Volumes и Persistent Volumes в Kubernetes

Что такое хранилища и как работать с постоянными данными в Pod.

Слайд 2 — Задача Volumes

- Pod сам по себе «не хранит» данные.
- Контейнер перезапускается — данные пропадают.
- Volume позволяет вынести данные за пределы контейнера.

Слайд 3 — Три группы хранилищ

1. **emptyDir** — временное хранилище внутри Pod.
2. **hostPath** — каталог на node.
3. **PV/PVC** — постоянное хранилище.

Слайд 4 — emptyDir

- Создаётся вместе с Pod.
- Хранится на node, где запущен Pod.
- Очищается при удалении Pod.
- Использование: кеш, временные файлы.

Слайд 5 — Как работает emptyDir

- Pod стартует → Kubernetes создаёт пустую директорию.
- Контейнеры Pod используют общий каталог.
- После удаления Pod каталог исчезает.

Слайд 6 — hostPath

- Привязка Volume к конкретному пути на node.
- Примеры: `/var/log/app`, `/data/test`.
- Подходит только для тестов и дебага.

Слайд 7 — Ограничения hostPath

- Pod нельзя свободно перемещать по node.
- Риск дать Pod доступ ко всей файловой системе node.
- Не используется в проде.

Слайд 8 — Persistent Volume (PV)

- Объект кластера.
- Описывает реальное хранилище: диск в облаке, NFS, Ceph.
- Живёт независимо от Pod.

Слайд 9 — Что содержит PV

- Размер.
- AccessModes.
- StorageClass.
- ReclaimPolicy.
- Ссылка на реальный backend-диск.

Слайд 10 — Жизненный цикл PV

Статусы:

- **Available** — свободен.
- **Bound** — привязан к PVC.
- **Released** — PVC удалён, но данные остались.
- **Failed** — ошибка.

Слайд 11 — Persistent Volume Claim (PVC)

- Запрос на хранилище.
- Пользователь описывает, что нужно: размер, режим доступа.
- Kubernetes ищет подходящий PV или создаёт новый через StorageClass.

Слайд 12 — Как работает PVC

1. Создаётся PVC.
2. Kubernetes ищет PV с подходящими параметрами.
3. Если нет подходящего — StorageClass создаёт PV.
4. PVC становится Bound.
5. Pod может монтировать PVC как Volume.

Слайд 13 — Access Modes

- **ReadWriteOnce (RWO)** — монтируется к одной node.
- **ReadOnlyMany (ROX)** — только для чтения, к многим node.
- **ReadWriteMany (RWX)** — читается и пишется с разных node.

Слайд 14 — StorageClass

- Описывает, **как** создавать PV.
- Основные поля:
  - **provisioner** (например, AWS EBS, GCE PD, NFS-provisioner)
  - **reclaimPolicy**
  - **volumeBindingMode**

Слайд 15 — provisioner

- Пример:
  - `kubernetes.io/aws-ebs`
  - `kubernetes.io/gce-pd`
  - `nfs.csi.k8s.io`
- Provisioner создаёт реальный диск.

Слайд 16 — reclaimPolicy

Определяет, что делать с PV после удаления PVC:

- **Retain** — оставить данные.
- **Delete** — удалить диск.
- **Recycle** (устарело) — очистить диск.

Слайд 17 — volumeBindingMode

- **Immediate** — создаёт PV сразу после PVC.
- **WaitForFirstConsumer** — создаёт PV только когда Pod появится на конкретной node.
  Помогает избежать ошибки «диск в одной зоне, Pod в другой».

Слайд 18 — Связь Pod → PVC → PV

- Pod использует PVC.
- PVC связывается с PV.
- PV использует реальное хранилище.

Слайд 19 — Как Pod монтирует PVC

В Pod:

- `volumes:` → `persistentVolumeClaim:`
- В контейнере: `volumeMounts:`
  Под монтирует PVC в каталог внутри контейнера.

Слайд 20 — Почему PV/PVC дают постоянство

- Pod можно удалить.
- Node может измениться.
- Deployment может пересоздать Pod.
  Данные сохраняются, потому что PV живёт отдельно.

Слайд 21 — Сценарии использования PV/PVC

- Базы данных.
- Хранилище загрузок.
- Логи, которые нельзя терять.
- Stateful приложения.

Слайд 22 — Типичные ошибки

- PVC больше, чем PV — Bind не произойдёт.
- StorageClass неверный — PV не создаётся.
- AccessMode не поддерживается backend-диском.

Слайд 23 — Проверка PVC

Команда:

```
kubectl get pvc
kubectl describe pvc <имя>
```

Проверяем Bound / Pending.

Слайд 24 — Проверка PV

```
kubectl get pv
kubectl describe pv <имя>
```

Смотрим статус, StorageClass, реальный диск.

Слайд 25 — Как происходит отказоустойчивость

- Pod падает → новый Pod стартует.
- PVC остаётся.
- Volume монтируется снова.
- Данные сохраняются.

Слайд 26 — Сравнение типов хранилищ

Тип	Данные сохраняются?	Привязка к node	Назначение
emptyDir	Нет	Да	временные данные
hostPath	Да	Жёстко	тесты
PV/PVC	Да	Нет	постоянные данные

Слайд 27 — Итоги урока

- Volumes дают Pod файловую систему.
- emptyDir и hostPath — локальные варианты.
- PV/PVC — стандарт для постоянных данных.
- StorageClass управляет созданием PV.
- PVC → PV → хранилище.