Московский Государственный Технический Университет им. Н.Э. Баумана

СИСТЕМЫ ОБРАБОТКИ БОЛЬШИХ ДАННЫХ

Разработка приложений

к.т.н.
Папулин Сергей Юрьевич
papulin_bmstu@mail.ru

Москва, 2020

Лекция. Spark on Kubernetes

Kubernetes

Контейнеры позволяют запускать приложения, упакованные вместе с зависимостями, в изолированной среде. В процессе работы необходимо отслеживать, чтобы запущенные контейнеры не вышли из строя. И если такое произойдет, то предпринять какие-либо действия, например, выполнить повторный запуск.

Kubernetes – платформа для отказоустойчивой работы контейнеризированных приложений и систем, которая берет на себя задачи по масштабированию и восстановлению в случае отказа, предоставляет шаблоны по развертыванию приложений и др.

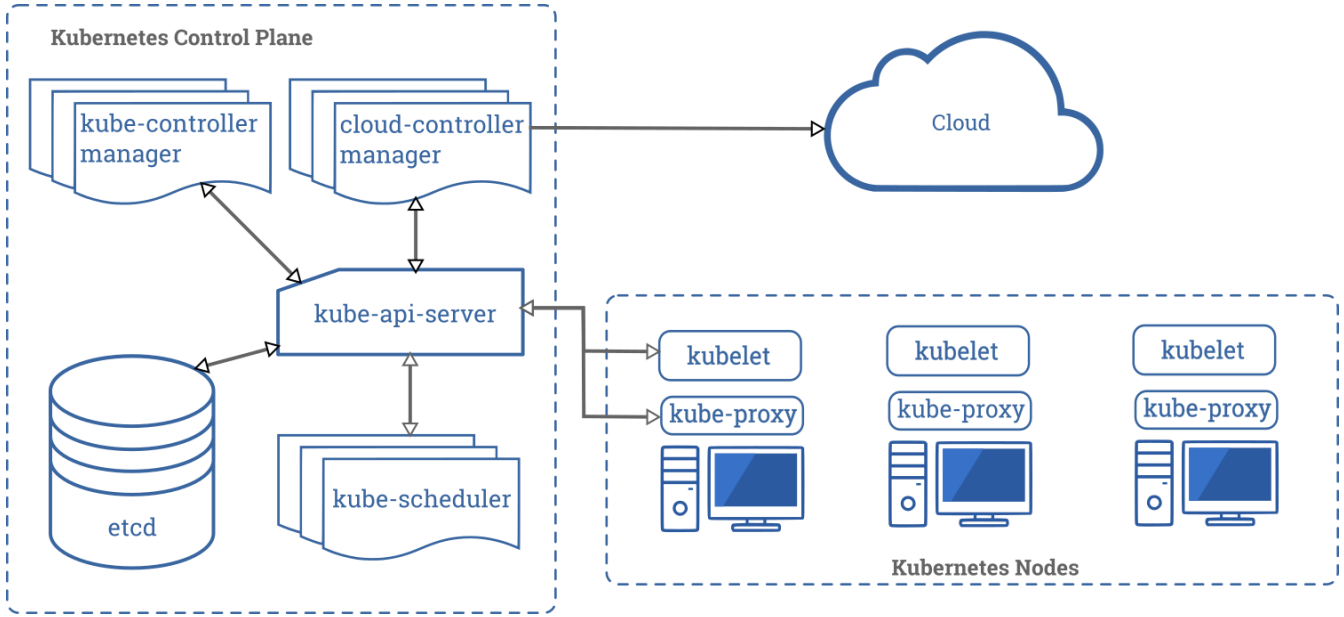
Kubernetes обеспечивает

- Сервис разрешения имен и балансировки нагрузки
- Управление хранилищами данных
- Автоматическое развертывание (rollout) и откат (rollback)
- Размещение контейнеров на основе спецификаций для лучшего использования имеющихся ресурсов
- Автоматическое восстановление контейнеров в случае отказа
- Управление конфиденциальным и конфигурационными данными

Архитектура Kubernetes

- K8s кластер состоит из набора рабочих узлов, на которых запускаются контейнеризированные приложения

