UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
KATEDRA ZA PRIMENJENE RAČUNARSKE NAUKE

Računarstvo u oblaku

ms Helena Anišić

Zimski semester 2025/2026.

Studijski program: Računarstvo i automatika

Modul: Računarstvo visokih performansi

Skladištenje podataka

Vrste podataka u Docker-u

Aplikacija (kod + okruženje)	Privremeni podaci aplikacije	Permanentni podaci aplikacije
Napisan od strane programera	Dobavljeno / Kreirano u pokrenutom kontejneru	Dobavljeno / Kreirano u pokrenutom kontejneru
Dodato u sliku kontejnera prilikom build faze	Skladišteno u memoriji ili privremenim fajlovima	Skladišteno u fajlovima ili u bazi podataka
Fiksno - ne može da se promeni nakon što se slika build-uje	Dinamični i promenljivi	Ne sme da se izgubi ako se kontejner obriše
Read-only, skladišteno u slici	Read + write, privremeni, skladišteni u kontejneru	Read + write, permanentni, skladišteni u kontejneru (volumes)

Skladištenje permanentnih podataka

- Docker skladišta podataka
  - Anonimna skladišta podataka (anonymous volume)
  - Imenovana skladišta podataka (named volume)
- Skladišne tačke uvezivanja (bind mount)

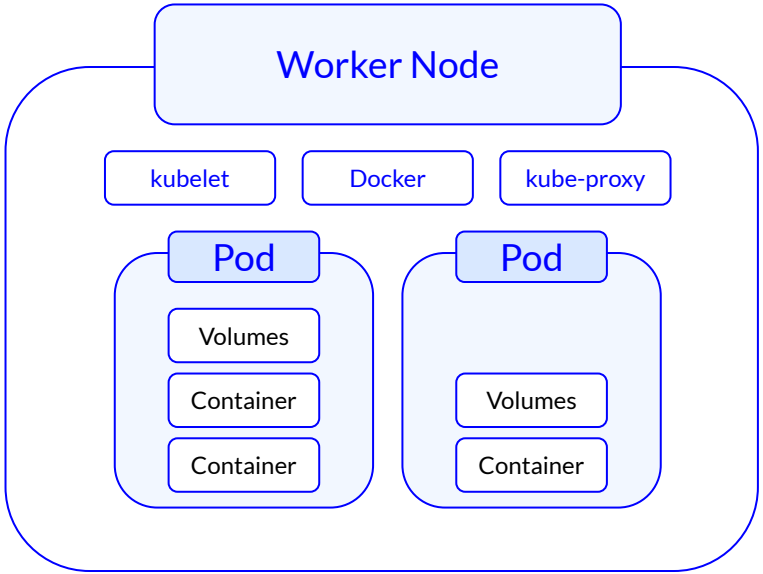
Problem skladištenje pri radu sa Kubernetesom?

- I dalje radimo sa volume-ima jer radimo sa docker kontejnerima, ali postoji jedan problem.
- Sada Kubernetes pokreće kontejnere!
  - Ne koristi se docker CLI za pokretanje kontejnera
    - Programer ne može da prosledi `-v` opciju kako bi dodao *volume* prilikom pokretanja kontejnera
  - Ne koristi se docker-compose klijent za pokretanje kontejnera
    - Programer ne može da definiše rad sa *volume*-ima u okviru *compose.yaml* fajla
- Potrebno je konfigurisati Kubernetes tako da koristi volume sa pokrenutim kontejnerima u Pod-u

Kubernetes & Volume

- Kubernetes omogućava učitavanje volume-a unutar kontejnera.
- Pomoću Kubernetesa se aplikacija može izvršavati na:
  - više različitih čvorova
  - različitim cloud provajderima
- Kubernetes podržava veliki broj tipova skladišta podataka (engl. *Volume types*) i drajvera (engl. *Driver*)
  - da bi podaci mogli da se čuvaju na različitim mestima
  - https://kubernetes.io/docs/concepts/storage/volumes/
- Životni vek volume-a zavisi od životnog veka Pod-a
  - Volume je deo Pod-a
    - Preživeće restartovanje kontejnera
    - Neće preživeti uništavanje Poda i njegovo ponovno kreiranje

Worker Node


Worker Node
kubelet
Docker
kube-proxy
Pod
Volumes
Container
Container
Pod
Volumes
Container

Kubernetes volumes vs Docker volumes

Kubernetes volume	Docker volume
Podržava različite tipove i drajvere.	Ne podržava dodatne tipove ni drajver.
Volume-i nisu suštinski trajni (neće preživeti restartovanje Pod-a).	Volume-i su trajni dok se ručno ne obrišu.
Volume-i preživljavaju restartovanje i brisanje kontejnera.	Volume-i preživljavaju restartovanje i brisanje kontejnera.

