One powerful feature of Kubernetes deployments is the ability to perform rolling updates and rollbacks. These allow you to push out new versions without incurring downtime, and they allow you to quickly return to a previous state in order to recover from problems that may arise when deploying changes. In this lesson, we will discuss rolling updates and rollback, and we will demonstrate the process of performing them on a deployment in the cluster.

Relevant Documentation

- https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#updating-a-deployment
- https://kubernetes.io/docs/concepts/workloads/controllers/deployment/#rolling-back-a-deployment

Lesson Reference

Here is a sample deployment you can use to practice rolling updates and rollbacks.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: rolling-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.1
        ports:
        - containerPort: 80
```

Perform a rolling update.

```
kubectl set image deployment/rolling-deployment nginx=nginx:1.7.9 --record
```

Explore the rollout history of the deployment.

```
kubectl rollout history deployment/rolling-deployment

kubectl rollout history deployment/rolling-deployment --revision=2
```

You can roll back to the previous revision like so.

```
kubectl rollout undo deployment/rolling-deployment
```

You can also roll back to a specific earlier revision by providing the revision number.

```
kubectl rollout undo deployment/rolling-deployment --to-revision=1
```

You can also control how rolling updates are performed by setting `maxSurge` and `maxUnavailable` in the deployment spec:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: rolling-deployment
spec:
  strategy:
    rollingUpdate:
      maxSurge: 3
      maxUnavailable: 2
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.1
        ports:
        - containerPort: 80
```