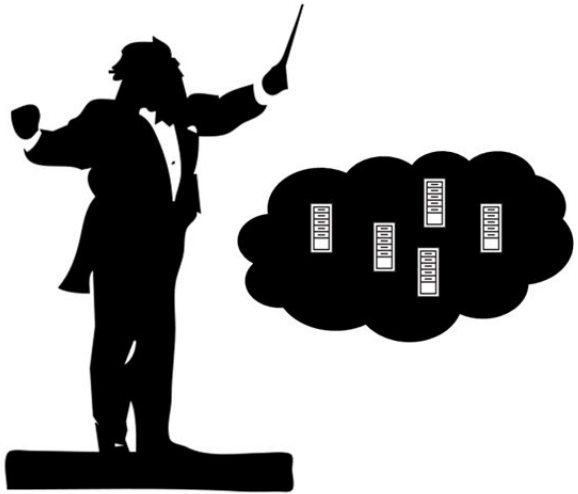
Kubernetes as orchestration

Оркестрация

Контейнеры - это хороший способ объединения и запуска ваших приложений.

В производственной среде вам необходимо управлять контейнерами, в которых выполняются приложения, и следить за тем, чтобы не было простоев. Например, если контейнер выходит из строя, необходимо запустить другой контейнер.

Не было бы проще, если бы такое поведение обрабатывалось системой (**оркестратором**)?



Что конкретно делают оркестраторы

1. Поиск образов и запуск контейнеров
2. Обеспечение контейнеров ресурсами
3. Конфигурация сети
4. Управление жизненным циклом контейнеров
5. Балансировка нагрузки и маршрутизация трафика
6. Мониторинг состояния контейнеров, обнаружение сбоев и восстановление

Kubernetes as orchestration

Самым популярным решением с открытым исходным кодом считается **Kubernetes (k8s)**.

Разработан компанией Google, а в настоящее время поддерживается Cloud Native Computing Foundation (CNCF).

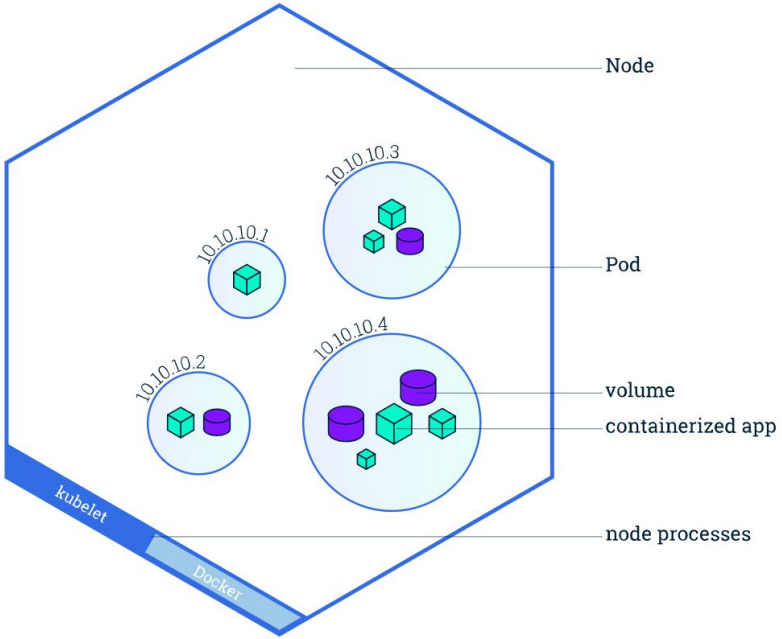
Эта система хорошо известна своей способностью автоматизировать развертывание, управление и, самое главное, масштабирование контейнеров.

kubernetes

Возможности Kubernetes

- **Service discovery**: может предоставлять доступ к контейнеру, используя DNS-имя или используя свой собственный IP-адрес
- **Load balancing**: может балансировать нагрузку и распределять сетевой трафик
- **Storage orchestration**: позволяет вам автоматически подключать систему хранения (локальные хранилища, общедоступные облачные провайдеры и т.д.)
- **Automated rollouts and rollbacks**: можете описать желаемое состояние и система поменяет фактическое состояние на желаемое с контролируемой скоростью
- **Automatic bin packing**: может разместить контейнеры на ваших узлах так, чтобы наилучшим образом использовать ваши ресурсы
- **Self-healing**: перезапускает контейнеры, которые выходят из строя, уничтожает контейнеры, которые не отвечают на проверку работоспособности
- **Secret and configuration management**: позволяет хранить конфиденциальную

Как работает Kubernetes?