- Каждый кластер имеет по крайней мере один рабочий узел



- На рабочих узлах располагаются поды (pods), являющиеся компонентами приложения

- Управляющий уровень (Control Plane) отвечает за управление рабочими узлами и подами в кластере

Компоненты Управляющего уровня

Предназначены для управления кластером:

- Планирование
- Обработка событий

Компоненты могут быть запущены на любом узле. Однако обычно все компоненты запускают на одном узле без возможности запуска пользовательских контейнеров на этой машине

Компоненты:

- kube-apiserver
- etcd
- kube-scheduler
- kube-controller-manager
- cloud-controller-manager

Компоненты Управляющего уровня

kube-apiserver

API сервер проверяет и конфигурирует данные для API объектов (поды, сервисы, контроллеры репликации и др.); обрабатывает REST операции и предоставляет общий интерфейс для взаимодействия всех компонентов кластера

etcd

Распределенное, отказоустойчивое, высокодоступное хранилище ключ-значение, которое используется как основное для всех данных кластера

kube-scheduler

Планирует размещение новых подов на узлах кластера. В качестве критериев могут быть использованы: требования к ресурсам, аппаратные и программные ограничения, расположение данных и пр.

Компоненты Управляющего уровня

kube-controller-manager

Компонент, который запускает контроллеры. Контроллер – это управляющий цикл, который отслеживает состояние кластера через API сервер и при необходимости изменяет текущее состояние на желаемое.

Примеры контроллеров:

- Контроллер узла (Node Controller) отвечает за обнаружение и реагирование на выход из строя узлов
- Контроллер репликации (Replication Controller) отвечает за поддержание корректного числа подов
- Контроллер конечных точек (Endpoints Controller) заполняет объекты сетевых конечных точек, связывая сервисы и поды
- Контроллеры учетных записей и токенов (Service Account & Token controllers) создают стандартные аккаунты и токены доступа для новых пространств имен (namespaces).

Компоненты Управляющего уровня

cloud-controller-manager

Компонент, в который встроена логика управления облачного провайдера

Компоненты:

- Контроллер узла (Node Controller) отвечает за создание Node объектов при запуске нового сервера облачной инфраструктуры. Контроллер узла получает информацию о хостах облачного провайдера
- Контроллер маршрута (Route Controller) отвечает за конфигурирование маршрутов в облаке таким образом, чтобы контейнеры разных узлов могли общаться друг с другом
- Контроллер сервисов (Service Controller) предназначен для интеграции с такими облачными компонентами, как балансировка нагрузки, IP адресации, фильтрации пакетов, проверка работоспособности.

Компоненты Рабочего узла

Компоненты работают на каждом узле, поддерживают запущенные поды и обеспечивают среду выполнения

