Kubernetes

Содержание

- О чем будет лекция
- Что такое Kubernetes и зачем он нужен
- Архитектура
- Инструменты для работы с Kubernetes
- Создание кластера
- Развертывание приложения
- Элементы Kubernetes
- Масштабирование приложения
- Обновление приложения

Kubernetes

Kubernetes — это портативная расширяемая платформа с открытым исходным кодом для управления контейнеризованными рабочими нагрузками и сервисами, которая облегчает как декларативную настройку, так и автоматизацию. У платформы есть большая, быстро растущая экосистема. Сервисы, поддержка и инструменты Kubernetes широко доступны.

Название Kubernetes происходит от греческого, что означает рулевой или штурман.

Зачем нужен Kubernetes

- **Мониторинг сервисов и распределение нагрузки**
- **Оркестрация хранилища**
- **Автоматическое развертывание и откаты**
- **Автоматическое распределение нагрузки**
- **Самоконтроль**
- **Управление конфиденциальной информацией и конфигурацией**

Чем Kubernetes не является

- **Не ограничивает типы поддерживаемых приложений**
- **Не развертывает исходный код и не собирает приложение**
- **Не предоставляет сервисы для приложения**
- **Не включает решения для ведения журнала**
- **Не указывает и не требует настройки языка/системы (например, Jsonnet)**
- **Не предоставляет и не принимает никаких комплексных систем конфигурации**

Архитектура Kubernetes

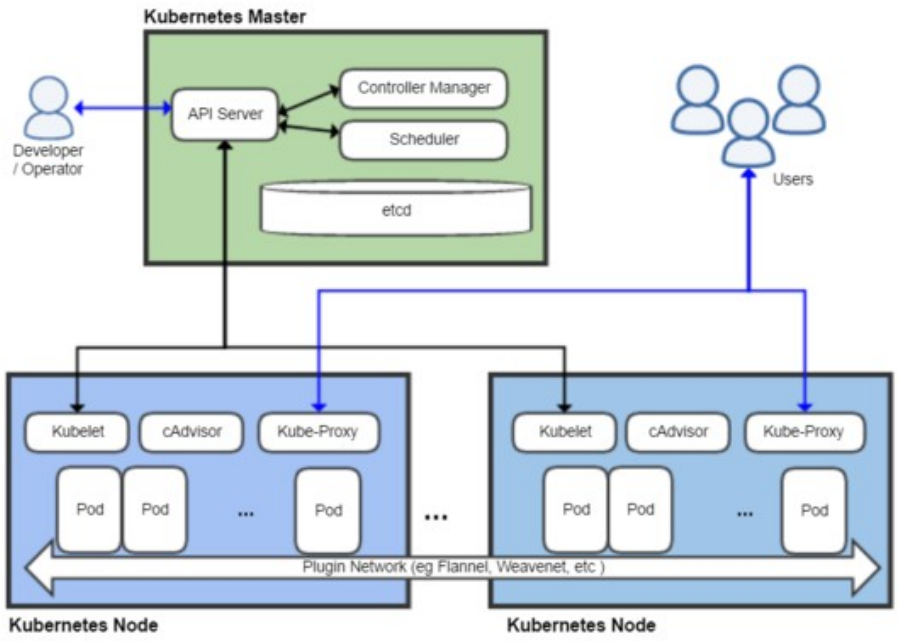
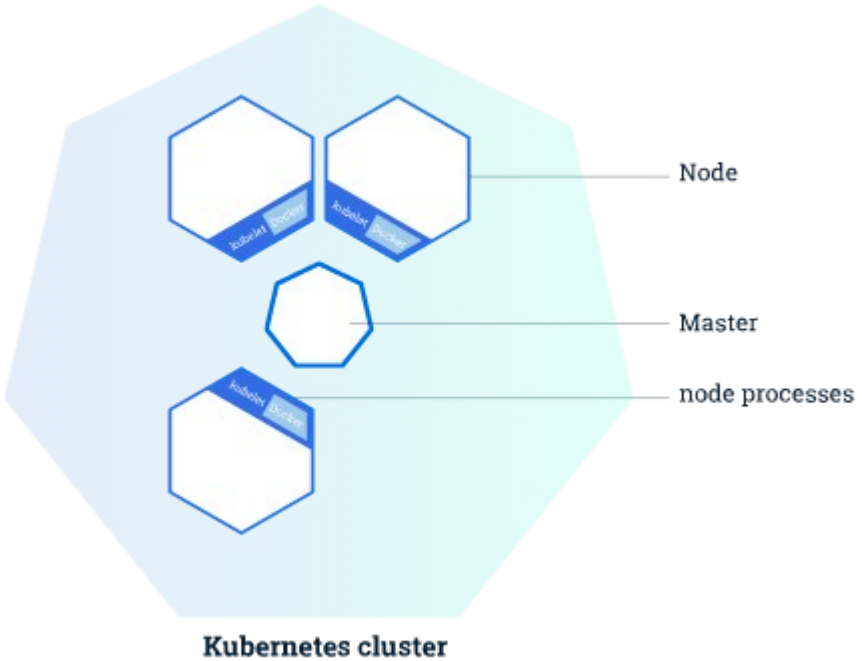


