Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Kubernetes
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

Kubernetes static code analysis

Unique rules to find Security Hotspots in your KUBERNETES code

All rules 7 | Security Hotspot 6 | Code Smell 1

Tags

Search by name...

Mounting sensitive file system paths is security-sensitive
Security Hotspot

Using host operating system namespaces is security-sensitive
Security Hotspot

Allowing process privilege escalations is security-sensitive
Security Hotspot

Exposing Docker sockets is security-sensitive
Security Hotspot

Running containers in privileged mode is security-sensitive
Security Hotspot

Setting capabilities is security-sensitive
Security Hotspot

Kubernetes parsing failure
Code Smell

Allowing process privilege escalations is security-sensitive

Analyze your code

Security Hotspot Major cwe

Allowing process privilege escalations exposes the Pod to attacks that exploit setuid binaries.

This field directly controls whether the `no_new_privs` flag is set in the container process.
When this flag is enabled, binaries configured with setuid or setgid bits cannot change their runtime uid or gid: Potential attackers must rely on other privilege escalation techniques to successfully operate as root on the Pod.

Depending on how resilient the Kubernetes cluster and Pods are, attackers can extend their attack to the cluster by compromising the nodes from which the cluster started the Pod.

The `allowPrivilegeEscalation` field should not be set to true unless the Pod's risks related to setuid or setgid bits have been mitigated.

Ask Yourself Whether

- This Pod is accessible to people who are not administrators of the Kubernetes cluster.
- This Pod contains binaries with setuid or setgid capabilities.

There is a risk if you answered yes to all of these questions.

Recommended Secure Coding Practices

Disable privilege escalation.

Sensitive Code Example

```
apiVersion: v1
kind: Pod
metadata:
  name: example
spec:
  containers:
    - name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
          protocol: TCP
      securityContext:
        allowPrivilegeEscalation: true # Sensitive
```

Compliant Solution

```
apiVersion: v1
kind: Pod
metadata:
  name: example
spec:
  containers:
    - name: web
      image: nginx
      ports:
```

```
      - name: web
        containerPort: 80
        protocol: TCP
    securityContext:
      allowPrivilegeEscalation: false
```

See

- MITRE, CWE-284 - Improper Access Control
- Linux Kernel Archives, no_new_privs - Official docs

Available In:

sonarcloud | sonarqube