Antony Kervazo-Canut

Kubescape for Teenagers

SUITABLE FOR ADULTS

Introduction

Kubescape is the tool of choice for testing the security of Kubernetes environments, offering comprehensive analysis to detect risky configurations according to several security frameworks, including those recommended by the National Security Agency (NSA) and the Cybersecurity and Infrastructure Security Agency (CISA). It is designed to identify vulnerabilities and misconfigurations that could jeopardize the security of containerized applications and Kubernetes infrastructure.

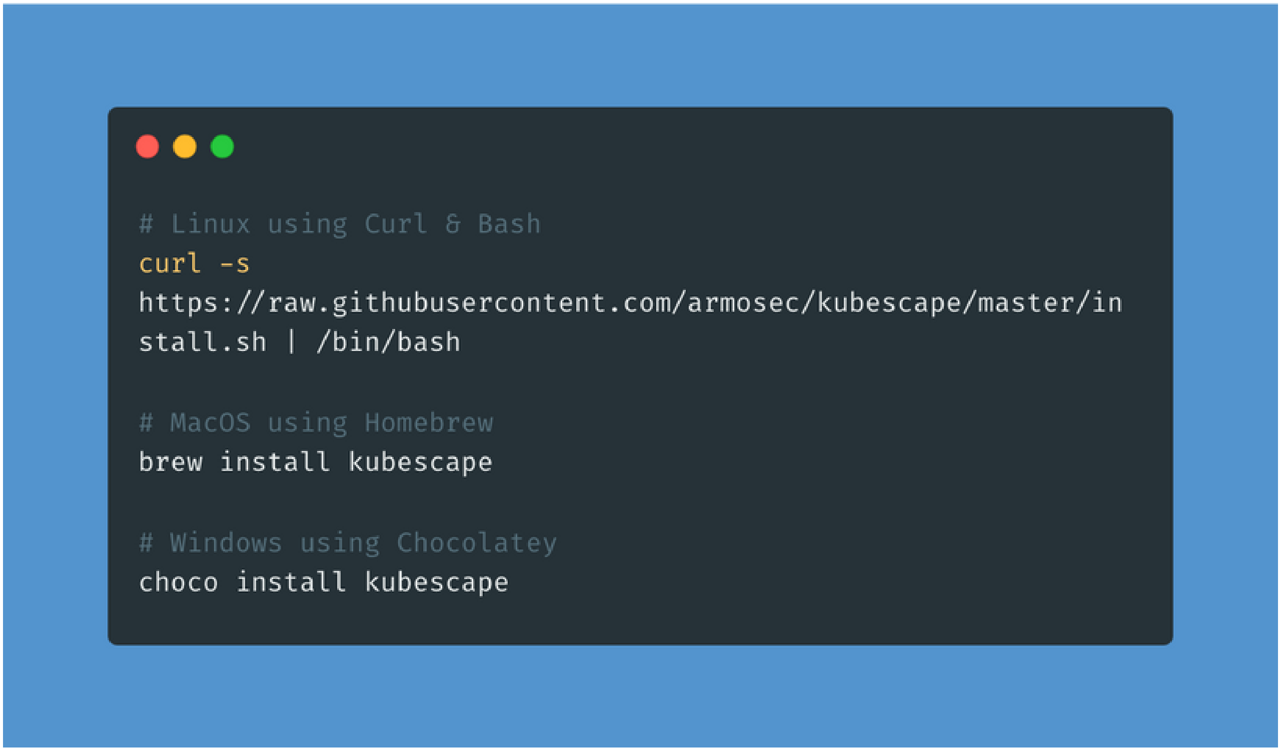
Kubescape was developed by ARMO, a company specializing in the security of cloud-native applications. Over time, the management of the Kubescape project was transferred to the Cloud Native Computing Foundation (CNCF), a foundation under the auspices of the Linux Foundation that supports open-source projects related to cloud computing. This transfer is part of a desire to place Kubescape under open governance and to promote its growth within the cloud-native technology community, ensuring neutrality and increased transparency in its development.

The use of Kubescape is essential for several reasons:

- Enhanced Security: It helps detect and correct risky configurations and vulnerabilities in Kubernetes clusters, contributing to a safer infrastructure.
- Compliance: Kubescape assesses the compliance of your Kubernetes environments against recognized security standards, facilitating the adoption of best practices and regulatory compliance.

Kubescape Installation - Local

Local installation allows running Kubescape from any workstation, offering flexibility for developers and security engineers who analyze multiple clusters or configurations without needing direct access to each cluster.

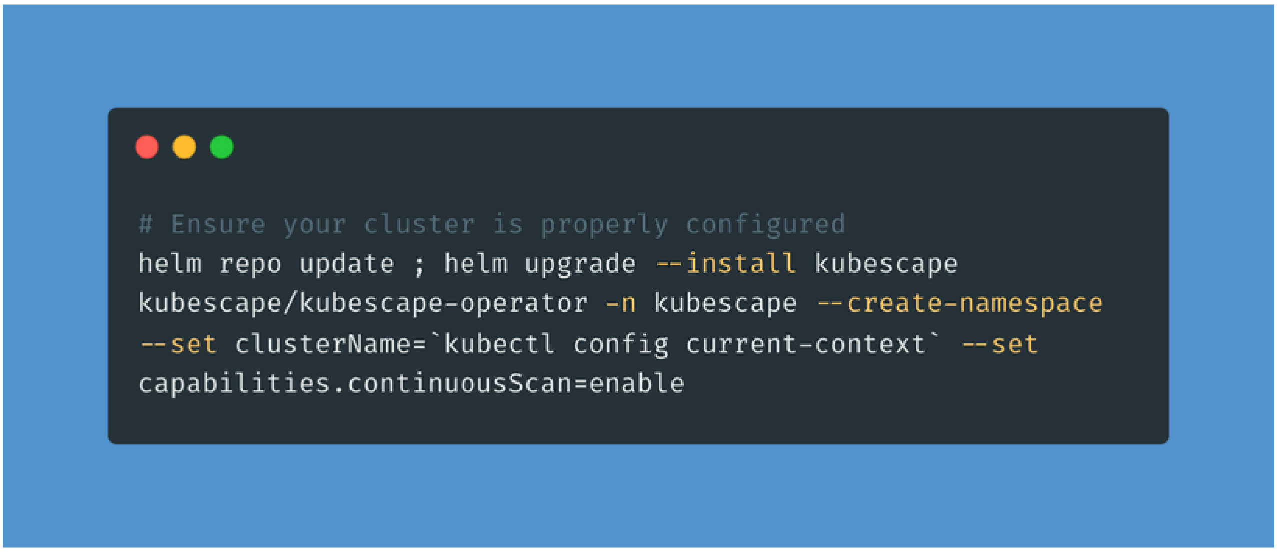


```
# Linux using Curl & Bash
curl -s
https://raw.githubusercontent.com/armosec/kubescape/master/in
stall.sh | /bin/bash

# MacOS using Homebrew
brew install kubescape

# Windows using Chocolatey
choco install kubescape
```

Kubescape Installation - Cluster

Running Kubescape directly on a cluster offers an accurate view of the current security state of the cluster, including live configurations and Kubernetes objects, which might not be available during a local analysis.