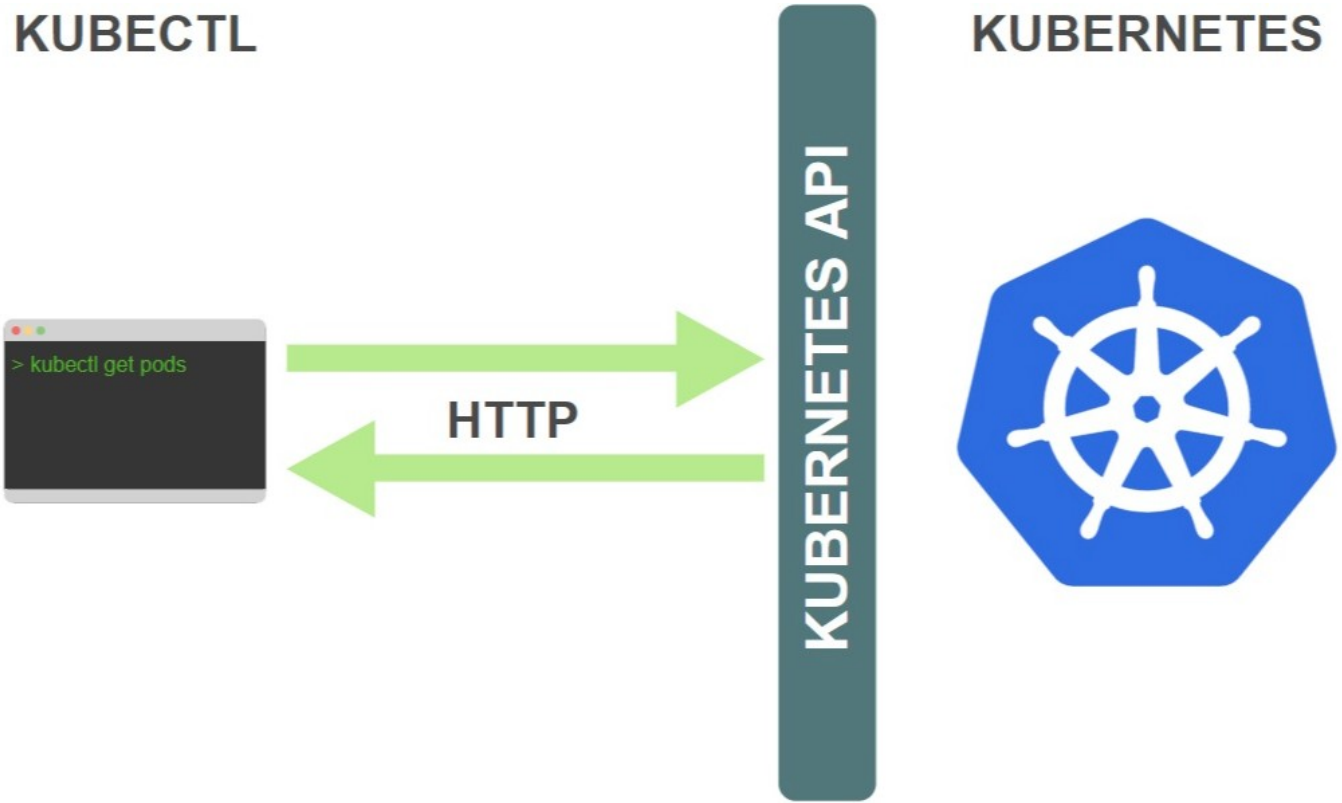
Схема класстера

Кластер Kubernetes состоит из двух типов ресурса:

- **Мастер (ведущий узел) управляет кластером**
- **Рабочие узлы — машины, на которых выполняются приложения**



kubectl



Установка kubectl

Ubuntu, Debian

```
sudo apt-get update && sudo apt-get install -y apt-transport-https gnupg2
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee -a /etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
sudo apt-get install -y kubectl
```

Snap

```
snap install kubectl --classic

kubectl version --client
```

https://kubernetes.io/docs/tasks/tools/install-kubectl/

Minikube

Зависимости Minikube

Убедитесь, что у вас установлен kubectl.

Если у вас ещё не установлен гипервизор, установите один из них:

- **KVM, который также использует QEMU**

- **VirtualBox**

Установка Minikube

Установка Minikube с помощью прямой ссылки

```
curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 \
  && chmod +x minikube
```

Чтобы исполняемый файл Minikube был доступен из любой директории выполните следующие команды:

```
sudo mkdir -p /usr/local/bin/
sudo install minikube /usr/local/bin/
```

https://kubernetes.io/docs/tasks/tools/install-minikube/

Minicube. Основные команды

Чтобы убедиться в том, что гипервизор и Minikube были установлены корректно, выполните следующую команду

```
minikube start --vm-driver=<driver_name>
```

После того, как команда minikube start отработала успешно, выполните команду для проверки состояния кластера:

```
minikube status
```

После того, как команда minikube start отработала успешно, остановите кластер:

```
minikube stop
```

Minicube. Драйвера виртуальной машины

- **docker**
- **virtualbox**
- **podman(EXPERIMENTAL)**
- **vmwarefusion**
- **kvm2**
- **hyperkit**
- **hyperv Note that the IP below is dynamic and can change. It can be retrieved with minikube ip.**
- **vmware (VMware unified driver)**
- **parallels**
- **none (Runs the Kubernetes components on the host and not in a virtual machine. You need to be running Linux and to have Docker installed.)**

