SkillFactory

ПРОФЕССИЯ

DevOps-инженер

КОНСПЕКТ

МОДУЛЬ D2. ОСНОВЫ KUBERNETES. ЧАСТЬ 1

D2.1 Введение в Kubernetes

Kubernetes — это **единая платформа для организации работы большого количества контейнеров** на различных типах инстансов: железных, виртуальных, облачных у различных провайдеров.

Ключевые свойства подобных платформ — это **портируемость** и **расширяемость**.

Kubernetes как платформы — предоставить **независимую базовую платформу как услугу (*Platform as a Service*, *PaaS*)**.

В классическом подходе ***PaaS*** делается упор на предоставление функциональных особенностей платформы, в то время как ***Kubernetes* делает упор на независимость, универсальность и абстрактный подход** функциональных особенностей. ***K8S*** предоставляет схожий набор функций как и в ***PaaS***, но ***K8S*** является портируемой, расширяемой, модульной и гибкой платформой, оставляя выбор компонента за вами.

Kubernetes похож на каркас дома, материалы для отделки и дальнейшей постройки вы выбираете сами.

Давайте выделим ключевые особенности ***OpenShift***:

1. Разработчики в ***Red Hat*** делают особый упор на безопасность продуктов и контроль качества. ***Red Hat*** максимально быстро выпускают критические обновления, при этом покрывая даже большую часть версий, чем ванильный ***K8S***.
2. ***Red Hat*** является одним из основных контрибьюторов в проект ***Kubernetes***. Многие функции в ***Kubernetes*** появились, благодаря активности ***Red Hat***.
3. ***OpenShift*** предоставляет «из коробки» решения таких вопросов, как работа с аутентификацией, сетевые вопросы, безопасность, мониторинг, журналирование и многое другое. Естественно, если у вас имеется альтернативный взгляд на некоторые компоненты, вы можете их заменить по своему усмотрению, как это и предусматривается в архитектуре ***K8S***.

OpenShift позиционирует себя как **умная *Kubernetes*-платформа, готовая для продакшена.**

Цель ***Rancher*** — это создание портируемых, переносимых контейнерных инфраструктур вокруг различных провайдеров.
Ниже приведены ключевые особенности этого проекта:

D2.2. Архитектура Kubernetes и основные компоненты

Компоненты можно разделить на управляющие кластером компоненты и компоненты, которые работают на узлах, где выполняется полезная нагрузка.

Kube-apiserver — это ключевой компонент системы, который нам предлагает стандартный интерфейс взаимодействия между ***K8S***-компонентами.

etcd — это **распределенное высоконадежное хранилище типа «ключ-значение»** с открытым исходным кодом.

Kubernetes использует ***etcd*** в качестве базы данных для хранения информации, которая критична для стабильности работы кластера.

Kube-scheduler — планирует размещение подов на узлах кластера.

Работу планировщика можно описать таким образом:

1. ***kube-scheduler*** составляет список нод, которые могут быть запланированы для размещения пода на основании критериев (***predicates***).
2. После этого каждый узел кластера оценивается по набору правил установленных планировщике для приоритизации (***priorites***).
3. В результате составленных оценок выбирается узел, набравший максимальное количество очков. В случае если присутствуют узлы, набравшие одинаковый максимальный балл, выбирается случайная.

kube-controller-manager — запускает процессы других контроллеров, каждый из которых выполняет свои узкоспециализированные задачи.

Компоненты *K8S*-ноды:

1. ***Kubelet*** — это агент, который **отвечает за жизненный цикл (запуск, остановка, управление контейнерами) подов на каждой ноде**.
   Важно отметить, что ***kubelet*** следит за теми контейнерами, которые были созданы только в среде ***Kuberntes***. Например, если создается контейнер с помощью докер-команды, то ***kubelet*** ничего не будет знать об этом.
2. ***Kube-proxy*** — этот компонент ответственен **за маршрутизацию траффика пода**.
   При помощи этого компонента возможны сетевые взаимодействия с подами как изнутри, так и снаружи кластера.
   Kube-proxy — это сетевой прокси, он конфигурирует сетевые правила взаимодействия на нодах кластера.
3. **Среда выполнения контейнера** (***Container Runtime Interface (CRI)***) — набор средств для того, чтобы контейнер мог работать. До недавнего времени по умолчанию использовался ***Docker*** как среда выполнения по умолчанию.

Workers nodes (рабочие ноды) — это **узлы *Kubernetes* кластера, где размещаются ваши контейнизированные приложения**. На этих узлах **отсутствуют управляющие компоненты (*Control Plane*) *K8S***-кластера.

D2.3. Установка K8S

Minikube — это удобная консольная утилита, которая позволяет быстро поднять локальный ***Kubernetes***-кластер.