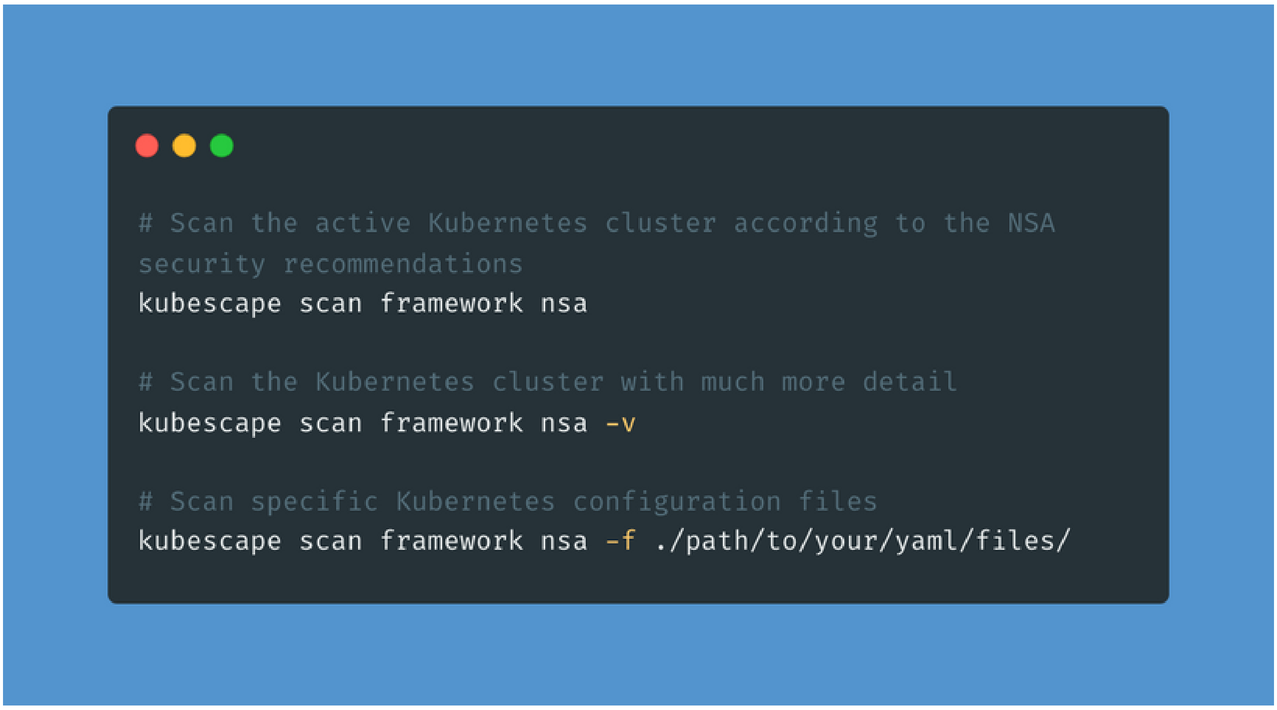
This allows for automating security scans as part of cluster operations, for example, through scheduled jobs or event-based triggers.



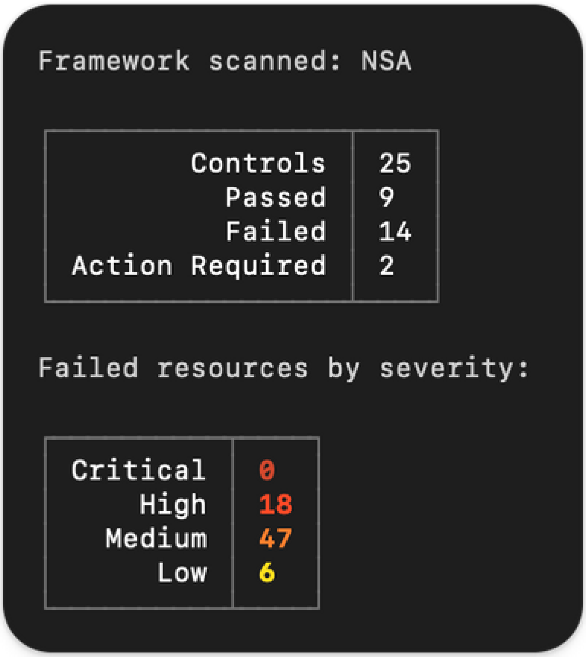
```
# Ensure your cluster is properly configured
helm repo update ; helm upgrade --install kubescape
kubescape/kubescape-operator -n kubescape --create-namespace
--set clusterName=`kubectl config current-context` --set
capabilities.continuousScan=enable
```

Cluster Scan

To start your first scan, you can run Kubescape against your active Kubernetes cluster or against specific configuration files.



```
# Scan the active Kubernetes cluster according to the NSA
security recommendations
kubescape scan framework nsa

# Scan the Kubernetes cluster with much more detail
kubescape scan framework nsa -v

# Scan specific Kubernetes configuration files
kubescape scan framework nsa -f ./path/to/your/yaml/files/
```

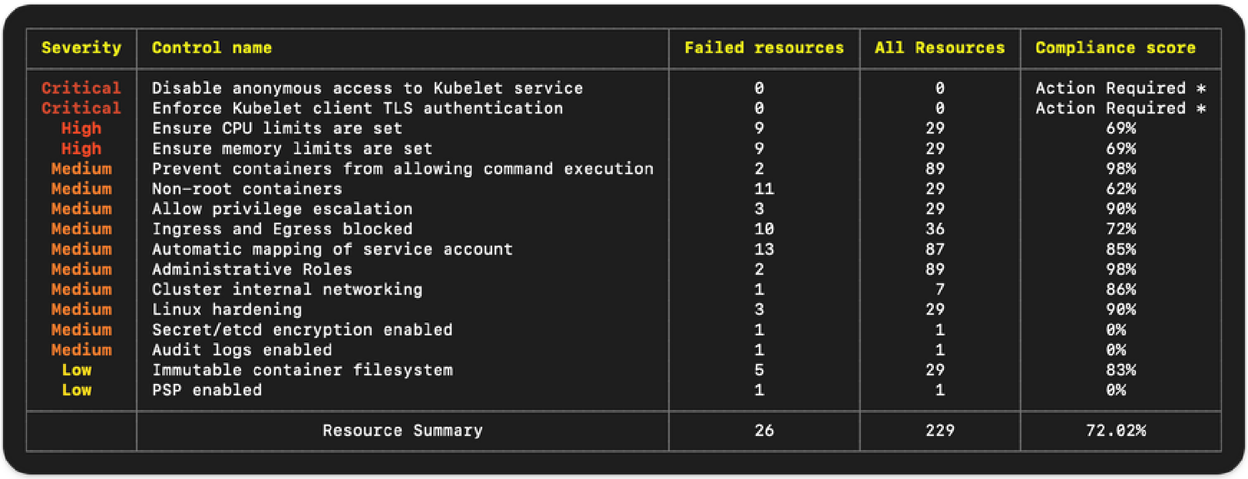


```
Framework scanned: NSA

┌─────────────────┬────┐
│        Controls │ 25 │
│          Passed │ 9  │
│          Failed │ 14 │
│ Action Required │ 2  │
└─────────────────┴────┘

Failed resources by severity:

┌──────────┬────┐
│ Critical │ 0  │
│     High │ 18 │
│   Medium │ 47 │
│      Low │ 6  │
└──────────┴────┘
```

Interpreting the Results

After running Kubescape to analyze the security of your Kubernetes cluster or configuration files, you will receive a detailed report. Understanding and interpreting this report is crucial for improving the security of your environment.

- Compliance Summary: This section provides an overall view of your environment's compliance with the analyzed security standards. It may include an overall compliance score and statistics on passed/failed controls.
- Details of Identified Risks: Here, Kubescape lists all the risks detected during the analysis, categorized by severity (critical, high, medium, low). For each risk, you will find a description of the issue, the potential impact, and recommendations for mitigation or correction.

