CKAD Ders Dökümanı #24 — Persistent Volumes (PV/PVC)

Dil: Türkçe • CKAD Kategorisi: Application Design and Build (20%) • Güncelleme: ReadWriteOncePod accessMode satırı kaldırıldı, Reclaim Policy bölümü genişletildi

1) Konu Anlatımı

Kubernetes’te Pod’lar geçicidir; Pod yeniden yaratıldığında container dosya sistemi sıfırlanabilir. Persistent Storage, Pod’lar yeniden başlasa bile verinin kalıcı olmasını sağlar.

Bu mimaride iki ana kaynak vardır: • PV (PersistentVolume): Cluster seviyesinde “gerçek depolama kaynağını” temsil eder. • PVC (PersistentVolumeClaim): Uygulama/namespace tarafının “depolama isteği”dir.

2) PV ve PVC Arasındaki Farklar

Kapsam	PV	PVC
Scope	Cluster-scoped	Namespace-scoped
Amaç	Gerçek storage kaynağını tanımlar (kapasite, erişim, backend).	Uygulamanın storage talebini tanımlar (boyut, erişim modu, storageClass).
Kim yönetir?	Çoğunlukla cluster admin / platform ekibi.	Uygulama geliştiricisi / ekip.
Binding	PVC ile eşleşince ‘Bound’ olur.	Uygun PV bulunursa PV’ye bağlanır.
Değişiklik etkisi	PV değişimi cluster kaynaklarını etkileyebilir.	PVC değişimi çoğu zaman sınırlıdır (örn. büyütme).

2.1 PVC bağlanma mantığı (binding)

PVC oluşturulduğunda Kubernetes, istenen kapasite/accessMode/storageClass gibi kriterlere göre uygun bir PV bulur ve PVC’yi ona bağlar. Dinamik provisioning açıksa, PV otomatik üretilebilir.

3) Hangi Durumda PV Oluşturmalıyız?

Statik provisioning: Storage’ı önceden siz hazırlarsınız ve PV olarak tanımlarsınız (örn. belirli NFS export, belirli disk).Dinamik provisioning (StorageClass): PVC verildiğinde storage sistemi otomatik PV üretir (örn. CSI provisioner).

4) accessModes Nedir? Ne İşe Yarar?

accessModes, volume'un Pod/Node'lar tarafından nasıl mount edilebileceğini ifade eder. Bu, teorik bir "niyet" beyanıdır; gerçek destek storage backend'ine bağlıdır.

Mod	Açıklama	Ne zaman uygun?
ReadWriteOnce (RWO)	Volume aynı anda tek bir node tarafından RW mount edilir. Pod başka node'a taşınırsa re-attach gerekir.	Çoğu blok disk ve local disk. Tek replica stateful app (DB, queue).
ReadOnlyMany (ROX)	Birden fazla node volume'u read-only mount edebilir.	Static içerik paylaşımı; nadir kullanılır.
ReadWriteMany (RWX)	Birden fazla node aynı anda RW mount edebilir.	Paylaşımlı dosya sistemi (NFS, CephFS, EFS). Çok replica aynı datayı yazacaksa.

4.1 Hızlı karar tablosu

Tek replica DB → genelde RWO.Birden çok replica aynı dosya alanına yazacak → RWX.Sadece okuma, çok kopya → ROX.

5) Statik PV + PVC Örneği

Aşağıdaki örnek statik provisioning mantığını göstermek içindir. PV'de backend tanımı; PVC'de ise talep (request) bulunur.

```
# pv.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-demo
spec:
  capacity:
    storage: 1Gi
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: manual
  hostPath:
    path: /mnt/data/pv-demo
```

```
# pvc.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-demo
spec:
  accessModes:
  - ReadWriteOnce
  storageClassName: manual
  resources:
    requests:
      storage: 1Gi
```

6) localPath / local PV Kullanımı

local volume, storage'un node'un lokal diski olmasıdır. Volume belirli bir node'a bağlıdır; Pod başka node'a giderse aynı veriye erişemez. Bu yüzden genellikle node affinity ile birlikte kullanılır.

6.1 Local PV örneği (nodeAffinity ile)

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: local-pv
spec:
  capacity:
    storage: 2Gi
  volumeMode: Filesystem
  accessModes:
  - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  storageClassName: local-storage
  local:
    path: /mnt/local-data
  nodeAffinity:
    required:
      nodeSelectorTerms:
      - matchExpressions:
        - key: kubernetes.io/hostname
          operator: In
          values:
          - worker-1
```

7) Reclaim Policy (PVC silinince PV/Veri ne olur?)

PV üzerindeki persistentVolumeReclaimPolicy, PVC silindiğinde PV'nin ve alttaki verinin nasıl ele alınacağını belirler.

Policy	Davranış	Ne zaman tercih edilir?
Retain	PVC silinse bile PV 'Released' olur; alttaki veri silinmez. Temizlik/yeniden kullanım çoğunlukla manuel yapılır.	Üretimde kritik veriler. Yanlışlıkla PVC silinse bile veriyi korumak için.
Delete	PVC silinince PV ve alttaki storage kaynağı silinir (dinamik provisioner destekliyorsa).	Geçici ortamlar, dev/test. Yanlış kullanım veri kaybına yol açar.

7.1 Reclaim policy örnekleri

```
# Retain (veriyi koru)
spec:
  persistentVolumeReclaimPolicy: Retain

# Delete (PVC silinince storage da silinsin)
spec:
  persistentVolumeReclaimPolicy: Delete
```

Not: Retain kullandığınızda PVC silindikten sonra PV genellikle 'Released' durumda kalır. Aynı PV'yi tekrar kullanmak için manuel müdahale gerekebilir (platform politikanıza bağlı).

8) En Sık Kullanılan Komutlar

```
kubectl get pv
kubectl get pvc -A

kubectl describe pv pv-demo
kubectl describe pvc pvc-demo

kubectl get pv pv-demo -o jsonpath='{.spec.persistentVolumeReclaimPolicy}{"\n"}'
```

9) Troubleshooting Hızlı Rehber

PVC Pending: Uygun PV yok → storageClassName, accessModes, kapasite uyuşuyor mu?Pod Pending: Local PV ise Pod doğru node’a gidemiyor olabilir → nodeAffinity/selector/taint kontrol.Permission denied: fsGroup/securityContext veya initContainer ile chmod/chown gerekebilir.Veri silindi: reclaimPolicy Delete olabilir → kritik veride Retain düşün.