Deploy Replica Set and Replication Controller, and deployment. Also learn the advantages and disadvantages of each

1. ReplicationController

The original Kubernetes controller for ensuring a specified number of pod replicas are running at any time.

rc.yaml

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: rc-demo
spec:
  replicas: 2
  selector:
    app: rc-demo
  template:
    metadata:
      labels:
        app: rc-demo
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

Apply:

```
$ kubectl apply -f rc.yaml
```

2. ReplicaSet

A newer, more flexible controller that replaces ReplicationController. Supports set-based selectors.

rs.yaml

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: rs-demo
spec:
  replicas: 2
  selector:
    matchLabels:
      app: rs-demo
  template:
    metadata:
      labels:
        app: rs-demo
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

Apply:

```
$ kubectl apply -f rs.yaml
```

3. Deployment

A higher-level controller that manages ReplicaSets and provides declarative updates, rollbacks, and rollouts.

deploy.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deploy-demo
spec:
  replicas: 2
  selector:
    matchLabels:
```

```
      app: deploy-demo
  template:
    metadata:
      labels:
        app: deploy-demo
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
```

Apply:

$ kubectl apply -f deploy.yaml

4. Advantages and Disadvantages

Replication Controller

-Advantages:
Simple and straightforward configuration
Ensures desired number of pod replicas
Automatic pod replacement on failure
Basic load distribution across nodes
Suitable for simple, stateless applications

-Disadvantages:
Limited selector capabilities (equality-based only)
No rolling update support
No deployment history or rollback features
Manual update process is disruptive
Deprecated in favor of ReplicaSets and Deployments
Limited scaling and management features

ReplicaSet

-Advantages:
Advanced selector capabilities (set-based selectors)
Better label matching with matchExpressions

More flexible pod selection criteria
Improved performance over Replication Controllers
Foundation for Deployments
Supports complex label queries

-Disadvantages:
No built-in update strategy
No rollback capabilities
Manual rolling updates are complex
No deployment history tracking
Requires manual management for updates
Not recommended for direct use in production

Deployment

-Advantages:
Declarative updates and rollbacks
Built-in rolling update strategies
Automatic rollback on failed deployments
Deployment history and revision tracking
Pause and resume deployment capabilities
Multiple update strategies (RollingUpdate, Recreate)
Automatic ReplicaSet management
Production-ready with advanced features
Integration with HPA and other controllers
Comprehensive status reporting

-Disadvantages:
More complex configuration
Higher resource overhead
May be overkill for simple use cases
Additional abstraction layer
Requires understanding of underlying ReplicaSetss