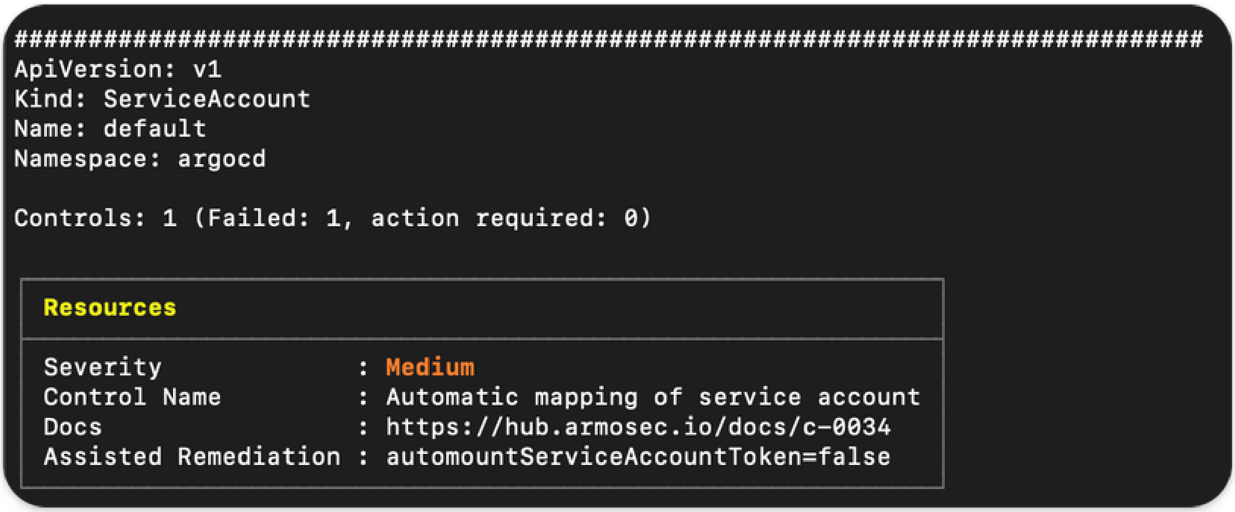


```
 Severity │ Control name                                     │ Failed resources │ All Resources │ Compliance score
──────────┼──────────────────────────────────────────────────┼──────────────────┼───────────────┼──────────────────
 Critical │ Disable anonymous access to Kubelet service      │        0         │       0       │ Action Required *
 Critical │ Enforce Kubelet client TLS authentication        │        0         │       0       │ Action Required *
   High   │ Ensure CPU limits are set                        │        9         │      29       │       69%
   High   │ Ensure memory limits are set                     │        9         │      29       │       69%
  Medium  │ Prevent containers from allowing command execution │      2         │      89       │       98%
  Medium  │ Non-root containers                              │       11         │      29       │       62%
  Medium  │ Allow privilege escalation                       │        3         │      29       │       90%
  Medium  │ Ingress and Egress blocked                       │       10         │      36       │       72%
  Medium  │ Automatic mapping of service account             │       13         │      87       │       85%
  Medium  │ Administrative Roles                             │        2         │      89       │       98%
  Medium  │ Cluster internal networking                      │        1         │       7       │       86%
  Medium  │ Linux hardening                                  │        3         │      29       │       90%
  Medium  │ Secret/etcd encryption enabled                   │        1         │       1       │       0%
  Medium  │ Audit logs enabled                               │        1         │       1       │       0%
   Low    │ Immutable container filesystem                   │        5         │      29       │       83%
   Low    │ PSP enabled                                      │        1         │       1       │       0%
──────────┼──────────────────────────────────────────────────┼──────────────────┼───────────────┼──────────────────
          │               Resource Summary                   │       26         │      229      │     72.02%
```

Interpreting the Results

Start by addressing risks classified as critical or high, as they represent the most serious threats to the security of your environment. Vulnerabilities of medium and low severity can be addressed next, depending on your resources and schedule.

Remediation Planning: For each identified risk, assess the provided recommendations and plan a remediation strategy. This may involve modifying configurations, updating security policies, or revising development practices.



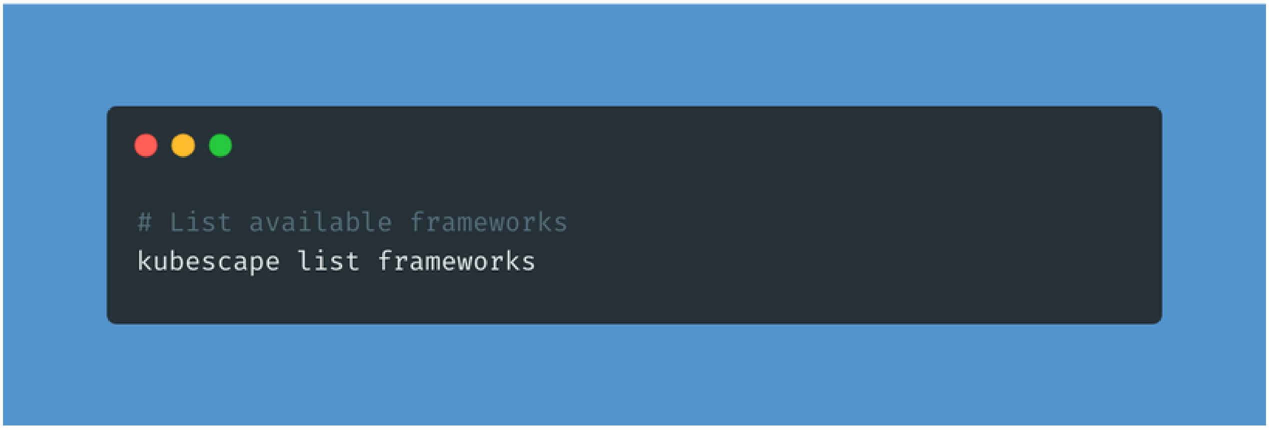
```
################################################################################
ApiVersion: v1
Kind: ServiceAccount
Name: default
Namespace: argocd

Controls: 1 (Failed: 1, action required: 0)


  Resources

  Severity            : Medium
  Control Name        : Automatic mapping of service account
  Docs                : https://hub.armosec.io/docs/c-0034
  Assisted Remediation : automountServiceAccountToken=false
```

Frameworks

Kubescape supports multiple compliance frameworks to assess the security of your Kubernetes environments. These frameworks provide sets of rules and best practices for securing your clusters. Here are a few:

- NSA (National Security Agency) and CISA (Cybersecurity and Infrastructure Security Agency): This framework provides recommendations for securing Kubernetes environments against common threats. It is designed to be applicable to a wide range of organizations.
- CIS Kubernetes Benchmark: Developed by the Center for Internet Security, this benchmark offers detailed guidance on securing Kubernetes clusters, based on recognized industry practices.
- MITRE ATT&CK®: This framework focuses on the tactics and techniques used by adversaries to attack cloud and Kubernetes environments.

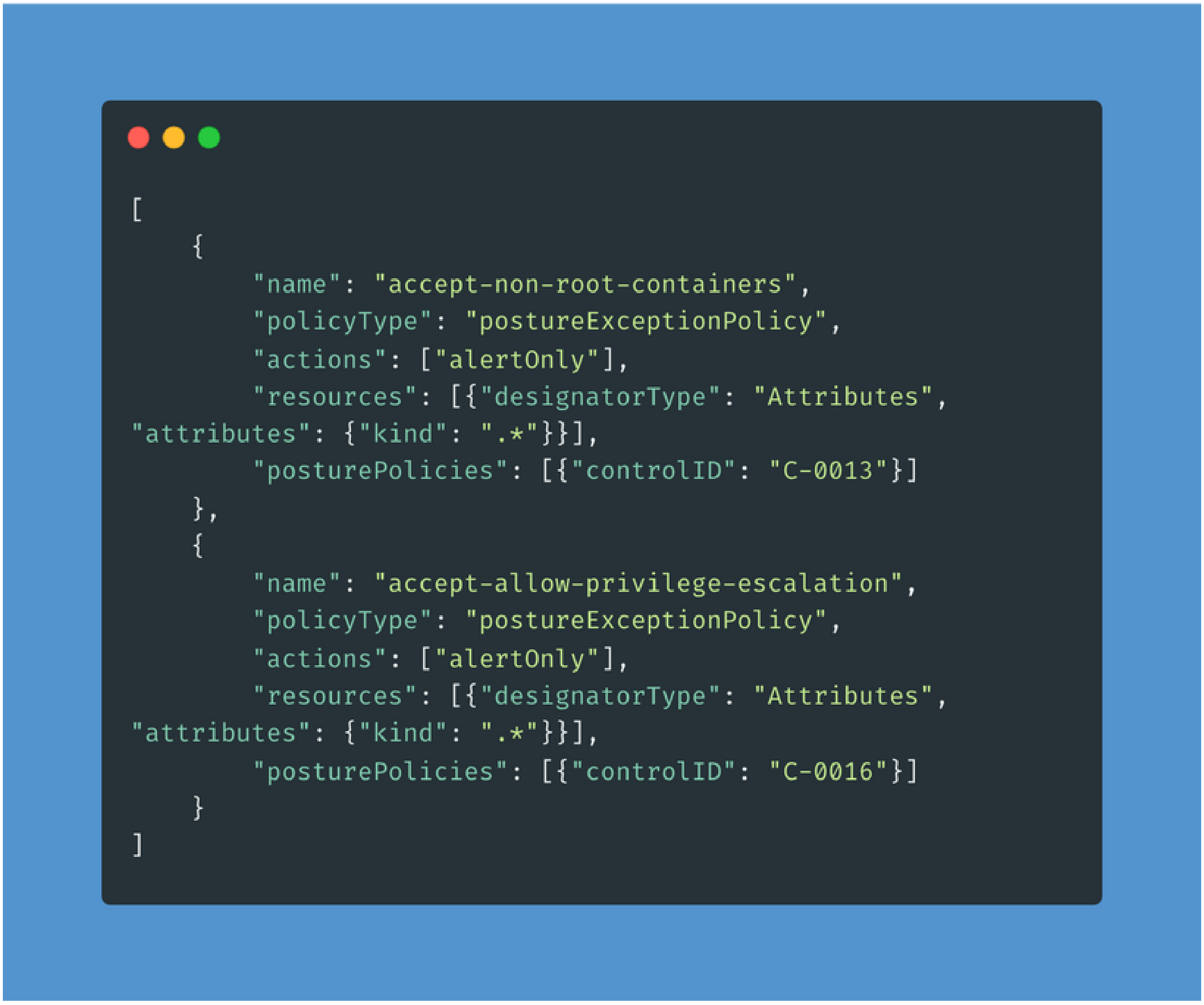
Each framework has its own strengths and focuses on different aspects of Kubernetes security. The choice of framework depends on your specific security objectives, regulatory requirements, and the desired level of security.



```
# List available frameworks
kubescape list frameworks
```