Primer 1

- Pokrenuti primer radi bez problema sve dok se kontejner ne restartuje
- Restartovanje aplikacija se izaziva pogađanjem **/error** *endpoint*-a
    - kontejner se sruši i ponovo podigne, a podaci nestanu jer ne postoji nikakav volume za njihovo trajno čuvanje

Kubernetes & Volumes

- Životni vek volume-a zavisi od životnog veka Pod-a
  - Volume se definiše na istom mestu gde i Pod
- Tip volume-a ne utiče na sam kontejner
- Tip volume se odnosi samo na to na koji način će se taj volume odnosno ti podaci skladištiti
  - Kod docker-a to nije bio problem, podaci su nam uvek bili fajlovi u fajlsistemu

Kubernetes volume - *EmptyDir*

- Kreira novi prazan direktorijum kad god se Pod pokrene
  - održava volume u životu i napunjen podacima sve dok Pod živi
- Kontejneri mogu da pišu u ovaj direktorijum, a ako se restartuju podaci će ostati sačuvani
- Ako se kontejner restartuje ili obriše
  - podaci ostaju sačuvani
- Ako se Pod ukloni podaci nestaju.
  - Kada se kreira novi Pod, kreira se i novi EmptyDir
- Odlična opcija ako se radi razvoj i koristi se samo jedna replika
  - *EmptyDir* je Pod specifičan
  - Ne čuvaju se isti podaci u okviru različitih Pod-ova

Primer 1 - *EmptyDir*

```
spec:
     containers:
       - name: data-node
         image: cloudftn/kub-data-demo:1
         volumeMounts:
           - mountPath: /app/story
             name: story-volume
     volumes:
       - name: story-volume
         emptyDir: {}
```

Kubernetes volume - *hostPath*

- Omogućava postavljanje putanje u okviru čvora
  - Podaci sa te putanje će biti učitani u odgovarajuće Pod-ove
- Nije vezano za tačno određeni Pod kao *EmptyDir*
- Učitava i postojeće podataka na prosleđenoj putanji
  - Slično skladišnoj tački uvezivanja (bind mount)
  - EmptyDir ne podržava ovu funkcionalnost
- Dobra opcija ako se radi samo na jednom čvoru
  - *hostPath* je vezan za čvor