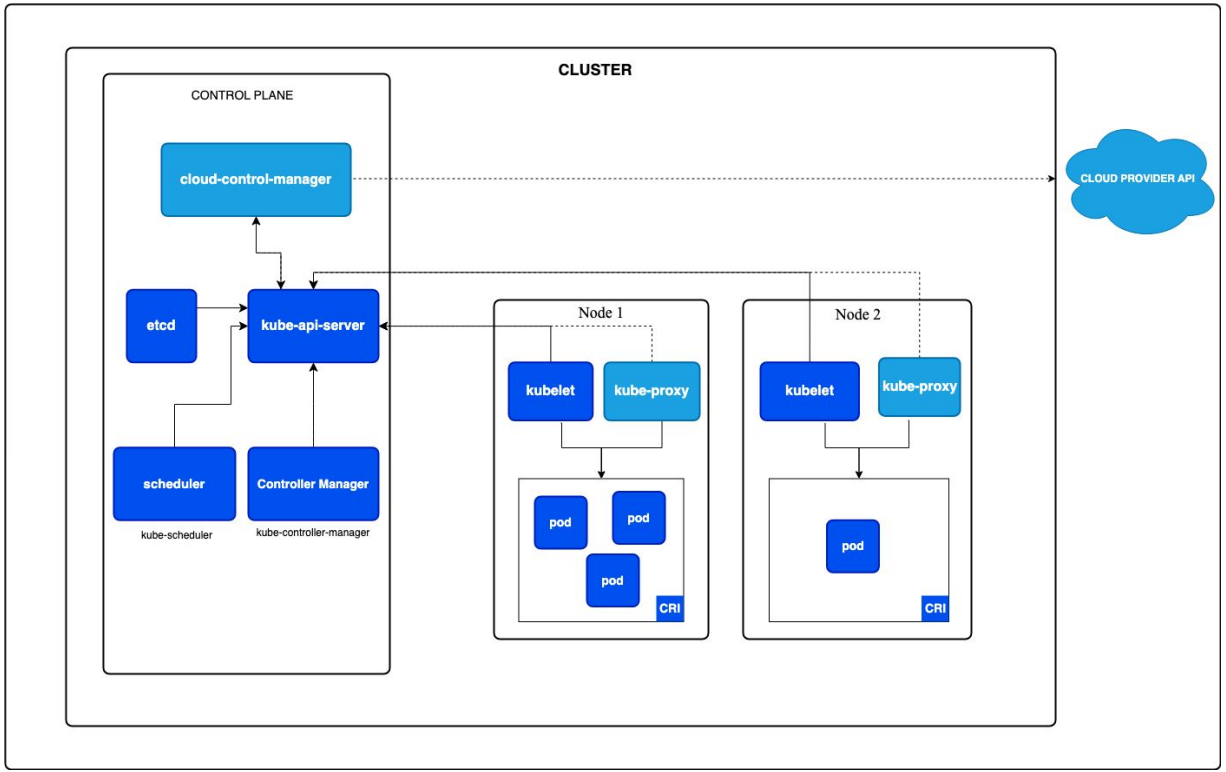


Когда вы разворачиваете Kubernetes, вы получаете **кластер**.

Кластер Kubernetes состоит из набора **worker nodes**, на которых выполняются контейнеризованные приложения. Может быть виртуальной машиной или физической машиной.

На worker nodes размещаются **pods**. Это наименьшие единица развертывания. Состоит из одного или нескольких контейнеров с общим хранилищем и сетевыми ресурсами, а также спецификацией для запуска контейнеров.