Accepting risk

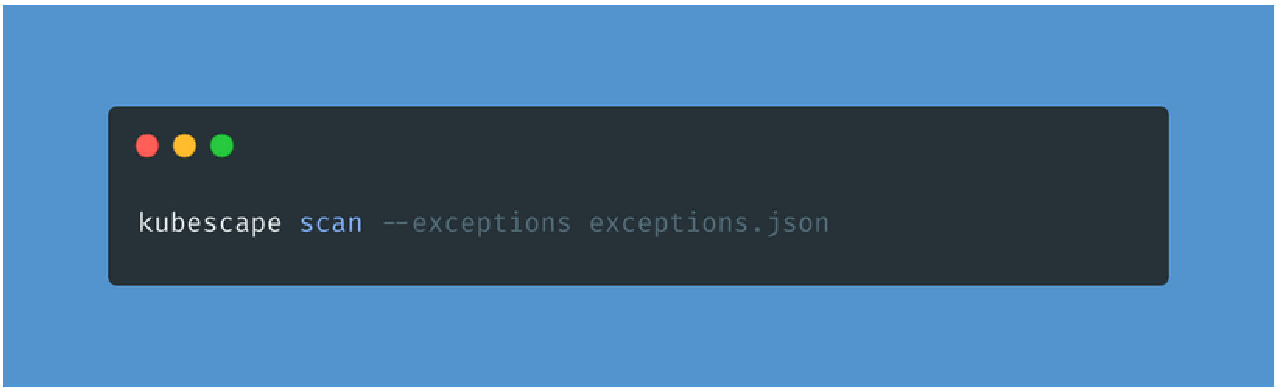
Exception management in Kubescape allows teams to define exception policies for specific security controls, thus offering flexibility in managing identified security risks. This approach is particularly useful for risks that you have chosen to accept due to operational constraints, functional necessities, or when compensatory measures are in place.



```
[
    {
        "name": "accept-non-root-containers",
        "policyType": "postureExceptionPolicy",
        "actions": ["alertOnly"],
        "resources": [{"designatorType": "Attributes",
"attributes": {"kind": ".*"}}],
        "posturePolicies": [{"controlID": "C-0013"}]
    },
    {
        "name": "accept-allow-privilege-escalation",
        "policyType": "postureExceptionPolicy",
        "actions": ["alertOnly"],
        "resources": [{"designatorType": "Attributes",
"attributes": {"kind": ".*"}}],
        "posturePolicies": [{"controlID": "C-0016"}]
    }
]
```

Accepting risk

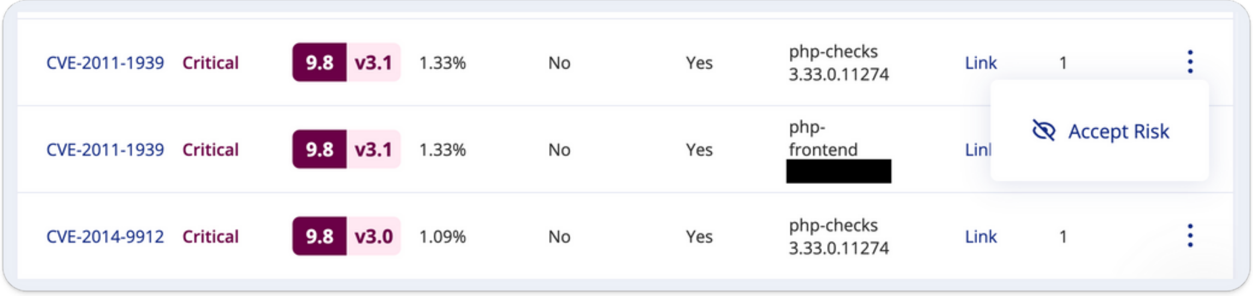
Then, it is used in the following way.



```
kubescape scan --exceptions exceptions.json
```

Using the ARMO Dashboard for risk management involves going directly through the "Vulnerabilities" tab.

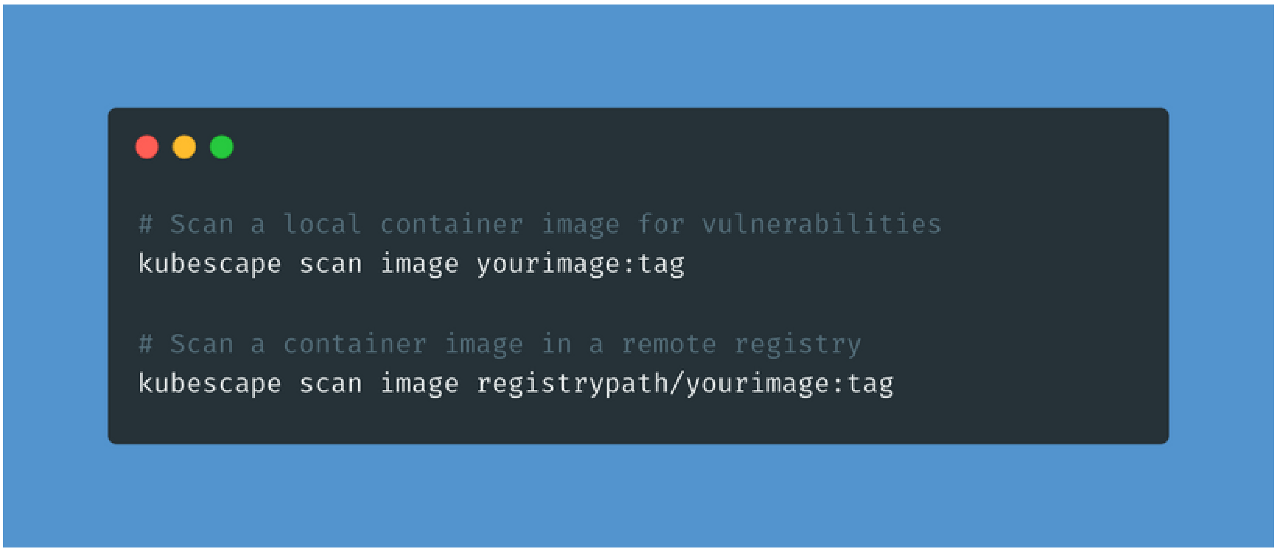
Then, it is possible to set up rules based on these vulnerabilities.



