KubeCon | CloudNativeCon

Europe 2021

Virtual

Enhancing Kubernetes with the Security Profiles Operator

Colleen Murphy & Sascha Grunert

What we will cover in this talk

1. How to enhance default workload security in Kubernetes
2. The current state of the Security Profiles Operator
3. Seeing the operator in action
4. What we plan for the future of the project

How to enhance default workload security in Kubernetes

- Kubernetes does not provide strong security defaults out of the box
- `SecurityContext` holds security configuration that will be applied to a pod or container
- Some security related features like AppArmor are not yet graduated to the `SecurityContext`
- Seccomp did this graduation in v1.19.0

How to enhance default workload security in Kubernetes

Goal 1: Make features easier usable by providing native fields, for example:

```
securityContext:
  seccompProfile:
    type: Localhost  # or RuntimeDefault, Unconfined
    localhostProfile: operator/security-profiles-operator/nginx-1.19.1.json  # relative path
```

Comparing to the former, annotation based syntax:

```
annotations:
  seccomp.security.alpha.kubernetes.io/pod:  # or `container-name` instead of `pod`
    'localhost/operator/security-profiles-operator/nginx-1.19.1.json'
```

How to enhance default workload security in Kubernetes

Goal 2: Increase user adoption of the feature.

- How to provide stronger defaults for seccomp in Kubernetes?
- How to distribute seccomp profiles across thousands of nodes?
- How to create seccomp profiles for my application?

How to enhance default workload security in Kubernetes

Let’s write a Kubernetes Enhancement Proposal?

- Features can be iterated quickly out of the Kubernetes source code tree
- Kubernetes SIGs can sponsor the project, which provides community infrastructure access

How to enhance default workload security in Kubernetes

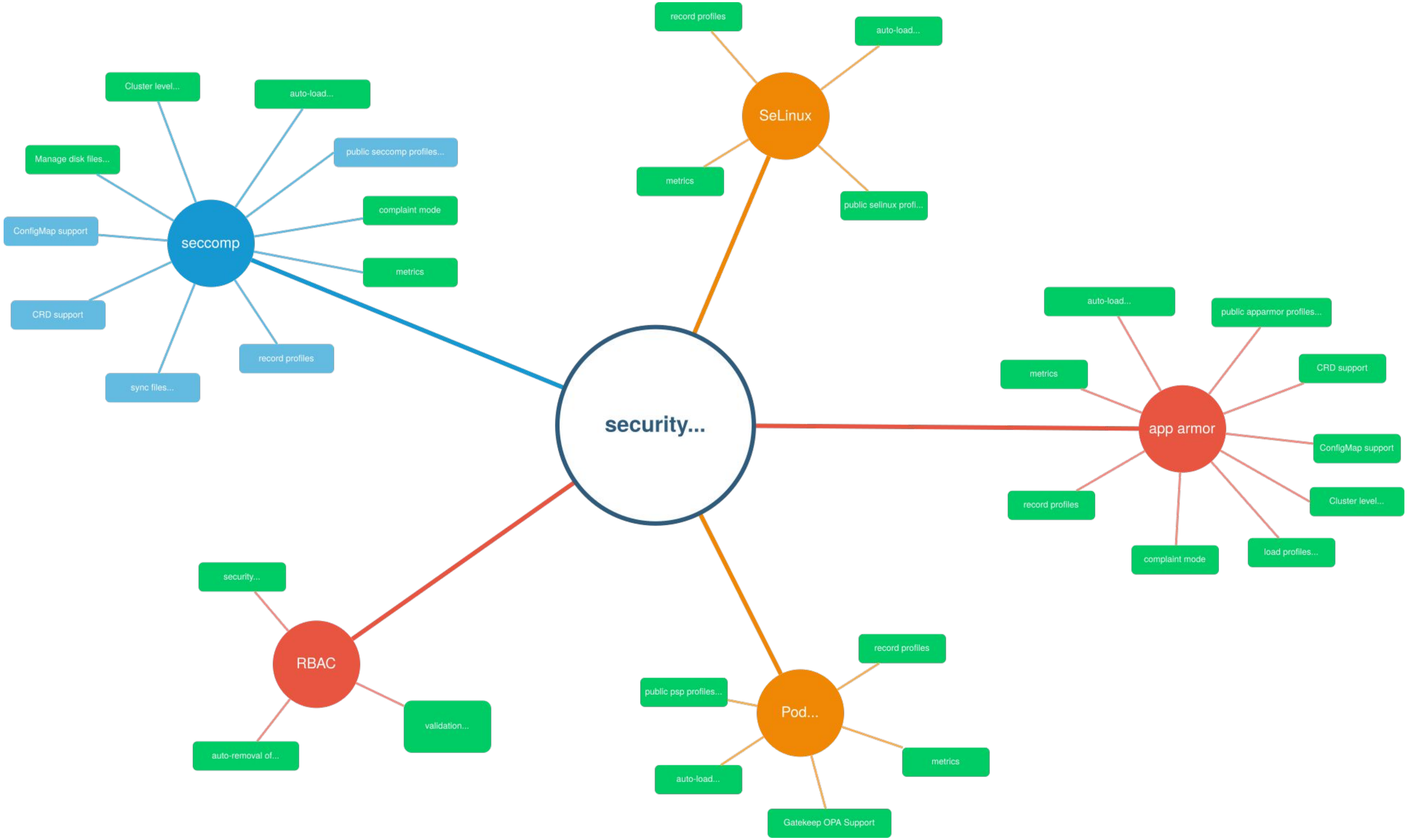
The **Security Profiles Operator** (initially Seccomp Operator) was born!