Службы кластера Kubernetes



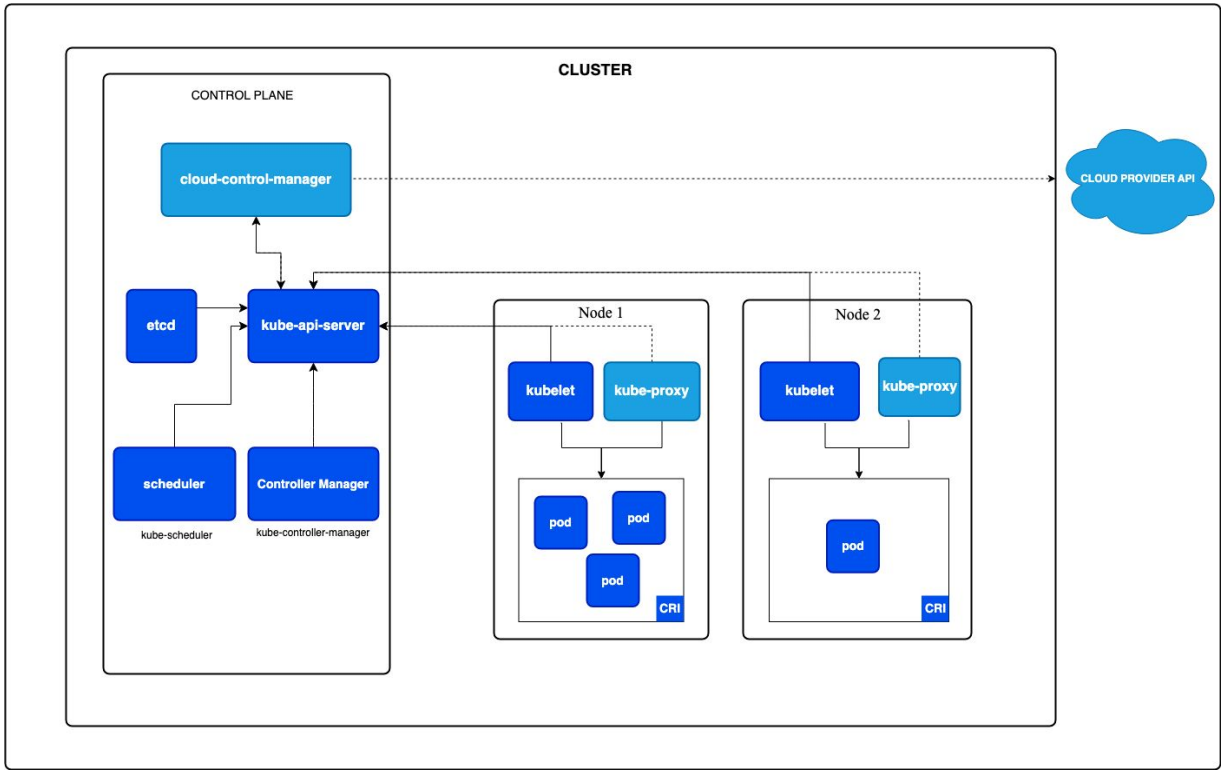
Control plane это уровень оркестровки контейнеров, который предоставляет API и интерфейсы для определения, развертывания и управления жизненным циклом.

kube-apiserver: интерфейс для control plane

kubelet: это агент, который отвечает за взаимодействие с control plane

kube-proxy: это сетевой прокси, работающий на каждом узле

Службы кластера Kubernetes



kube-controller-manager: управление объектами K8S

kube-scheduler: отвечает за планирование pods на доступных рабочих узлах

cloud-controller-manager: позволяет связать кластер с облачным провайдером

etcd: это распределенное хранилище ключей и значений, где хранятся все данные, связанные с кластером.