https://minikube.sigs.k8s.io/docs/drivers/

Создание кластера

Закрепим команды minicube в связки с kubectl

minikube version

minikube start

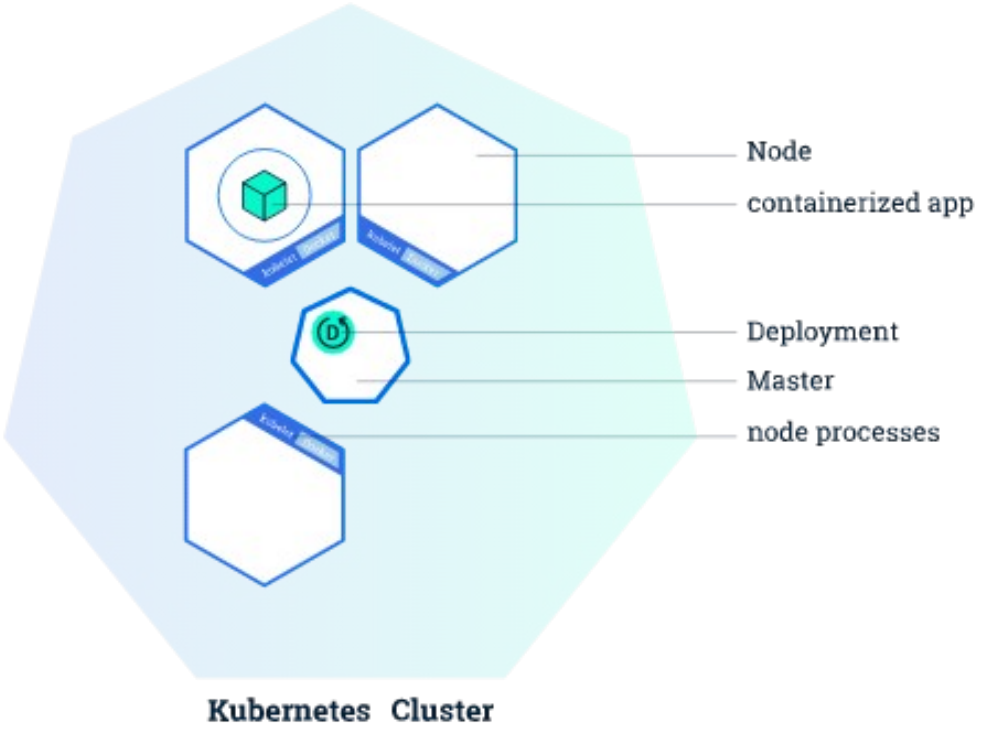
kubectl version

kubectl cluster-info

kubectl get nodes

Развертывание приложения

Как только вы запустили кластер Kubernetes, вы можете развернуть свои контейнеризированные приложения в него. Для этого вам нужно создать конфигурацию развёртывания (Deployment) в Kubernetes. Развёртывание сообщает Kubernetes, как создавать и обновлять экземпляры вашего приложения.