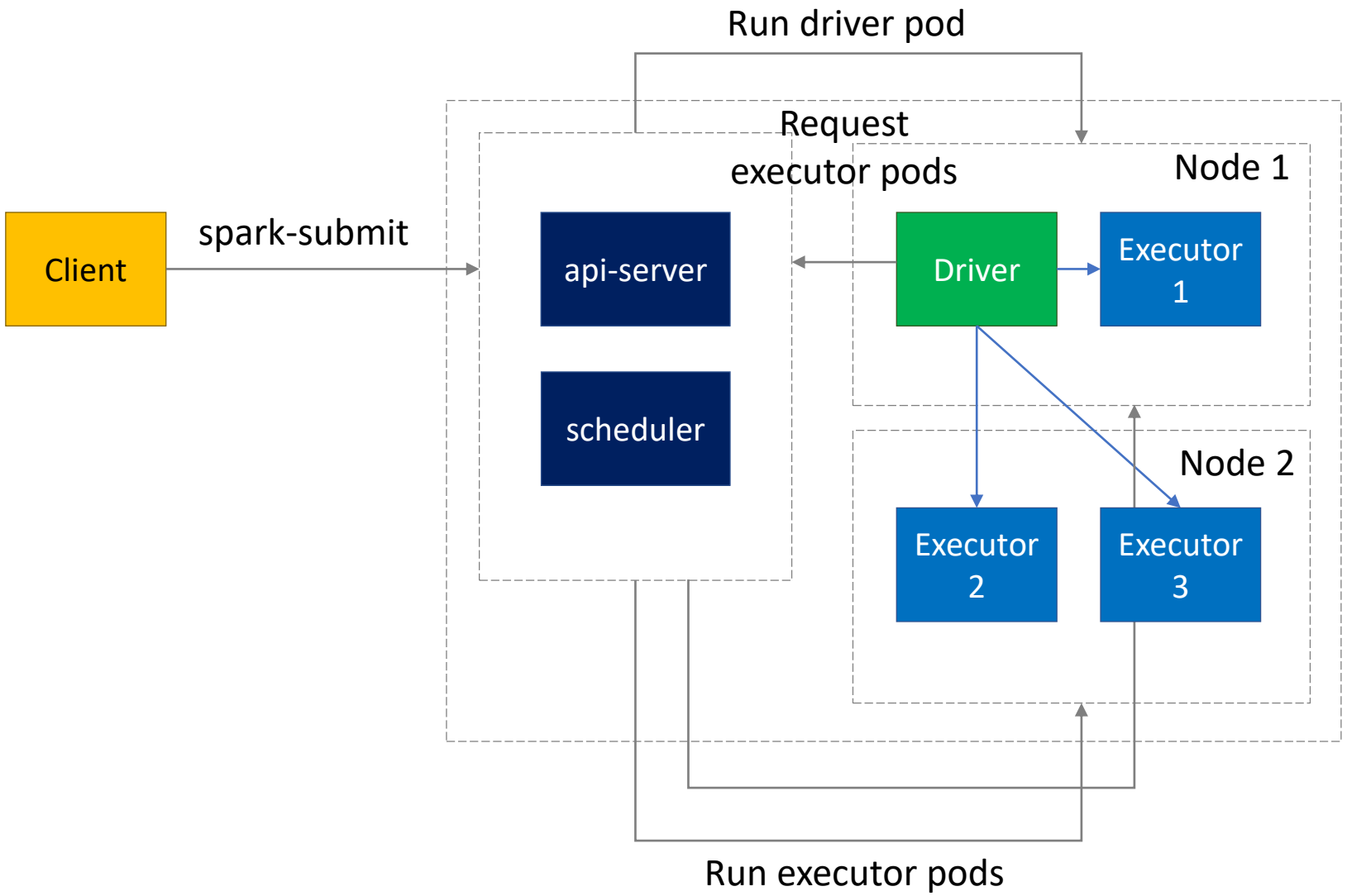
Компоненты:

kubelet
Агент на каждом узле кластера. Следит за тем, чтобы контейнеры были запущены в поде и соответствовали PodSpecs

kube-proxy
Сетевой прокси, который работает на каждом узле кластера. Поддерживает сетевые правила узла, которые обеспечивают сетевую коммуникацию с подами из сетей внутри и снаружи кластера.

container runtime
Среда выполнения контейнера отвечает за работающие контейнеры. K8s поддерживает: Docker, conainerd, CRI-O и некоторые реализации Kubernetes CRI (Container Runtime Interface)