CVE-2011-1939	Critical	9.8	v3.1	1.33%	No	Yes	php-checks 3.33.0.11274	Link	1	⋮
CVE-2011-1939	Critical	9.8	v3.1	1.33%	No	Yes	php-frontend [redacted]	Linl		
CVE-2014-9912	Critical	9.8	v3.0	1.09%	No	Yes	php-checks 3.33.0.11274	Link	1	⋮

Image Scan

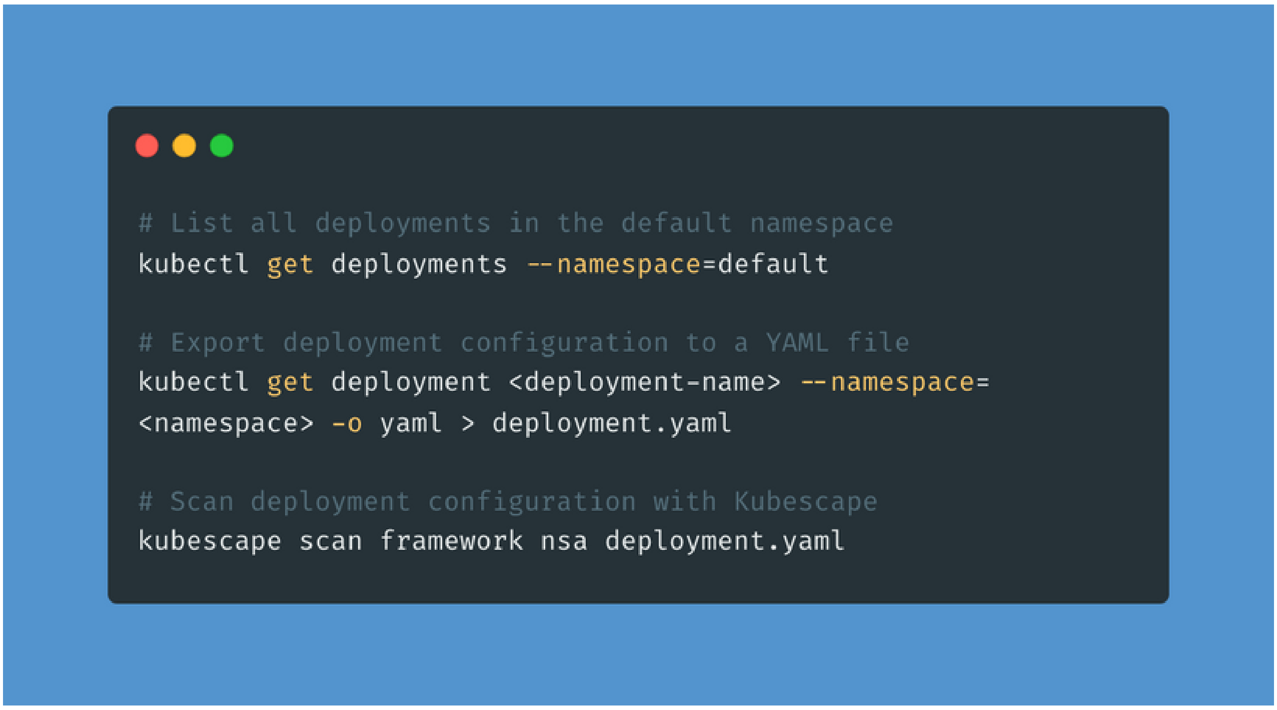
The image scan by Kubescape allows for the identification of known vulnerabilities in container images, based on up-to-date vulnerability databases. This feature helps to detect and correct vulnerabilities before the images are deployed in a production environment.



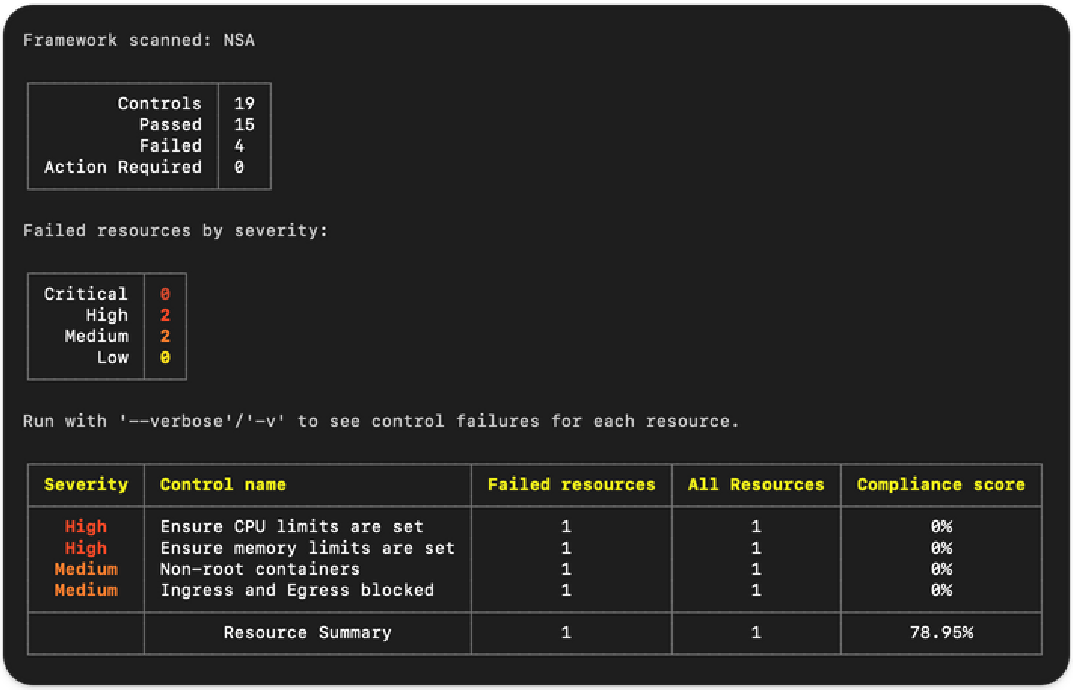
```
# Scan a local container image for vulnerabilities
kubescape scan image yourimage:tag

# Scan a container image in a remote registry
kubescape scan image registrypath/yourimage:tag
```

Deployment Scan

Kubescape can be used for the security analysis of Kubernetes deployments themselves. This is done by focusing on the detection of risky configurations and potential vulnerabilities.



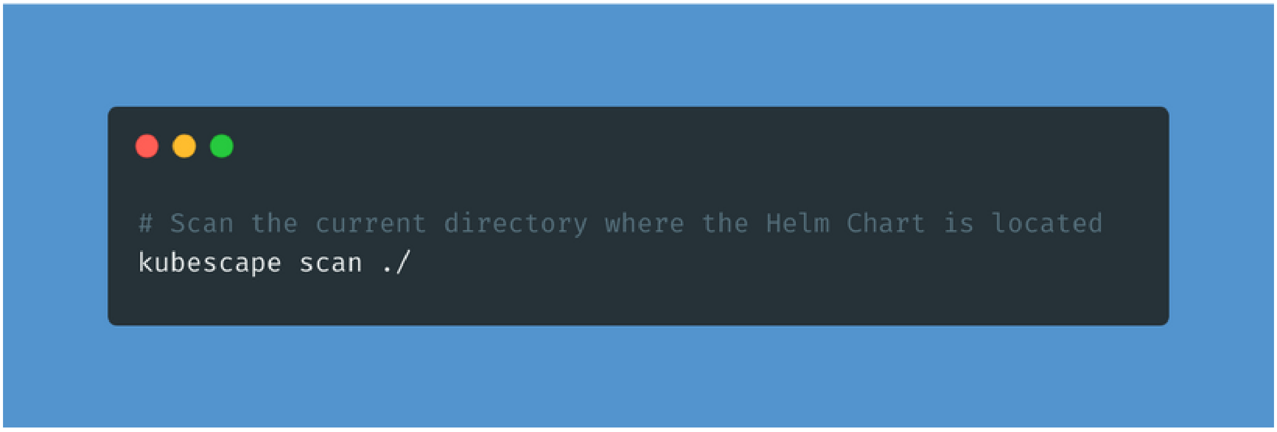
```
# List all deployments in the default namespace
kubectl get deployments --namespace=default

# Export deployment configuration to a YAML file
kubectl get deployment <deployment-name> --namespace=
<namespace> -o yaml > deployment.yaml

# Scan deployment configuration with Kubescape
kubescape scan framework nsa deployment.yaml
```



```
Framework scanned: NSA

┌─────────────────┬────┐
│        Controls │ 19 │
│          Passed │ 15 │
│          Failed │ 4  │
│ Action Required │ 0  │
└─────────────────┴────┘

Failed resources by severity:

┌──────────┬───┐
│ Critical │ 0 │
│     High │ 2 │
│   Medium │ 2 │
│      Low │ 0 │
└──────────┴───┘

Run with '--verbose'/'-v' to see control failures for each resource.

┌──────────┬──────────────────────────────┬──────────────────┬───────────────┬──────────────────┐
│ Severity │ Control name                 │ Failed resources │ All Resources │ Compliance score │
├──────────┼──────────────────────────────┼──────────────────┼───────────────┼──────────────────┤
│   High   │ Ensure CPU limits are set    │        1         │       1       │        0%        │
│   High   │ Ensure memory limits are set │        1         │       1       │        0%        │
│  Medium  │ Non-root containers          │        1         │       1       │        0%        │
│  Medium  │ Ingress and Egress blocked   │        1         │       1       │        0%        │
├──────────┼──────────────────────────────┼──────────────────┼───────────────┼──────────────────┤
│          │       Resource Summary       │        1         │       1       │      78.95%      │
└──────────┴──────────────────────────────┴──────────────────┴───────────────┴──────────────────┘
```

