



KubeCon



CloudNativeCon

North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

Migrating from PodSecurityPolicy

Sam Stoelinga, Google

Tim Allclair, Google

A photograph of a weathered, light-colored stone tombstone in a cemetery. The tombstone has a decorative, arched top and is set on a multi-tiered base. The background is dark and out of focus, showing other graves and trees.

R. I. P.

PodSecurityPolicy

v1.3 - v1.25

What's up with Pod Security Admission?

Stable as of v1.25 🎉

In brief, 1 of 3 predefined standards are applied **per-namespace** as labels:

1. **Privileged** - Anything goes
2. **Baseline** - Allow pod defaults, but not much more
3. **Restricted** - Additional requirements to meet hardening best practices

docs.k8s.io/concepts/security/pod-security-standards/

Agenda

- ☐ Introduction
- ☐ **Demo:** quick and dirty migration
- ☐ Gotchas of the fast path
- ☐ **Demo:** more careful migration
- ☐ Extending beyond Pod Security Admission
- ☐ Conclusion & Resources

Demo!

Fast Path

[illegible]

Not so fast ...

Problem: No representative pods

Examples: On-demand controllers, Scaled-to-zero workloads, Cronjobs, etc.

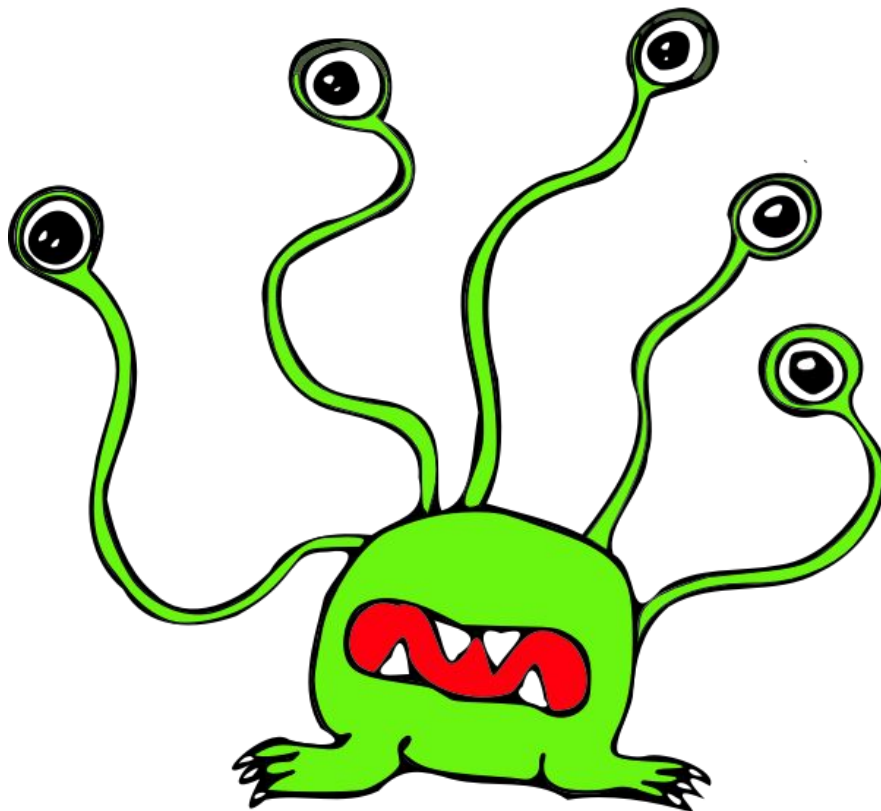
Warn mode: `pod-security.kubernetes.io/warn=$LEVEL`

- User-facing feedback on `kubectl` commands
- Applied to pods AND workloads (Deployment, StatefulSet, etc.)

Audit mode: `pod-security.kubernetes.io/audit=$LEVEL`

- Like warn mode, but captured in audit logs:
`pod-security.kubernetes.io/audit-violations`

Problem: Mutations!



Quiz: Which PSP fields are mutating?

defaultAddCapabilities

requiredDropCapabilities

runAsUser

"Mapping PodSecurityPolicies to Pod Security Standards" -

fsGroup

docs.k8s.io/reference/access-authn-authz/psp-to-pod-security-standards/

readOnlyRootFilesystem

defaultAllowPrivilegeEscalation

allowPrivilegeEscalation

Full list of Mutating PSP fields

"Mapping PodSecurityPolicies to Pod Security Standards" -
docs.k8s.io/reference/access-authn-authz/psp-to-pod-security-standards/

defaultAddCapabilities

requiredDropCapabilities

seLinux

runAsUser

runAsGroup

supplementalGroups

fsGroup

} Only with a
MustRunAs*
strategy

readOnlyRootFilesystem

defaultAllowPrivilegeEscalation

allowPrivilegeEscalation

runtimeClass.defaultRuntimeClassName

Annotations:

seccomp.security.alpha.kubernetes.io/defaultProfileName

apparmor.security.beta.kubernetes.io/defaultProfileName

Demo!