- Visit sigs.k8s.io/security-profiles-operator
- Project turned 1 year in April
- Idea is to cover common use cases around security profiles

  (seccomp, SELinux and AppArmor) in Kubernetes

  https://sigs.k8s.io/security-profiles-operator/doc/user-stories.md

Enhancing Kubernetes with the Security Profiles Operator



security...

seccomp

Cluster level...

auto-load...

public seccomp profiles...

complaint mode

metrics

record profiles

sync files...

CRD support

ConfigMap support

Manage disk files...

SeLinux

record profiles

auto-load...

metrics

public selinux profi...

app armor

auto-load...

public apparmor profiles...

metrics

CRD support

ConfigMap support

record profiles

Cluster level...

complaint mode

load profiles...

Pod...

record profiles

public psp profiles...

metrics

auto-load...

Gatekeep OPA Support

RBAC

security...

validation...

auto-removal of...

The current state of the Security Profiles Operator

Two profile CRDs

- SeccompProfile
- SelinuxPolicy

Two composition CRDs:

- ProfileBinding
- ProfileRecording

Two operational CRDs:

- SecurityProfilesOperatorDaemon
- SecurityProfileNodeStatus

```
kind: SeccompProfile
metadata:
  name: nginx-1.19.1
spec:
  architectures:
  - SCMP_ARCH_X86_64
  defaultAction: SCMP_ACT_ERRNO
  syscalls:
  - action: SCMP_ACT_ALLOW
    names: ["read", "write", "open"]
```

```
kind: SelinuxPolicy
metadata:
  name: errorlogger
spec:
  apply: true
  policy: |
    (blockinherit container)
    (allow process var_log_t (
           file ( read write open )))
```

Enhancing Kubernetes with the Security Profiles Operator

```yaml
kind: ProfileBinding
metadata:
  name: profile-binding
spec:
  profileRef:
    kind: SeccompProfile
    name: nginx-1.19.1
  image: nginx:1.19.1
```

```yaml
kind: ProfileRecording
metadata:
  name: alpine-recording
spec:
  kind: SeccompProfile
  podSelector:
    matchLabels:
      app: alpine
```

```
kind: SecurityProfilesOperatorDaemon
metadata:
  name: spod
spec:
  enableSelinux: true
  enableLogEnricher: true
```

```
kind: SecurityProfileNodeStatus
metadata:
  name: nginx-1.19.1-worker-1
nodeName: worker-1
status: Installed
```

Log enricher

```
$ grep seccomp /var/log/audit/audit.log | tail -1
audit(1611996299.149:466250) type=seccomp node=kube-worker1 pid=20923 ns=default
pod=my-pod c=container1 exe=/init syscall=epoll_pwait
```

Profile stacking

```
kind: SeccompProfile
metadata:
  name: alpine-profile
spec:
  baseProfileName: base-profile
  defaultAction: SCMP_ACT_ERRNO
  syscalls:
  - action: SCMP_ACT_ALLOW
    names:
    - mkdir
```

Seeing the operator in action (demo!)

- Profile reconciliation
- Profile stacking
- Profile binding
- Profile recording

What we plan for the future of the project

- Node status for profile reconciliation to avoid races
- Further simplify deployment in favor of automatism
- Audit logs for syscall recording
- Full featured SELinux and AppArmor support
- Metrics

KubeCon | CloudNativeCon

Europe 2021

Virtual

❤️

sigs.k8s.io/security-profiles-operator