Pamięć masowa w Kubernetesie

inż. Wojciech Baranowski inż. Michał Łubiński

Gdańsk, 2024

Ephemeral storage

Pamięć efemeryczna jest tymczasowa i związana bezpośrednio z instancją maszyny wirtualnej.

- Tymczasowość
- Wysoka wydajnosć
- Niski koszt utrzymania

Ephemeral storage

Przykładowe zastosowania:

- Tymczasowe przechowywanie danych sesji w aplikacjach internetowych.
- Pliki tymczasowe generowane podczas obliczeń i analizy danych.
- Cache aplikacji, które mogą być łatwo odtworzone.

Persistent storage

Pamięć trwała jest zaprojektowana do długoterminowego przechowywania danych, niezależnie od stanu instancji maszyny wirtualnej.

- Trwałość
- Elastyczność
- Wyższy koszt
- Bezpieczeństwo

Persistent storage

Przykładowe zastosowania:

- Bazy danych.
- Systemy plików przechowujące dane użytkowników (np. zdjęcia, dokumenty).
- Kopie zapasowe i archiwizacja danych.

PersistentVolume

PersistentVolume (PV) to zasób w klastrze Kubernetes, który reprezentuje rzeczywiste wolumeny dyskowe. PV jest niezależny od węzłów klastra.

- Trwałość
- Dostępność
- Różnorodność
- Poziomy dostępu (RWO, ROX, RWX)

PersistentVolume

```
apiVersion: "v1"
kind: "PersistentVolume"
metadata:
  name: "test-pv-1"
  namespace: "zsch"
  labels:
    type: "local"
spec:
  storageClassName: "manual"
  capacity:
    storage: "1Gi"
  accessModes:
    - "ReadWriteOnce"
  hostPath:
    path: "/mnt/data"
```

PersistentVolumeClaim

PersistentVolumeClaim (PVC) jest żądaniem zasobu PV przez aplikację w klastrze Kubernetes.

- Abstrakcja
- Dynamika działania
- Elastyczność

PersistentVolumeClaim

```
apiVersion: "v1"
kind: "PersistentVolumeClaim"
metadata:
  name: "test-pvc"
  namespace: "zsch"
spec:
  storageClassName: "manual"
  accessModes:
    - "ReadWriteOnce"
  resources:
    requests:
      storage: "2Gi"
```

StorageClass

Storage class jest zasobem używanym do automatycznego zarządzania i provisionowania zasobów typu storage w klastrze. Storage class umożliwia zdefiniowanie różnych klas pamięci masowej, które mogą być wykorzystywane w zależności od potrzeb. StorageClass zawiera następujące atrybuty:

- Provisioner
- Parameters
- ReclaimPolicy (Retain, Recycle, Delete)
- BindingMode (Immediate, WaitForFirstConsumer)

StorageClass

```
kind: "StorageClass"
apiVersion: "storage.k8s.io/v1"
metadata:
  name: "longhorn"
provisioner: "driver.longhorn.io"
allowVolumeExpansion: true
reclaimPolicy: "Delete"
volumeBindingMode: "Immediate"
parameters:
  numberOfReplicas: "2"
  staleReplicaTimeout: "3600"
  fromBackup: ""
  fsType: "ext4"
```

Storage blokowy

Storage blokowy odnosi się do przechowywania danych w postaci bloków. Każdy blok jest niezależnie adresowany i może być przechowywany w różnych miejscach. Zastosowania:

- Przechowywanie danych dla baz danych i systemów wysokiej wydajności
- Idealny dla aplikacji o dużej intensywności I/O
- Podstawowy storage w przypadku VM

Storage blokowy

Zalety:

- wysoka wydajność
- elastyczność
- bezpieczeństwo

Wady:

- złożoność zarządzania
- koszty

Storage plikowy

Storage plikowy przechowuje dane w postaci plików i katalogów, które są organizowane w strukturze hierarchicznej. Zastosowania:

- Aplikacje o mniejszej intensywności I/O
- Systemy współdzielenia plików oparte o sieć (NAS)
- Środowiska wymagające zrównoleglonej pracy nad plikami

Storage plikowy

Zalety:

- Łatwość użycia
- Współdzielenie danych
- Koszty

Wady:

- Wydajność
- Skalowalność
- Ograniczenia systemów plików

Mechanizmy provisioningowe

Longhorn

Longhorn

Zalety:

- Pełna integracja z kubernetesem
- Łatwość konfiguracji przez UI oraz CLI
- Automatyczna replikacja danych
- Obsługa snapshotów i backupów
- Ławte skalowanie

Wady:

- Wymaga dodatkowych zasobów na replikowane dane
- Mniej wydajny, niż mechanizmy natywnie działające w przestrzeni jądra

GlusterFS

GlusterFS

Zalety:

- Wysoka skalowalność
- Automatyczna replikacja danych
- Funkcje snapshotów

Wady:

- Skomplikowana konfiguracja i zarządzania (CLI + heketi)
- Wymaga manualnej konfiguracji
- Problemy wydajnościowe w dużych klastrach

Ceph RBD

Ceph RBD

Zalety:

- Wysoka wydajność i skalowalność
- Automatyczna replikacja i zarządzanie uszkodzonymi dyskami
- Obsługa snapshotów i klonowania wolumenów

Wady:

- Złożoność zarządzania
- Wymaga zaawansowanej wiedzy technicznej
- Wymaga dużych nakładów zasobów

CephFS

CephFS

Zalety:

- Wysoka skalowalność i elastyczność
- Automatyczna replikacja i zarządzanie uszkodzonymi dyskami
- Obsługa snapshotów i klonowania wolumenów

Wady:

- Złożoność zarządzania
- Wymaga zaawansowanej wiedzy technicznej
- Wymaga dużych nakładów zasobów

iSCSI

iSCSI

Zalety:

- Wysoka wydajność
- Szeroka kompatybilność i wsparcie

Wady:

- Wymaga konfiguracji sieciowej i zarządzania siecią
- Brak natywnej replikacji
- Problemy w przypadku awarii sieci

Porównanie

- Klaster złożony z 3 węzłów master i 16 węzłów worker
- Baza danych SQLite z podmontowanym PVC
- RWO dla storage'u blokowego i RWX dla storage'u plikowego
- Operacje I/O na kolejno 1, 2, 4 i 8 wątkach
- Phoronix Test Suite

Porównanie

Liczba wątków	Longhorn	GlusterFS	Ceph RBD	CephFS
1	238.461	380.979	173.766	186.274
2	433.586	377.084	365.782	207.837
4	616.192	422.790	477.536	229.304
8	905.618	737.335	603.924	286.458

GlusterFS - przykład z życia

Ile może zająć rebalance 250Gb danych?

```
root@kube-prod-data2:/# gluster volume rebalance vol_6d4b17caaf619d2d2ce3b2c04584efe5 status
                                    Node Rebalanced-files          size       scanned      failures       skipped               status  run time in h:m:s
                               ---------      -----------   -----------   -----------   -----------   -----------         ------------     --------------
                               localhost             7916        1.4MB         35570            24            65          in progress       2:07:57
          kube-prod-data1.cui.pg.gda.pl             7830        1.4MB         35103             6            20          in progress       2:07:57
                             10.241.2.8             8313        1.4MB         37197            23           621          in progress       2:07:57
```

GlusterFS - przykład z życia

302 dni!

```
root@kube-prod-data2:/# gluster volume rebalance vol_6d4b17caaf619d2d2ce3b2c04584efe5 status
                                    Node Rebalanced-files          size       scanned      failures       skipped               status  run time in h:m:s
                               ---------      -----------   -----------   -----------   -----------   -----------         ------------     --------------
                               localhost             7916        1.4MB         35570            24            65          in progress       2:07:57
          kube-prod-data1.cui.pg.gda.pl             7830        1.4MB         35103             6            20          in progress       2:07:57
                              10.241.2.8             8313        1.4MB         37197            23           621          in progress       2:07:57
Estimated time left for rebalance to complete :     7203:14:31
```

GlusterFS - przykład z życia

Rozwiązanie:

```
            for filename in files:
                filesrc = os.path.join(path, filename)
                filedst = os.path.join(destination, filename)

                if is_sticky(filesrc):
                    if not args.check:
                        print(f"S: {filesrc}")
                    else:
                        if not os.path.isfile(filedst):
                            print(f"M: {os.path.join(root, filename)}")
                    continue

                try:
                    if not args.dry and not args.check:
                        shutil.copy2(filesrc, filedst)
                    elif args.dry:
                        print(f"C: {filesrc} -> {filedst}")
                except OSError as e:
                    print(f"E: {filesrc}: {e}")
```

GlusterFS - przykład z życia

A co w przypadku storage'u blokowego?

Pytania?

Źródła

- slajdy wykładowe
- materiały udostępnione przez Centrum Usług Informatycznych PG
- skrypt udostępniony przez Kacpra Donata
- https://kubernetes.io/docs/home/
- https://longhorn.io
- https://www.gluster.org
- https://ceph.io
- https://en.wikipedia.org/wiki/ISCSI
- wiedza własna