Primer 1 - *hostPath*

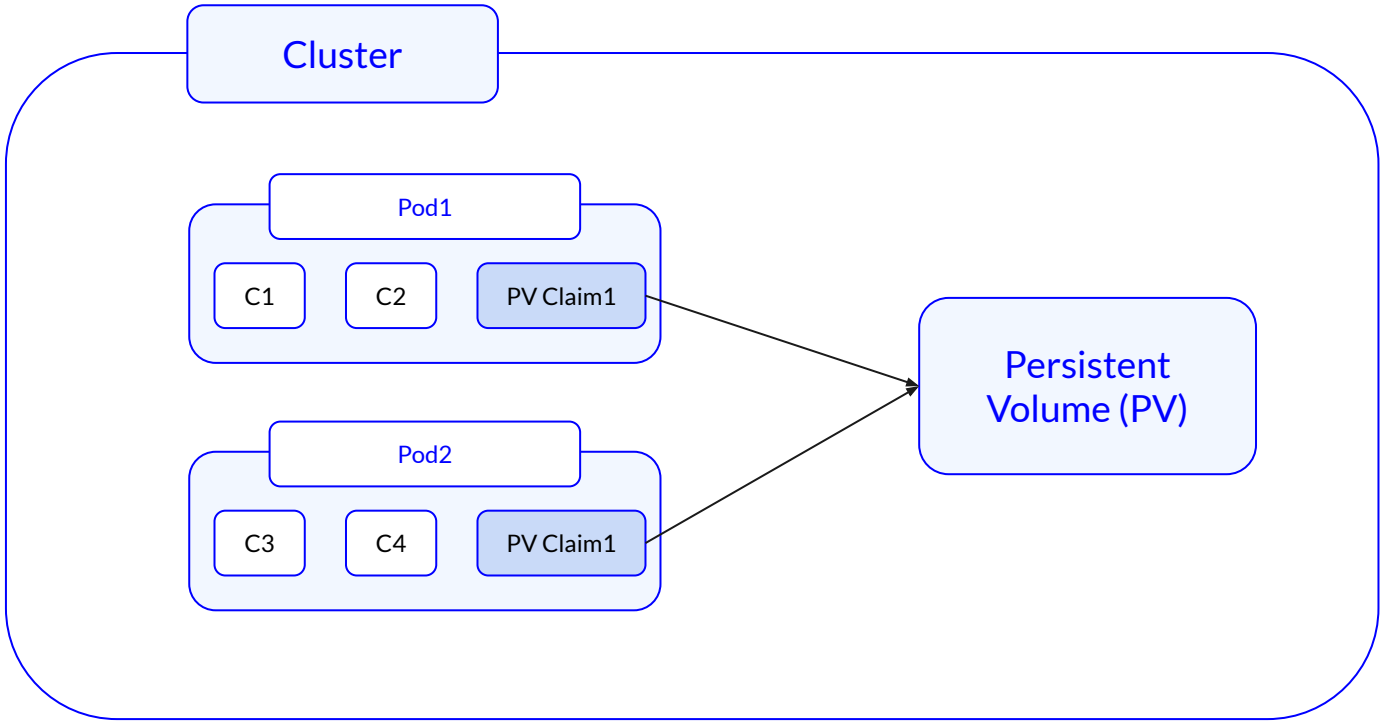
```
spec:
     containers:
       - name: data-node
         image: cloudftn/kub-data-demo:1
         volumeMounts:
           - mountPath: /app/story
             name: story-volume
     volumes:
       - name: story-volume
         hostPath:
           path: /data
           type: DirectoryOrCreate
```

Kubernetes volume - *hostPath*

- **"" (prazan string):** Podrazumevano ponašanje. Navedena putanja može biti fajl ili direktorijum. Kubernetes ne proverava da li putanja postoji niti kog je tipa.
- **DirectoryOrCreate:** Osigurava da je navedena putanja direktorijum. Ako ne postoji, Kubernetes kreira prazan direktorijum.
- **Directory:** Navedena putanja mora već postojati i biti direktorijum. U suprotnom, pod neće biti pokrenut.
- **FileOrCreate:** Osigurava da je navedena putanja fajl. Ako ne postoji, Kubernetes kreira prazan fajl na toj putanji.
- **File:** Navedena putanja mora već postojati i biti fajl. U suprotnom, pod neće biti pokrenut.
- **Socket:** Navedena putanja mora postojati i biti UNIX soket. U suprotnom, pod neće biti pokrenut.
- **CharDevice:** Navedena putanja mora postojati i biti uređaj sa karakterima (character device). U suprotnom, pod neće biti pokrenut.
- **BlockDevice:** Navedena putanja mora postojati i biti uređaj sa blokovima (block device). U suprotnom, pod neće biti pokrenut.

Persistent Volume PV

- *EmptyDir* je dobra opcija prilikom upotrebe jednog Pod-a
  - PROBLEM: ako se Pod uništi podaci nestaju
- *hostPath* je dobra opcija prilikom upotrebe jednog čvora sa više Pod-ova
  - PROBLEM: za svaki Pod treba pojedinačno da se definiše specifikacija volume-a
    - čak iako je specifikacija ista za sve Pod-ove
- Persistent Volume - PV je nezavistan i od kontejnera i od Pod-a i od čvora
  - Ne uništava se kada Pod nestane
  - Ne zavisi od životnog ciklusa Pod-a
    - Jednom se konfiguriše i koristi više puta


Cluster
Pod1
C1
C2
PV Claim1
Pod2
C3
C4
PV Claim1
Persistent Volume (PV)

