Pods

» Multi-Container Pods (Sidecar)

Multi-container Pods

Create a new pod manifest that specifies two containers.

```
cat > pod-multi-container.yaml <<EOF
apiVersion: v1
kind: Pod
metadata:
  name: my-two-container-pod
  labels:
    environment: dev
spec:
  containers:
    - name: server
      image: nginx:1.13-alpine
      ports:
        - containerPort: 80
          protocol: TCP
    - name: side-car
      image: alpine:latest
      command: ["/bin/sleep","600"]
  restartPolicy: Never
EOF
```

Create the pod by specifying the manifest.

```
kubectl create -f pod-multi-container.yaml
```

View the detail for the pod and look at the events.

```
kubectl describe pod my-two-container-pod
```

Let's first execute a shell session inside the server container by using the `-c` flag.

```
kubectl exec -it my-two-container-pod -c server -- /bin/sh
```

Run some commands inside the server container.

```
ip address
netstat -ntlp
hostname
ps
exit
```

Let's now execute a shell session inside the side-car container.

```
kubectl exec -it my-two-container-pod -c side-car -- /bin/sh
```

Run the same commands in side-car container. Each container within a pod runs it's own cgroup, but shares IPC, Network, and UTC (hostname) namespaces.

```
ip address
netstat -ntlp
hostname
```

Try to send a TERM signal to PID 1 (/bin/sleep)

```
ps
kill -s TERM 1
ps
exit
```

Clean Up

Delete the pod by specifying the manifest.

```
kubectl delete -f pod-multi-container.yaml
```

If we wanted to force a delete of a pod we can use `--grace-period=0 --force .`

View any remaining resources.

```
kubectl get all
```