Helm Chart Scan

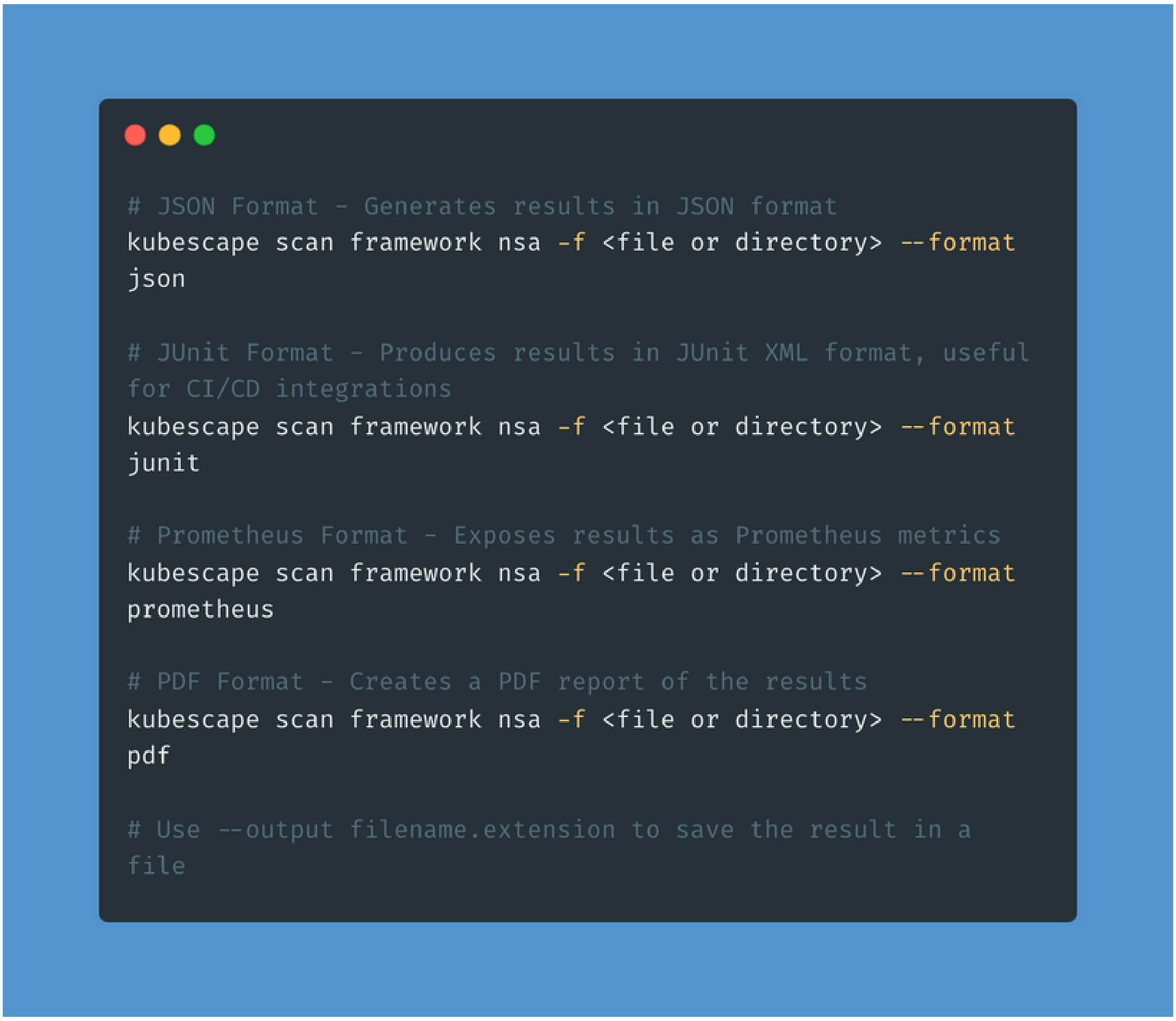
Direct analysis of Helm charts with Kubescape is a valuable feature that allows for assessing the security of Helm packages before their deployment in a Kubernetes cluster. This approach facilitates the identification and correction of vulnerabilities or risky configurations within the Helm charts themselves, thus strengthening the security posture from the beginning of the deployment cycle.



```
# Scan the current directory where the Helm Chart is located
kubescape scan ./
```

Result Formats

These commands offer great flexibility for analysis and presenting results, allowing you to choose the format that best suits your needs, whether for direct review, integration into continuous integration/continuous deployment (CI/CD) pipelines, or for generating reports for stakeholders.

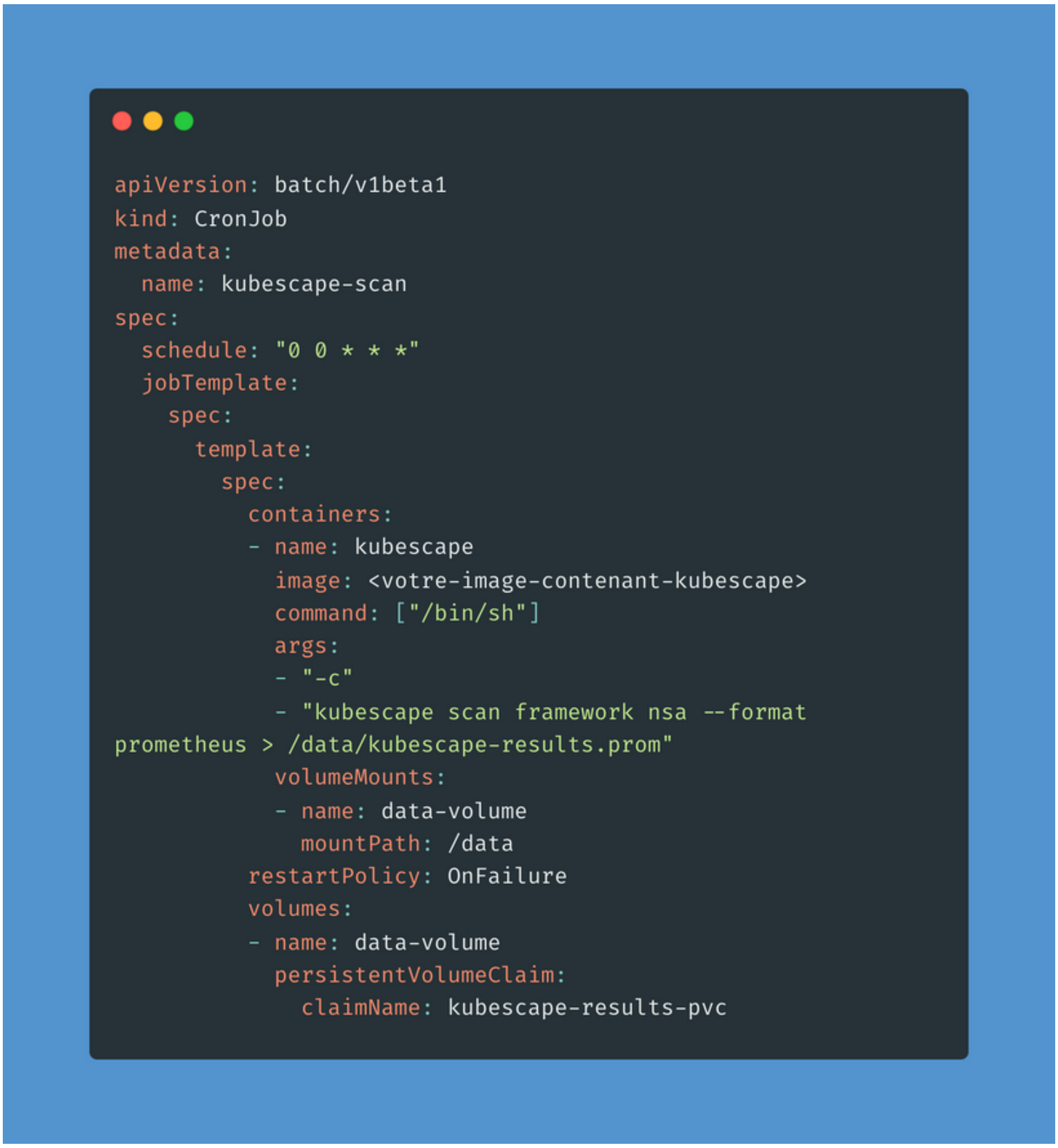


```
# JSON Format - Generates results in JSON format
kubescape scan framework nsa -f <file or directory> --format
json

# JUnit Format - Produces results in JUnit XML format, useful
for CI/CD integrations
kubescape scan framework nsa -f <file or directory> --format
junit

# Prometheus Format - Exposes results as Prometheus metrics
kubescape scan framework nsa -f <file or directory> --format
prometheus

# PDF Format - Creates a PDF report of the results
kubescape scan framework nsa -f <file or directory> --format
pdf

# Use --output filename.extension to save the result in a
file
```

Cluster Monitoring

Continuous monitoring of the security of a Kubernetes cluster is essential for quickly detecting and mitigating vulnerabilities and risky configurations that might emerge over time.