Primer 1 - *host-pv.yaml*

```
apiVersion: v1
kind: PersistentVolume
metadata:
 name: host-pv
spec:
 capacity:
   storage: 1Gi #maksimalna količina prostora koju PV može da koristi
 volumeMode: Filesystem #podaci se formatiraju kao datotečni sistem (npr. ext4)
 storageClassName: standard
 accessModes:
   - ReadWriteOnce    #isključivo jedan čvor ima pristup čitanju i pisanju podataka
  #- ReadOnlyMany #više čvorova ima pristup čitanju podataka
  #- ReadWriteMany #više čvorova ima pristup pisanju podataka
persistentVolumeReclaimPolicy: Delete #brisanjem PVC-a, briše se i PV na koji povezan sa njim
                                      #druga opcija je Retain, koja obezbeđuje prelazak PV-a u Release stanje i
                                      #čuvanje podataka sve dok se ručno ne obriše PV
hostPath:
   path: /data
   type: DirectoryOrCreate
```

Primer 1 - *host-pvc.yaml*

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: host-pvc
spec:
 volumeName: host-pv #definiše tačno određeni PV na koji će se povezati PVC
                     #polje nije obavezno, a ukoiliko se izostavi, PVC će pokušati da pronađe PV sa
                     #odgovarajućim podešavanjima (resources, accessModes i storageClassName)
 accessModes:
   - ReadWriteOnce
  #- ReadOnlyMany
  #- ReadWriteMany
 storageClassName: standard
 resources:
   requests:
     storage: 1Gi
```

Primer 1 - *deployment.yaml*

```
spec:
  containers:
    - name: data-node
      image: cloudftn/kub-data-demo:1
      volumeMounts:
        - mountPath: /app/story
          name: story-volume
  volumes:
    - name: story-volume
      persistentVolumeClaim:
        claimName: host-pvc
```

Normalni volume vs Perzistentni volume

Normalni volume	Perzistentni volume
Volume je povezan sa Pod-om i njegovim životnim ciklusom.	Volume je poseban resurs klastera (nije povezan sa Pod-om).
Definisan i kreiran zajedno sa Pod-om.	Kreiran posebno, potraživan preko PVC-a.
Repetativno i teško za administraciju na globalnom nivou.	Može da se definiše jednom i koristi više puta.

Static VS. Dynamic Provisioning

- Statičko obezbeđivanje (engl. provisioning) podrazumeva **ručno kreiranje Persistent Volume-a (PV)**, koji se kasnije povezuje sa **Persistent Volume Claim-om (PVC)** na osnovu definisanih kriterijuma.

Prednosti:

- **Potpuna kontrola:** Administratori definišu tačno gde i kako se skladište koristi.
- **Kompatibilnost:** Pogodno za sisteme koji ne podržavaju dinamičko obezbeđivanje.
- **Predefinisani resursi:** Omogućava upotrebu postojećih skladišnih resursa.

Ograničenja:

- **Ručno upravljanje:** Zahteva dodatni rad za kreiranje PV-a.
- **Nije skalabilno:** Teško za korišćenje u dinamičnim okruženjima.

Static VS. Dynamic Provisioning

- Dinamičko obezbeđivanje (engl. provisioning) omogućava **automatsko kreiranje Persistent Volume-a (PV)** kada aplikacija zatraži skladište putem Persistent Volume Claim-a (PVC), koristeći definisani **Storage Class**.

Prednosti:

- **Automatizacija:** Kubernetes sam kreira PV na osnovu zahteva PVC-a.
- **Fleksibilnost:** Jednostavno za prilagođavanje različitim potrebama putem Storage Class-a.
- **Skalabilnost:** Pogodno za dinamička okruženja i cloud infrastrukture.

Ograničenja:

- **Oslanjanje na Storage Class:** Zahteva unapred definisane Storage Class-eve za određene provisionere.
- **Manja kontrola:** Administratori nemaju direktnu kontrolu nad PV konfiguracijom.

Dodatni materijali

- https://github.com/ramitsurana/awesome-kubernetes
- https://cloud.google.com/kubernetes-engine/kubernetes-comic/
- https://kubernetes.io/docs/home/