kubescape config view`
Show stored configuration

`kubescape download artifacts --output <dir>`
Save frameworks and controls for offline use

`--use-artifacts-from <dir>`
Scan using saved artifacts

`kubescape version`
Print version

`kubectl krew install kubescape`
Install as a kubectl plugin

RECIPES

Gate a CI build on compliance

```
kubescape scan framework nsa
<manifests-dir> \
  --compliance-threshold 80 \
  --exceptions exceptions.json
```

Cluster audit with reports

```
kubescape scan framework
nsa,mitre --verbose \
  --format json,html --output
results
```

Scan, fix and re-check manifests

```
kubescape scan <dir> --format
json --output r.json
kubescape fix r.json --dry-run
kubescape fix r.json
kubescape scan <dir>
```

GitHub Action on pull requests

```
- uses: kubescape/github-
action@main
  with:
    frameworks: NSA,MITRE
    severityThreshold: low
    exceptions: exceptions.json
```