The Hard way

```

$ cat privileged-psp.yaml
apiVersion: v1
kind: PodSecurityPolicy
metadata:
  name: privileged
spec:
  privileged: true
  runAsUser:
    rule: 'MustRunAs'
    ranges:
    - from: 0
      to: 4294967295
  runAsGroup:
    rule: 'MustRunAs'
    ranges:
    - from: 0
      to: 4294967295
  fsGroup:
    rule: 'MustRunAs'
    ranges:
    - from: 0
      to: 4294967295
  volumes:
  - '*'

$ kubectl get pod -l app=nginx -o json | jq '.items[].spec.securityContext'
{
  "fsGroup": 1000
}

$ kubectl port-forward service/nginx 8080:80
Forwarding from 127.0.0.1:8080 -> 9796
Forwarding from [::]:8080 -> 9796
Handling connection for 8080

$ kubectl label --overwrite ns default pod-security.kubernetes.io/enforce-baseline
namespace/default labeled
$ kubectl apply -f privileged-psp.yaml
Warning: policy/v1beta1 PodSecurityPolicy is deprecated in v1.21+, unavailable in v1.25+
podsecuritypolicy.policy/privileged created
$ kubectl create clusterrole privileged-psp --verb use --resource podsecuritypolicies.policy --resource-name privileged
clusterrole.rbac.authorization.k8s.io/privileged-psp created
$ kubectl create -n default rolebinding disable-psp --clusterrole privileged-psp --group system:serviceaccounts:default
rolebinding.rbac.authorization.k8s.io/disable-psp created
$ kubectl rollout restart deployment/nginx-nonpriv
deployment.apps/nginx-nonpriv restarted
$ kubectl port-forward service/nginx 8080:80
Forwarding from 127.0.0.1:8080 -> 9796
Forwarding from [::]:8080 -> 9796
Handling connection for 8080
```

Demo!

pspmigrator

DOI: 10.1002/2297

Limitations of Pod Security Admission

1. Controlled by namespace labels
2. No profile customization: privileged, baseline or restricted
3. No mutations!
4. Applies namespace-wide*

Pod Security Alternatives

1. [Open Policy Agent \(OPA\)](#) / [GateKeeper](#)

- [gatekeeper-library/pod-security-policy](#)

2. [Kyverno](#)

... and other 3rd party controllers

3. *(coming soon)* CEL Admission

- [Webhook Fatigue? You're Not Alone: Introducing the CEL Expression Language Features Solving This Problem - Joe Betz, Google](#)
- [KEP-3488: CEL for Admission Control](#)

4. Roll your own *Easier than you might think!*

- [Kubebuilder - Admission Webhooks](#)

Why not both?

- Defense in depth!
- Pod Security Admission is low-overhead
- Minimize custom policies
- Pod Security Standards maintained by K8s OSS
 - New Feature? No problem, the Pod Security Standard will get updated
 - Remember ephemeral containers?

Recommendation:

Apply the minimum viable Pod Security Standard using PSA, and layer on additional controls with a custom approach

Conclusion

- Start early & take it slow - unblock your v1.25 upgrade!
- Use an incremental approach: one namespace at a time
- Considerations: mutations, scaled-to-zero & occasional workloads

Thank you!

Resources:

PSP Migration Guide - docs.k8s.io/tasks/configure-pod-container/migrate-from-psi/

Pod Security Standards - docs.k8s.io/concepts/security/pod-security-standards/

PSP Spec Mapping - docs.k8s.io/reference/access-authn-authz/psi-to-pod-security-standards/

pspmigrator tool - sigsg.k8s.io/pspmigrator



Please give us feedback!

Alternatives:

Open Policy Agent (OPA) - openpolicyagent.org/docs/latest/kubernetes-introduction/

GateKeeper - open-policy-agent.github.io/gatekeeper/website/docs/

- PSP library -

github.com/open-policy-agent/gatekeeper-library/blob/master/library/pod-security-policy/

Kyverno - kyverno.io