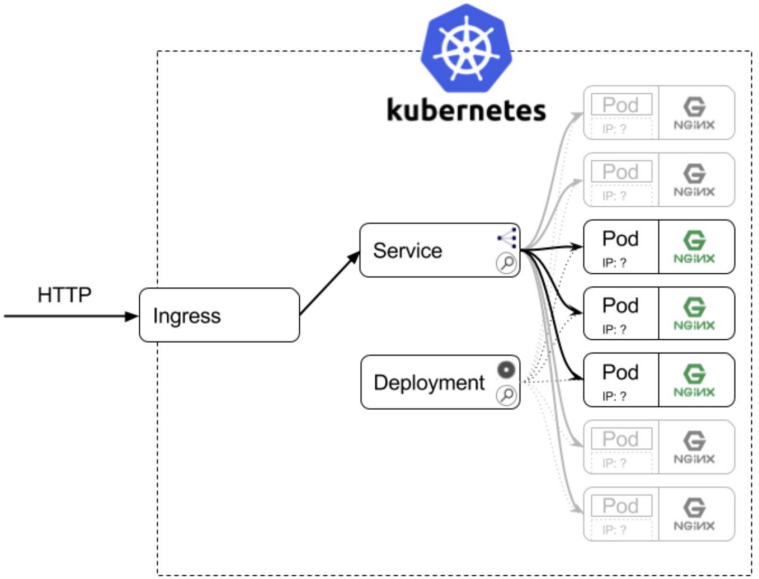
Основные объекты Kubernetes

- **Pod** - это самый маленький юнит работы в Kubernetes, обёртка вокруг одного или нескольких контейнеров
- **Service** - это абстракция, которая определяет стабильную конечную точку для доступа к группе pod-ов
- **Controllers** - отвечают за наблюдение за объектом (например, pod-ом) и следит за тем, чтобы текущее состояние этого объекта соответствовало указанному нами желаемому состоянию. Например, есть Cron Job контроллер, который умеет запускать их по расписанию. Или ещё есть Replica Set, который может их масштабировать. Но самый универсальный контроллер — это **Deployment**.
- **Ingress** предоставляет возможность доступа за пределами кластера
- **Volumes** - позволяют контейнерам в pod-ах получать доступ к данным и обмениваться ими через файловую систему
- **Namespace** - это способ разделения ресурсов кластера между несколькими пользователями

Демонстрация Kubernetes

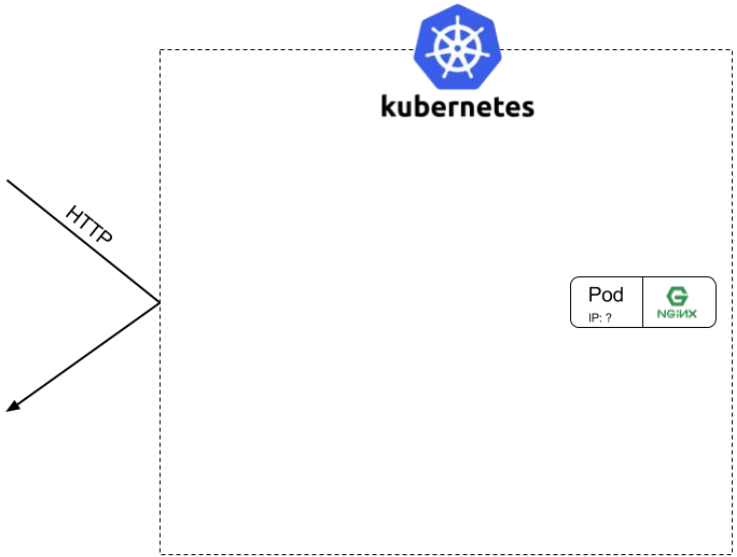
Minikube это инструмент, который позволяет разработчикам использовать и запускать кластер Kubernetes локально.

Это отличный способ быстро запустить кластер и начать взаимодействовать с API Kubernetes.



Демонстрация Kubernetes: Простой POD

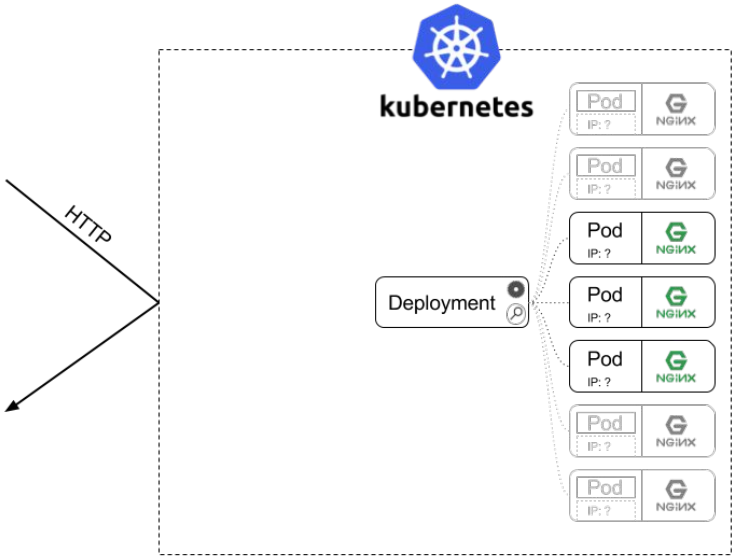
Правда, **pod-одиночка** — уязвим и удивительно бесполезен. Во-первых, снаружи по HTTP к нему не достучаться. Во-вторых, если к контейнеру или хосту придёт костлявая, то, собственно, история nginx на этом и закончится. Если же нам нужно будет его отмасштабировать, то команду apply придётся повторить ещё кучу раз.



