Securing your Kubernetes Cluster - Pod Security Standards

Overview

We want to configure strict enforcement of rules to limit unauthorized manipulation of our kubernetes cluster.

Pod Security Standards vs Pod Security Policies

PSS Policy	PSS Control	Mapping to PSP Control
Baseline	Host Namespaces	Usage of host namespaces Usage of host networking and ports
	Privileged Containers	Running of privileged containers
	Capabilities	Linux capabilities
	HostPath Volumes	Usage of the host filesystem
	Host Ports	*Not covered in PSP*
	AppArmor (optional)	The AppArmor profile used by containers
	SELinux (optional)	The SELinux context of the container
	/proc Mount Type	The Allowed Proc Mount types for the container
	Sysctls	The sysctl profile used by containers
Restricted	Volume Types	Usage of volume types Allow specific FlexVolume drivers Requiring the use of a read-only root file system
	Privilege Escalation	Restricting escalation to root privileges
	Running as Non-root	*Not covered in PSP*
	Non-root groups	Allocating an FSGroup that owns the Pod's volumes The user and group IDs of the container
	Seccomp	The seccomp profile used by containers

The main drawbacks of the PSP are the lack of support for other resource types and a limited list of controls that don't cover some container runtime-specific parameters. PSP is planned to be deprecated in 2021, and a better alternative exists to address the same need. The actual deprecation date has been recently extended from February 1st, 2021 to May 3rd, 2021 to allow vendors that use PSP, such as Azure, to prepare for the change.

PSP is planned to be officially deprecated in Kubernetes version 1.21 and removed in version 1.25.

According to the Kubernetes deprecation policy, older versions will stop getting support nine months after the deprecation of the feature.

Example Pod Security Policy

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: example
spec:
  privileged: false  # Don't allow privileged pods!
  # The rest fills in some required fields.
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  runAsUser:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
  volumes:
  - '*'
```

Testing Out Violations Base on Pod Security Standard

Privileged

```
kubectl label --dry-run=server --overwrite ns --all pod-security.kubernetes.io/enforce=privileged
```

```
namespace/default labeled
namespace/kube-node-lease labeled
namespace/kube-public labeled
namespace/kube-system labeled
```

Baseline

```
kubectl label --dry-run=server --overwrite ns --all pod-security.kubernetes.io/enforce=baseline
```

```
Warning: existing pods in namespace "default" violate the new PodSecurity enforce level "baseline:latest"
Warning: fluent-bit-q9d8n: hostPath volumes
namespace/default labeled
namespace/kube-node-lease labeled
namespace/kube-public labeled
Warning: existing pods in namespace "kube-system" violate the new PodSecurity enforce level "baseline:latest"
Warning: etcd-docker-desktop (and 3 other pods): host namespaces, hostPath volumes
Warning: kube-proxy-zggs2: host namespaces, hostPath volumes, privileged
Warning: storage-provisioner (and 1 other pod): hostPath volumes
namespace/kube-system labeled
```

Restricted

```
kubectl label --dry-run=server --overwrite ns --all pod-security.kubernetes.io/enforce=restricted
```

```
Warning: existing pods in namespace "default" violate the new PodSecurity enforce level "restricted:latest"
Warning: ckaf-kafka-0 (and 5 other pods): unrestricted capabilities, seccompProfile
Warning: credential-management-5665fb95d4-ncssg (and 1 other pod): allowPrivilegeEscalation != false,
unrestricted capabilities, runAsNonRoot != true, seccompProfile
Warning: fluent-bit-q9d8n: allowPrivilegeEscalation != false, unrestricted capabilities, restricted volume
types, runAsNonRoot != true, seccompProfile
Warning: kowl-767d84f95f-qm5pj: allowPrivilegeEscalation != false, unrestricted capabilities, seccompProfile
namespace/default labeled
namespace/kube-node-lease labeled
namespace/kube-public labeled
Warning: existing pods in namespace "kube-system" violate the new PodSecurity enforce level "restricted:latest"
Warning: coredns-95db45d46-skl6z (and 1 other pod): unrestricted capabilities, runAsNonRoot != true,
seccompProfile
Warning: etcd-docker-desktop (and 3 other pods): host namespaces, allowPrivilegeEscalation != false,
unrestricted capabilities, restricted volume types, runAsNonRoot != true
Warning: kube-proxy-zggs2: host namespaces, privileged, allowPrivilegeEscalation != false, unrestricted
capabilities, restricted volume types, runAsNonRoot != true, seccompProfile
Warning: storage-provisioner (and 1 other pod): allowPrivilegeEscalation != false, unrestricted capabilities,
restricted volume types, runAsNonRoot != true, seccompProfile
namespace/kube-system labeled
```

From the previous output, you'll notice that applying the privileged Pod Security Standard shows no warnings for any namespaces. However, baseline and restricted standards both have warnings, specifically in the kube-systemnamespace.

Adding Pod Security Standard to your Namespace

Multiple pod security standards can be enabled on any namespace, using labels. Following command will enforce the baseline Pod Security Standard, but warn and audit for restricted Pod Security Standards as per the latest version (default value)

```
kubectl label --overwrite ns <NAMESPACE> \
  pod-security.kubernetes.io/enforce=baseline \
  pod-security.kubernetes.io/enforce-version=latest \
  pod-security.kubernetes.io/warn=restricted \
  pod-security.kubernetes.io/warn-version=latest \
  pod-security.kubernetes.io/audit=restricted \
  pod-security.kubernetes.io/audit-version=latest
```

Example: Enable Restricted on default namespace:

```
kubectl label --overwrite ns default \
  pod-security.kubernetes.io/enforce=restricted \
  pod-security.kubernetes.io/enforce-version=latest \
  pod-security.kubernetes.io/warn=restricted \
  pod-security.kubernetes.io/warn-version=latest \
  pod-security.kubernetes.io/audit=restricted \
  pod-security.kubernetes.io/audit-version=latest
```

```
namespace/default labeled
```

Exemptions

You can define exemptions from pod security enforcement in order to allow the creation of pods that would have otherwise been prohibited due to the policy associated with a given namespace.

....

References

Reference	URL
Apply Pod Security Standards at the Cluster Level	https://kubernetes.io/docs/tutorials/security/cluster-level-pss/
Apply Pod Security Standards at the Namespace Level	https://kubernetes.io/docs/tutorials/security/ns-level-pss/
Pod Security Admission	https://kubernetes.io/docs/concepts/security/pod-security-admission/
Migrate from PodSecurityPolicy to the Built-In PodSecurity Admission Controller	https://kubernetes.io/docs/tasks/configure-pod-container/migrate-from-psp/
Kubernetes Pod Security Policy Deprecation: All You Need to Know	https://blog.aquasec.com/kubernetess-policy