Развертывание приложения. Практика

kubectl get nodes --help

kubectl get nodes

kubectl create deployment kubernetes-bootcamp --image=gcr.io/google-samples/kubernetes-bootcamp:v1

kubectl get deployments

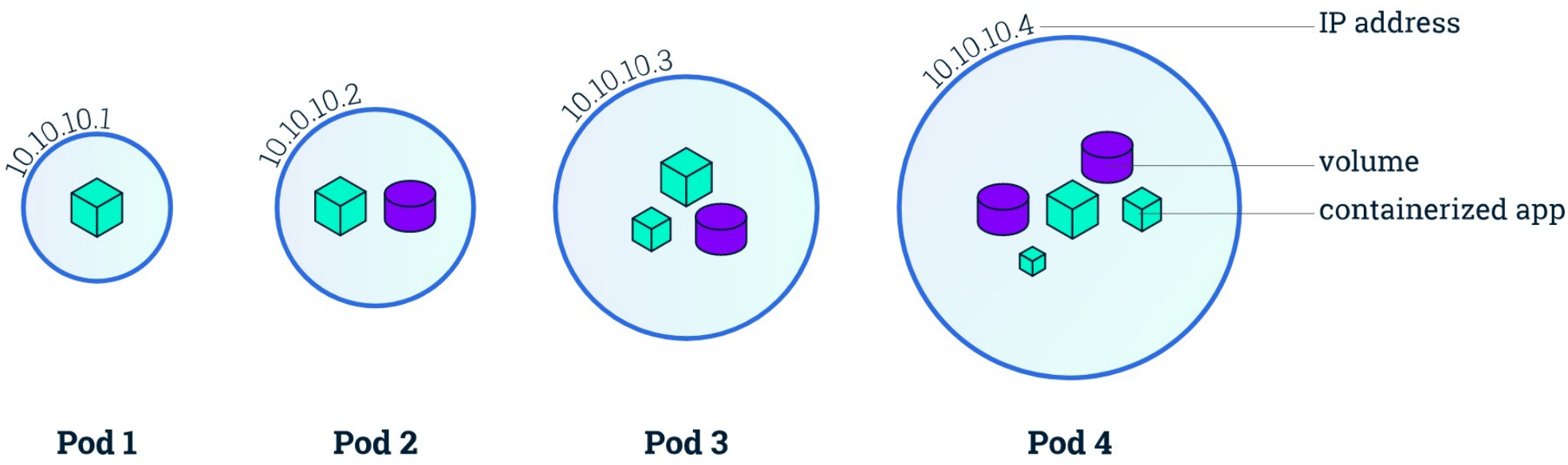
curl http://localhost:8001/version

Pod

Pod — это абстрактный объект Kubernetes, представляющий собой группу из одного или нескольких контейнеров приложения (например, Docker) и совместно используемых ресурсов для этих контейнеров. Ресурсами могут быть:

- **Общее хранилище (тома)**
- **Сеть (уникальный IP-адрес кластера)**
- **Информация по выполнению каждого контейнера (версия образа контейнера или используемые номера портов)**

Схема pod-ов


10.10.10.1
10.10.10.2
10.10.10.3
10.10.10.4
IP address
volume
containerized app
Pod 1
Pod 2
Pod 3
Pod 4

Node

Pod всегда работает в node. Node — это рабочая машина в Kubernetes, которая в зависимости от кластера может быть либо виртуальной, либо физической.

Каждый node управляется мастером (master node).

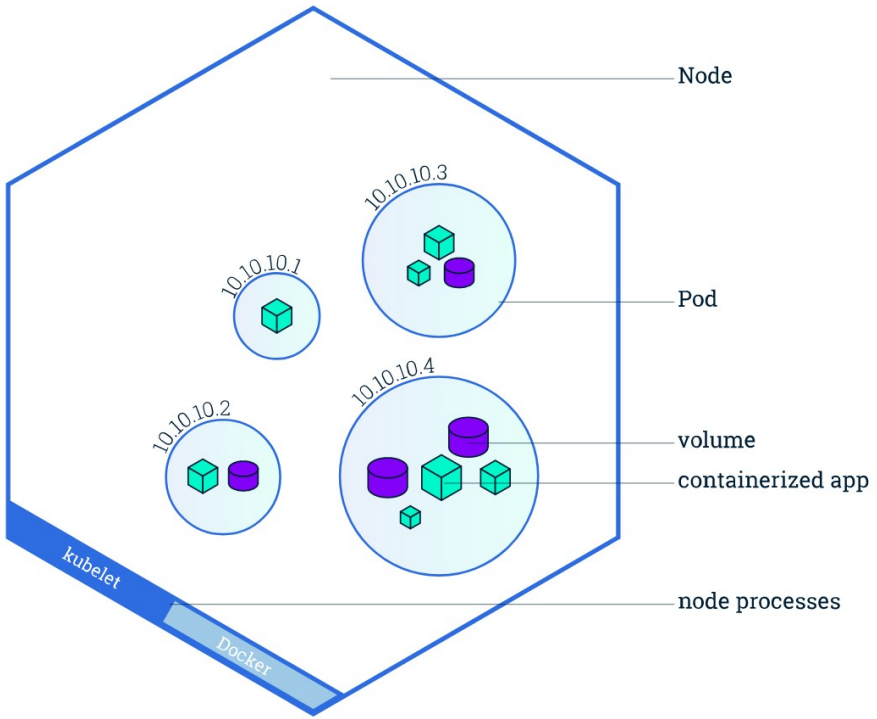
Node может содержать несколько pod-ов, которые мастер Kubernetes автоматически размещает на разные узлы кластера.

Master node при автоматическом планировании (распределении подов по узлам) учитывает доступные ресурсы на каждом узле.

Схема node-а

В каждом узле Kubernetes как минимум работает:

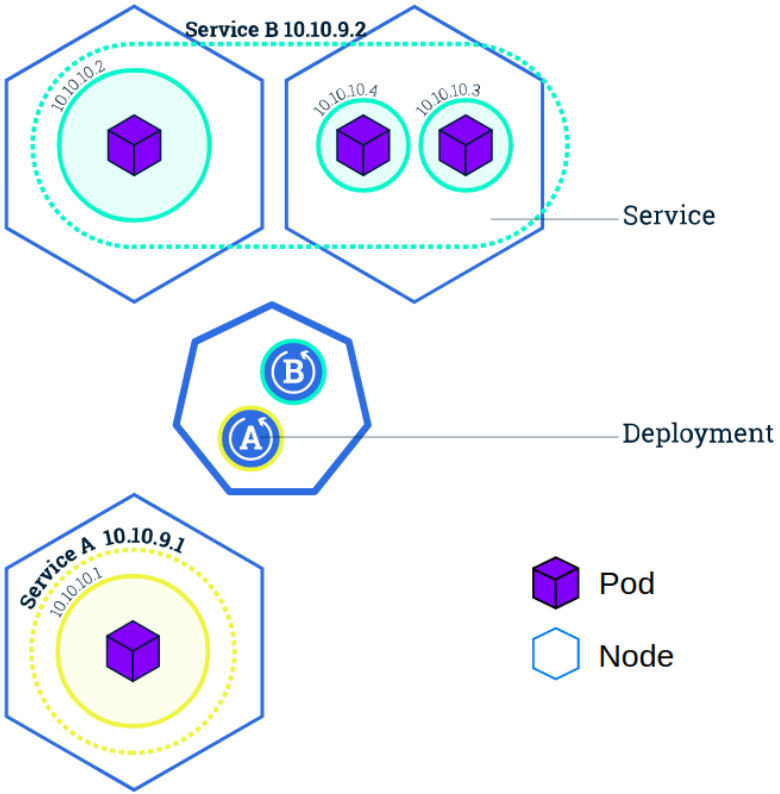
- **Kubelet — процесс, отвечающий за взаимодействие между мастером Kubernetes и узлом**
- **Среда выполнения контейнера (например, Docker)**



Изучение приложения. Практика

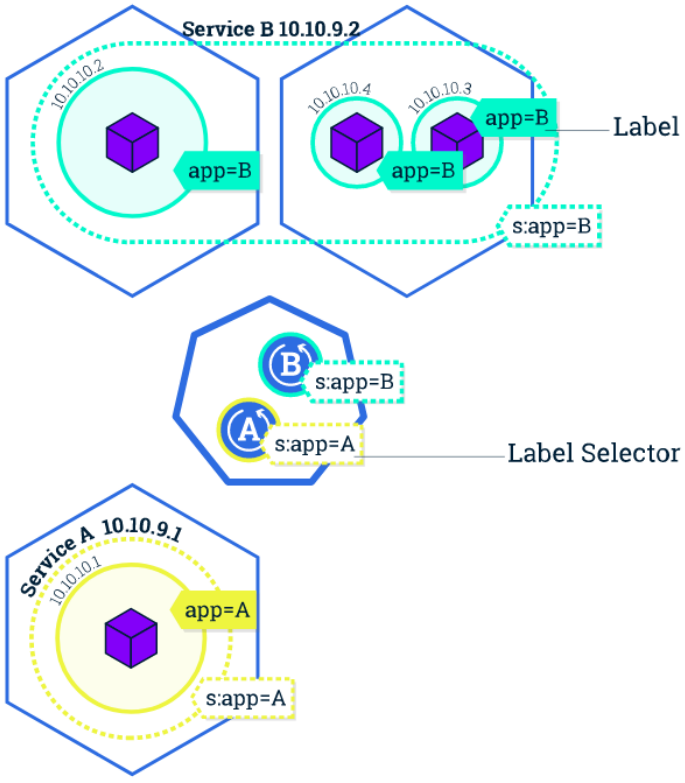
```
kubectl get pods
kubectl describe pods
kubectl proxy

curl http://localhost:8001/api/v1/namespaces/default/pods/$POD_NAME/proxy/

kubectl logs $POD_NAME

kubectl exec $POD_NAME env

kubectl exec -ti $POD_NAME bash
cat server.js
curl localhost:8080
```

Сервис в Kubernetes



- **Сервис в Kubernetes — это абстрактный объект, который определяет логический набор подов и политику доступа к ним.**
- **Сервисы создают слабую связь между подами, которые от них зависят.**
- **Сервис создаётся в формате YAML или JSON, как и все остальные объекты в Kubernetes.**