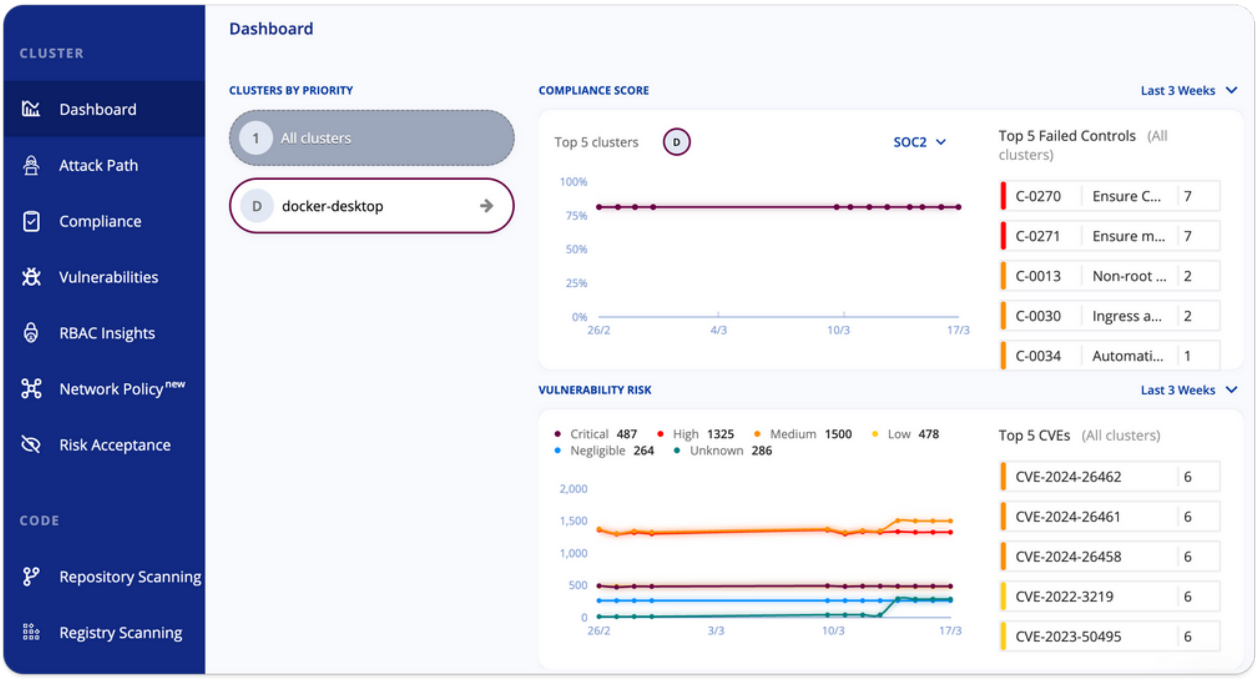
Continuous monitoring with Kubescape can be set up to run at regular intervals, for instance, as scheduled jobs within a Kubernetes cluster, and by sending data to a Prometheus service.



```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: kubescape-scan
spec:
  schedule: "0 0 * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: kubescape
            image: <votre-image-contenant-kubescape>
            command: ["/bin/sh"]
            args:
            - "-c"
            - "kubescape scan framework nsa --format
prometheus > /data/kubescape-results.prom"
            volumeMounts:
            - name: data-volume
              mountPath: /data
          restartPolicy: OnFailure
          volumes:
          - name: data-volume
            persistentVolumeClaim:
              claimName: kubescape-results-pvc
```

ARMO Dashboard

The ARMO Dashboard, developed by the creators of Kubescape, is a powerful tool for the visualization, monitoring, and in-depth analysis of the security scan results performed by Kubescape on your Kubernetes clusters.



Top 5 Failed Controls (All clusters)

C-0270	Ensure C...	7
C-0271	Ensure m...	7
C-0013	Non-root ...	2
C-0030	Ingress a...	2
C-0034	Automati...	1

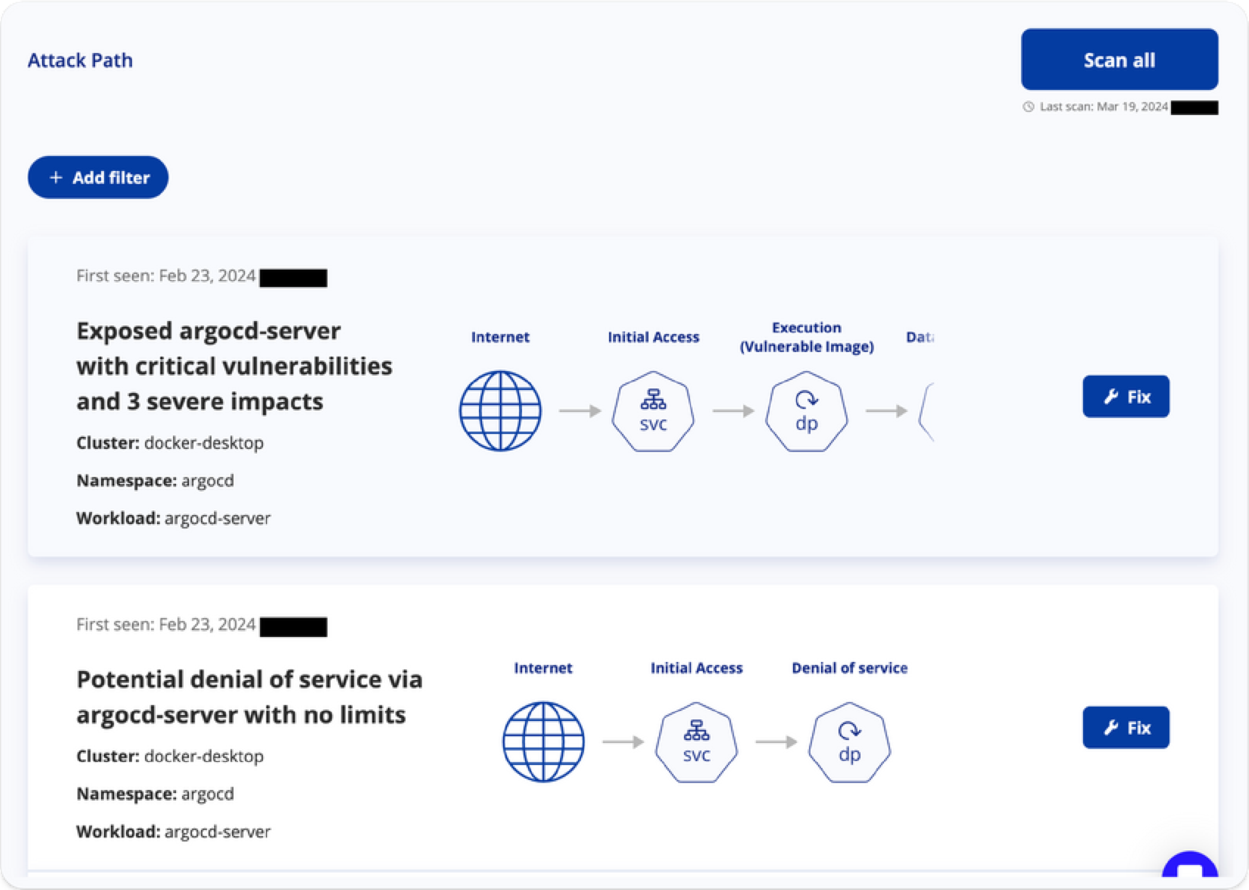
Top 5 CVEs (All clusters)

CVE-2024-26462	6
CVE-2024-26461	6
CVE-2024-26458	6
CVE-2022-3219	6
CVE-2023-50495	6

The ARMO Dashboard enhances Kubescape's capabilities by offering a rich platform for visualizing, monitoring, and analyzing security risks in your Kubernetes environments. By integrating Kubescape with the ARMO Dashboard, security and DevOps teams can work more proactively to identify, understand, and rectify vulnerabilities, thereby strengthening the security of their deployments.