Spark on Kubernetes



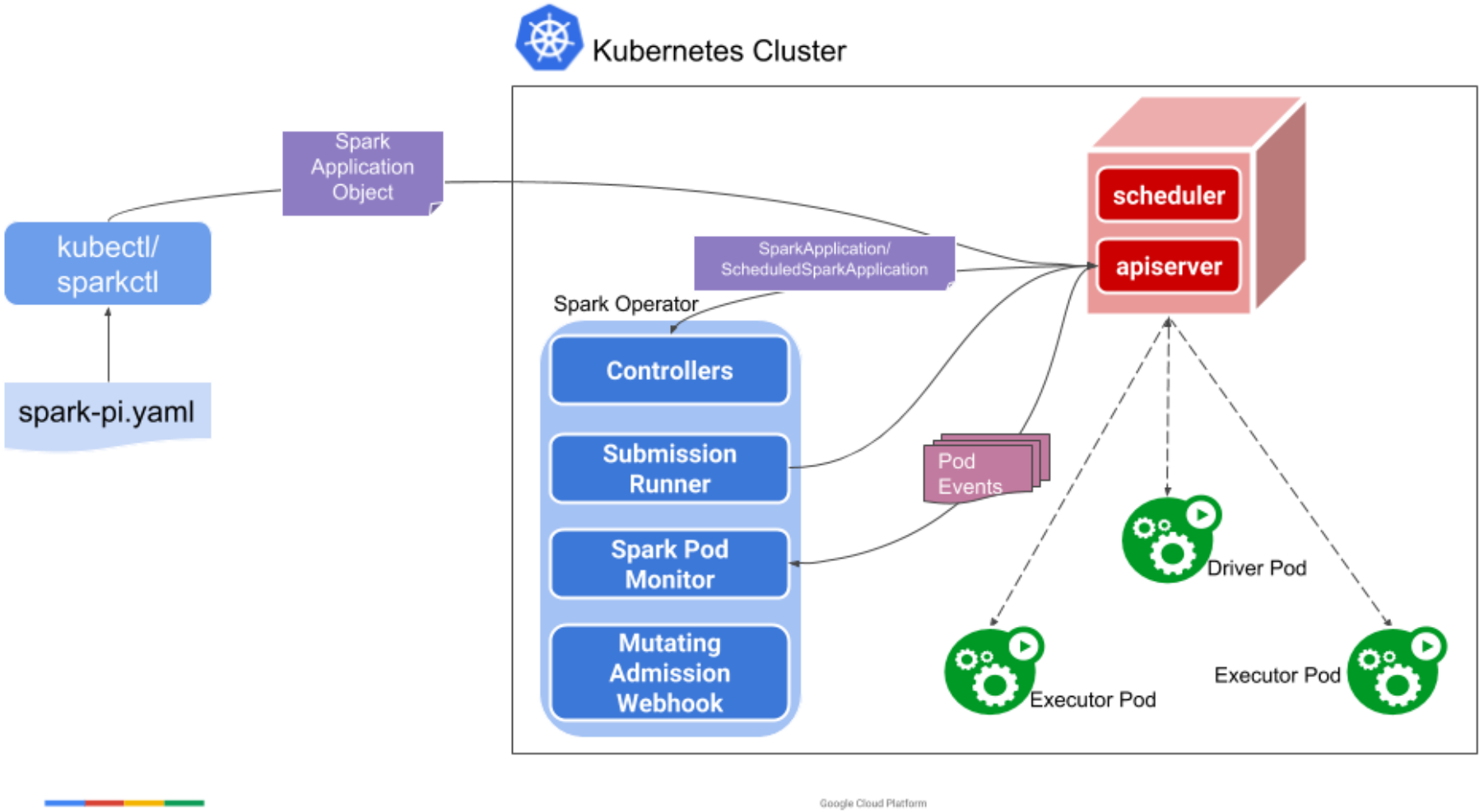
Spark on Kubernetes: spark-submit

Для запуска Spark приложения на K8s можно использовать команду *spark-submit*

Процесс запуска и выполнения следующий:

- spark-submit обращается к K8s (через клиента api-server) и создает Spark driver внутри K8s пода
- Driver запрашивает executor’ы посредством планировщика KubernetesClusterScheduleBackend. Планировщик обращается к K8s (через клиента api-server) для запуска executor’ов внутри подов. Запущенные executor’ы соединяются с driver’ом. После этого выполняется код приложения
- Когда приложение завершило работу, поды executor’ов останавливаются и очищаются. Под (pod) driver’а сохраняет логи и остается в состоянии «completed» до тех пор, пока не сработает сборщик мусора (GC) или не будет удален вручную. В состоянии «completed» под driver’а не использует вычислительные ресурсы и оперативную память.

Spark on Kubernetes: spark-on-k8s-operator



https://github.com/GoogleCloudPlatform/spark-on-k8s-operator/blob/master/docs/design.md

Источники

What is Kubernetes? (doc)

Kubernetes Components (doc)

Pods (doc)

Running Spark on Kubernetes (doc)

Kubernetes Operator for Apache Spark Design (doc)