Job, cronjob

Job

Выполнения разовой задачи. Если запуск задачи завершается с ошибкой, Job перезапускает поды до успешного выполнения или до истечения таймаутов. Когда задача выполнена, Job считается завершённым и больше никогда в кластере не запускается.

Параметры в spec:

Параметр	Описание
activeDeadlineSeconds	количество секунд, которое отводится всему Job (не для одного пода) на выполнение.
backoffLimit	количество попыток. Если указать 2, то Job дважды попробует запустить под и остановится.
ttlSecondsAfterFinished	через сколько секунд специальный TimeToLive контроллер должен удалить завершившийся Job вместе с подами и их логами

После успешного завершения задания манифесты (Job и созданные поды) остаются в кластере навсегда. Все поля Job имеют статус Immutable, и поэтому при создании Job из автоматических сценариев сначала удаляют Job, который остался от предыдущего запуска. Генерация уникальных имен для Job приведет к накоплению ненужных манифестов. Обязательно указание ttlseconds...

При создании бесконечного цикла через activeDeadlineSeconds будет отправлен sigterm, затем через 30 секунд sigkill.

Если указать backoffLimit без restartPolicy, то при ошибке Job будет выполняться бесконечно.

Cronjob

Создание Job по расписанию.

Параметры в spec:

Параметр	Описание
schedule	Расписание в виде строчки в cron-формате.
startingDeadlineSeconds	Опциональный. Если по прошествии этого времени job не стартовал, старт отменяется. Желательно вместе с Forbid.

Параметр	Описание
concurrencyPolicy	Одновременное выполнение заданий. Allow позволяет подам следующего задания запускаться. Если задание ежеминутное, за минуту Job не отработал, все равно будет создан ещё один. Одновременно могут выполняться несколько Job’ов. Есть риск перегрузки. Replace заменяет запущенную нагрузку: старый Job убивается, запускается новый. Не самый лучший вариант, этот вариант осознанно. Forbid запрет запуска новых Job’ов, пока не отработает предыдущий. С этой политикой можно быть уверенным, что всегда запускается только один экземпляр задачи. Используют наиболее часто
successfulJobsHistoryLimit	Глубина истории хранения удачных job, по умолчанию 3
failedJobsHistoryLimit	Глубина истории хранения неудачных job, по умолчанию 1

CronJob использовать аккуратно. Должны быть независимы и иметь возможность работать параллельно. В качестве альтернативы CronJob можно использовать под, в котором запущен самый обычный crond.

Основные команды

Команда	Доп. пар.	Описание
kubectl get job		список job
	--all-namespaces	
kubectl delete job jname		Удалить jname
kubectl get cronjobs.batch		Список cronjob

Примеры

Job

```
apiVersion: batch/v1
kind: Job
metadata:
  name: hello
spec:
  backoffLimit: 2
```

```
  activeDeadlineSeconds: 60
  ttlSecondsAfterFinished: 100
  template:
    spec:
      containers:
      - name: hello
        image: busybox
        args:
        - /bin/sh
        - -c
        - date; echo Hello from the Kubernetes cluster
      restartPolicy: Never
```

Cronjob

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  schedule: "*/1 * * * *"
  concurrencyPolicy: Allow
  jobTemplate:
    spec:
      backoffLimit: 2
      activeDeadlineSeconds: 100
      template:
        spec:
          containers:
          - name: hello
            image: busybox
            args:
            - /bin/sh
            - -c
            - date; echo Hello from the Kubernetes cluster
          restartPolicy: Never
```

Revision #2
Created 1 April 2025 18:12:28 by Admin
Updated 1 April 2025 19:15:45 by Admin