Attack Path

The Attack Path feature of the ARMO Dashboard helps to visualize weaknesses that malicious actors could exploit in Kubernetes clusters. By identifying these paths, ARMO highlights the specific steps where attacks can be blocked and guides engineers through the remediation steps. This facilitates an understanding of the risks and the implementation of effective countermeasures to strengthen the security of Kubernetes deployments.



Last scan: Mar 19, 2024 [redacted]

First seen: Feb 23, 2024 [redacted]

Exposed argocd-server with critical vulnerabilities and 3 severe impacts

Cluster: docker-desktop

Namespace: argocd

Workload: argocd-server

First seen: Feb 23, 2024 [redacted]

Potential denial of service via argocd-server with no limits

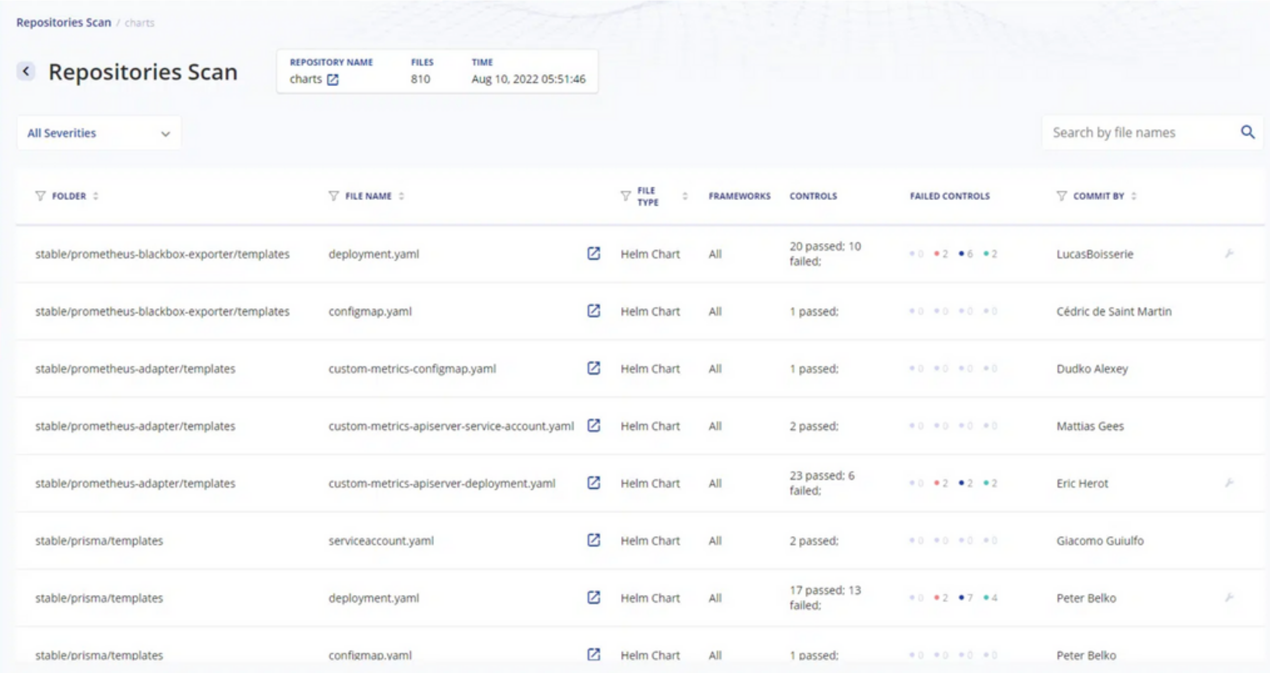
Cluster: docker-desktop

Namespace: argocd

Workload: argocd-server

Repository Scanning

Scanning code repositories and container image registries by ARMO allows identifying and remediating vulnerabilities before deployment. This approach integrates security from the early stages of development, analyzing both source code and container images for security flaws. By focusing on these two critical elements, ARMO helps create a solid line of defense against potential attacks, ensuring robust security from the beginning of the application lifecycle.



Repositories Scan / charts

Repositories Scan

REPOSITORY NAME	FILES	TIME
charts	810	Aug 10, 2022 05:51:46

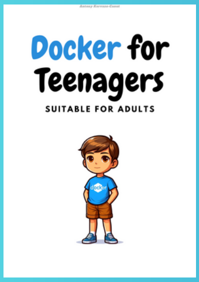
All Severities

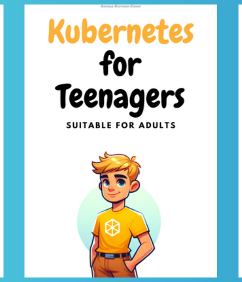
Search by file names

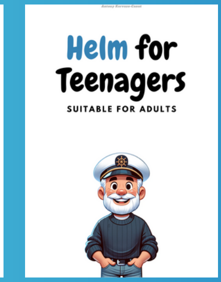
FOLDER	FILE NAME	FILE TYPE	FRAMEWORKS	CONTROLS	FAILED CONTROLS	COMMIT BY
stable/prometheus-blackbox-exporter/templates	deployment.yaml	Helm Chart	All	20 passed: 10 failed:	● 0 ● 2 ● 6 ● 2	LucasBoisserie
stable/prometheus-blackbox-exporter/templates	configmap.yaml	Helm Chart	All	1 passed:	● 0 ● 0 ● 0 ● 0	Cédric de Saint Martin
stable/prometheus-adapter/templates	custom-metrics-configmap.yaml	Helm Chart	All	1 passed:	● 0 ● 0 ● 0 ● 0	Dudko Alexey
stable/prometheus-adapter/templates	custom-metrics-apiserver-service-account.yaml	Helm Chart	All	2 passed:	● 0 ● 0 ● 0 ● 0	Mattias Gees
stable/prometheus-adapter/templates	custom-metrics-apiserver-deployment.yaml	Helm Chart	All	23 passed: 6 failed:	● 0 ● 2 ● 2 ● 2	Eric Herot
stable/prisma/templates	serviceaccount.yaml	Helm Chart	All	2 passed:	● 0 ● 0 ● 0 ● 0	Giacomo Guiulfo
stable/prisma/templates	deployment.yaml	Helm Chart	All	17 passed: 13 failed:	● 0 ● 2 ● 7 ● 4	Peter Belko
stable/prisma/templates	configmap.yaml	Helm Chart	All	1 passed:	● 0 ● 0 ● 0 ● 0	Peter Belko

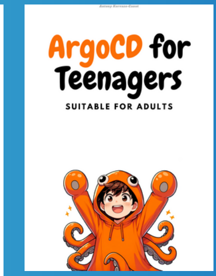
In the same collection

Container Orchestration and Management


Docker for Teenagers
SUITABLE FOR ADULTS

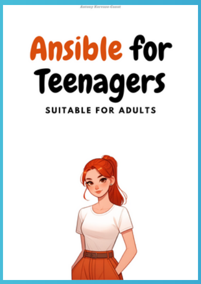

Kubernetes for Teenagers
SUITABLE FOR ADULTS


Helm for Teenagers
SUITABLE FOR ADULTS

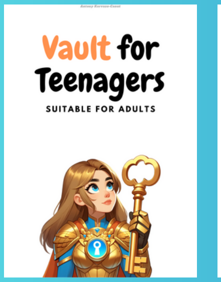

ArgoCD for Teenagers
SUITABLE FOR ADULTS

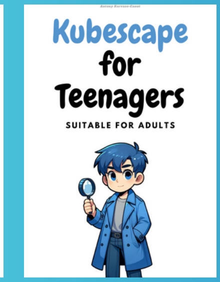
Infrastructure as Code

Terraform for Teenagers
SUITABLE FOR ADULTS


Ansible for Teenagers
SUITABLE FOR ADULTS

Security & Secrets Management


Vault for Teenagers
SUITABLE FOR ADULTS


Kubescape for Teenagers
SUITABLE FOR ADULTS

ANTONYCANUT

ANTONY KERVAZO-CANUT