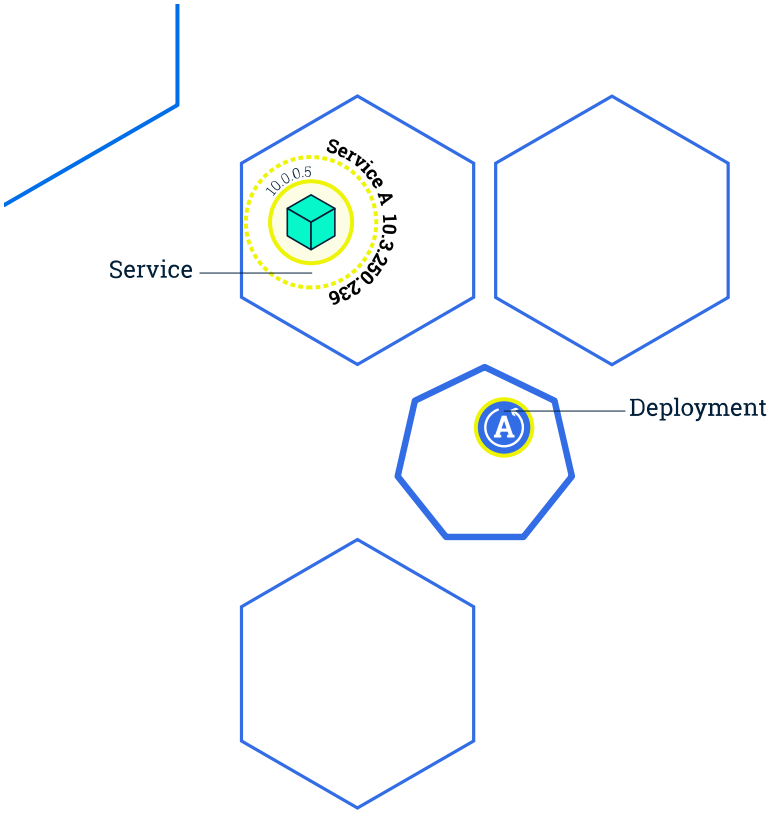
Метки

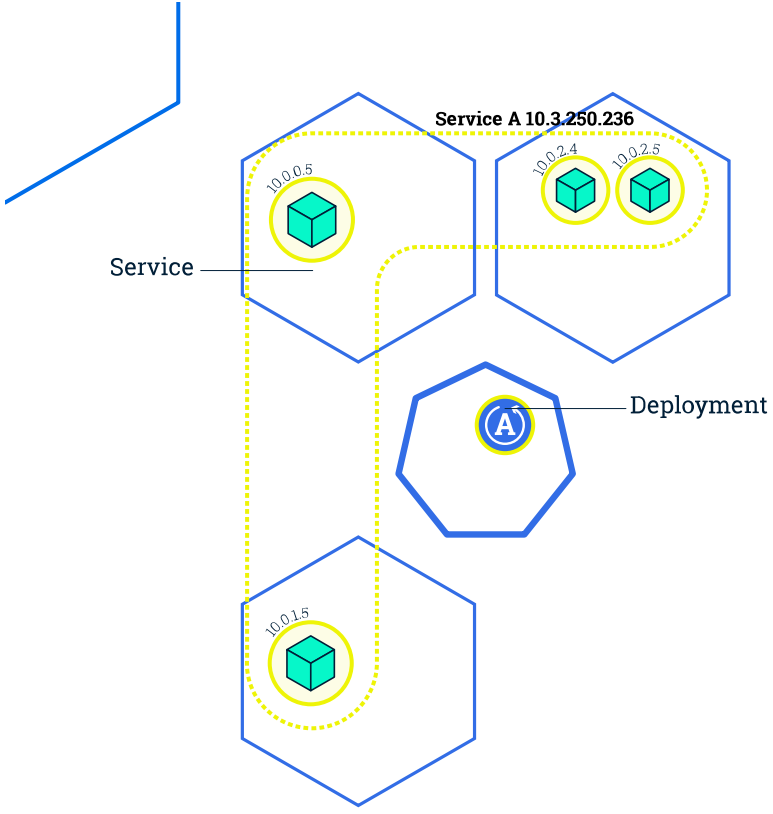


Метки — пары ключ-значение, добавленные к объектам; например, они могут использоваться чтобы:

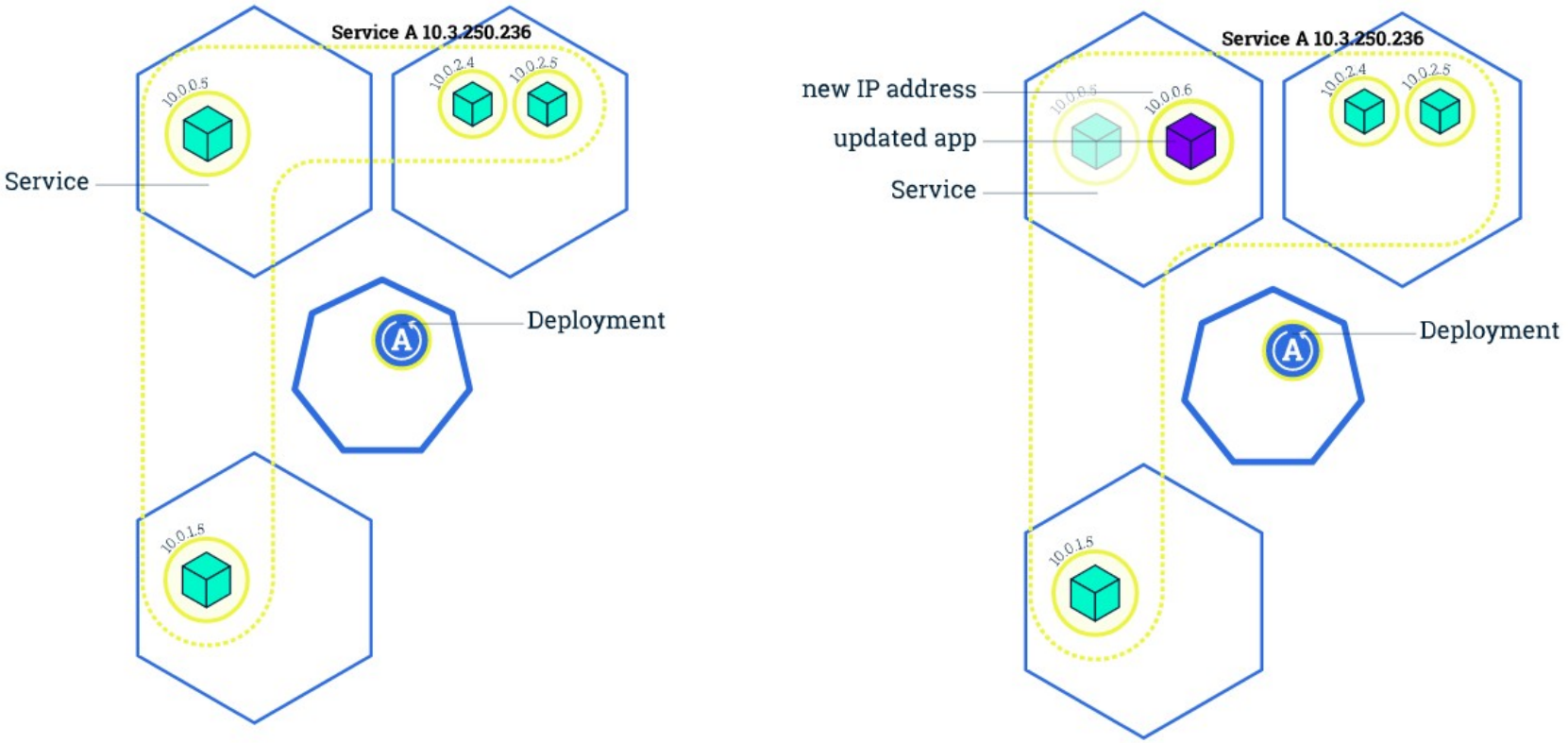
- **Идентифицировать объекты для окружений разработки, тестирования и продакшена**
- **Добавить теги версии**
- **Классифицировать объекты через теги**

Масштабирование приложения





Обновление приложения



Обновление приложения