Kubernetes PSP

Pod Security Policies

Defense: Pod Security Policies

A pod security policy (PSP) sets standards for pods' admission to the cluster.

You define standards, then state which users can use which pod security policy as the "minimum security bar" to clear.

Pod Security Policy Coverage

Pod Security Policies allow you to restrict the privilege with which a pod runs.

- Volume white-listing / Usage of the node's filesystem
- Read-only root filesystem
- Run as a specific (non-root) user
- Prevent privileged containers (all capabilities, all devices, …)
- Root capability maximum set
- SELinux or AppArmor profiles – choose from a set
- Seccomp maximum set
- Sysctl maximum set

Pod Security Policy Methodology

- Without pod security policies, the system allows any pod to be admitted.
- Once you apply a single pod security policy, it becomes default-deny.
- Multiple pod security policies can be in place – each one defines a set of standards that will grant a pod admission to the cluster.
- The pod security policies are evaluated in alphabetical order, such that people generally create numbered policies, this this example:

```
10-no-root-appamor-required
20-root-allowed-apparmor-required
30-root-allowed-no-apparmor-required
```

Pod Security Policy RBAC

To use a pod security policy to obtain admission to the cluster for your pods, your user or service account needs a role permitting you to use that PSP.

The policy is a cluster-wide resource, but RBAC permits different users and namespaces to have different policies.

This is easier to understand through exercise – we'll be doing one shortly where you can see how this part works.

RBAC Reminder

A role is a set of capabilities, given a name to group them.

Example: You could name a role "`create-pods-deployments`"

The capabilities are verb - object pairs, like:

```
create deployments
create pod
```

A role binding is what connects a user/service account to a role.

```
service account "frontend" is bound to "create-pods-deployments"
```

PSPs Require the Admission Controller

You can define Pod Security Policies, but they will only be effective if the PodSecurityPolicy admission controller is activated.

This is a security vulnerability, as a cluster operator can have a false sense of security if she applies PSPs, but they are silently unenforced.

On our test cluster, we do this by changing the manifest file that describes the kube-apiserver pod:

1. Edit the file /etc/kubernetes/manifests/kube-apiserver.yaml
2. On the line --admission-control, append “PodSecurityPolicy”.
3. Restart the kube-apiserver program’s container and you’re ready to go.

Exercise: Kubernetes Pod Security Policies

Let's do another defense exercise on the same scenario we just worked with.

Please:

Open the Firefox browser on the class machine to:
http://localhost:10000/exercises/kubernetes-psp