Демонстрация Kubernetes: Deployment-контроллер

Контроллеры — это такие Kubernetes объекты, которые могут манипулировать pod’ами.

Самый универсальный контроллер — это **Deployment**. Он и воскресить может, и отмасштабировать, и апдэйт накатить, если попросят.

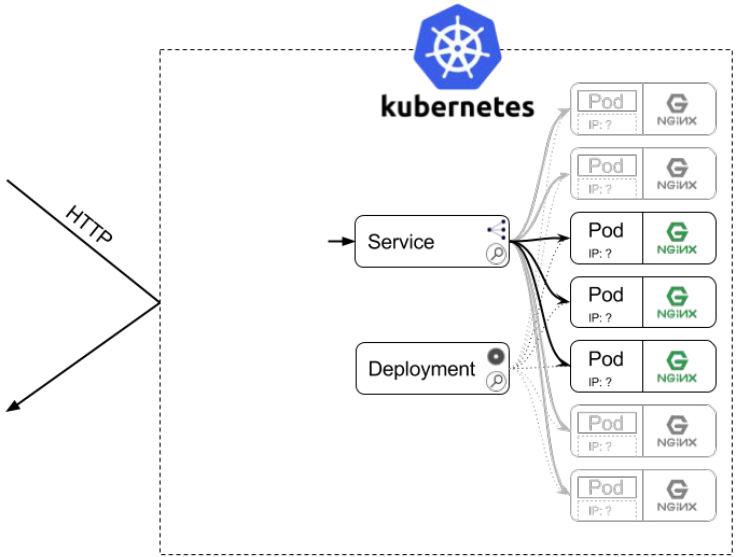


Демонстрация Kubernetes: Сервисы

Точку входа реально тяжело сделать, если pod’ы то умирают, то воскресают, то мигрируют по кластеру.

Сервис-объект, в свою очередь, может эти pod’ы отыскать по меткам и общаться с миром от их лица. Простейший вид - **ClusterIP**.

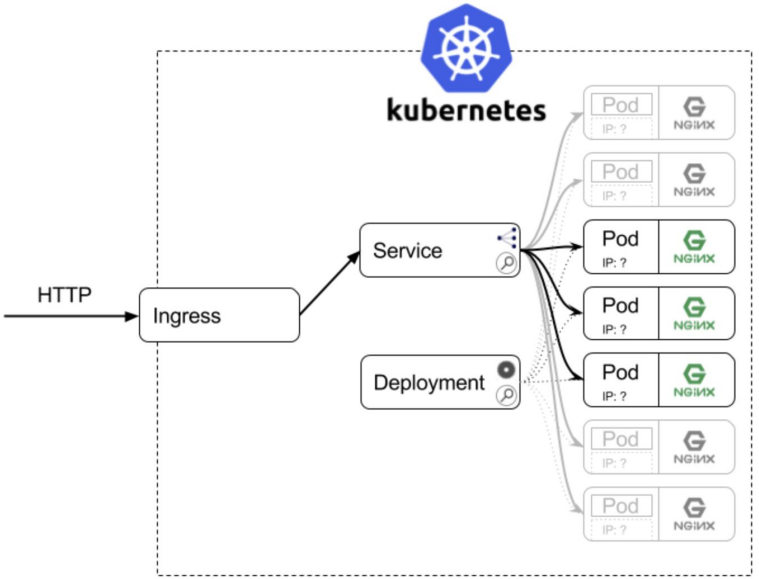
Но основная проблема-то никуда не ушла! К pod’ам всё не достучаться снаружи.



Демонстрация Kubernetes: Ингресс

Но основная проблема-то никуда не ушла! К pod'ам всё не достучаться снаружи.

Ингресс — это могущественный чёрный ящик, который может маршрутизировать запросы из внешнего мира к конкретным сервисам внутри кластера.



Задание: Базовые возможности Kubernetes

Оценка: 2

Описание: Деплой backend-frontend приложения используя Kubernetes.

- Backend: приложение, которое слушает внутренний порт и отправляет ответ, если пришел запрос. Должно быть реплицировано.
- Frontend: приложение, которое слушает внешний порт и отправляет входящие запросы на backend. Должен иметь IP, который может быть использован вне кластера.

DoD: GitLab/GitHub репозиторий с backend Deployment YAML, frontend Deployment YAML, backend Service YAML, frontend Service YAML, README со списком команд для запуска и тестирования + дополнительные файлы для работы системы.