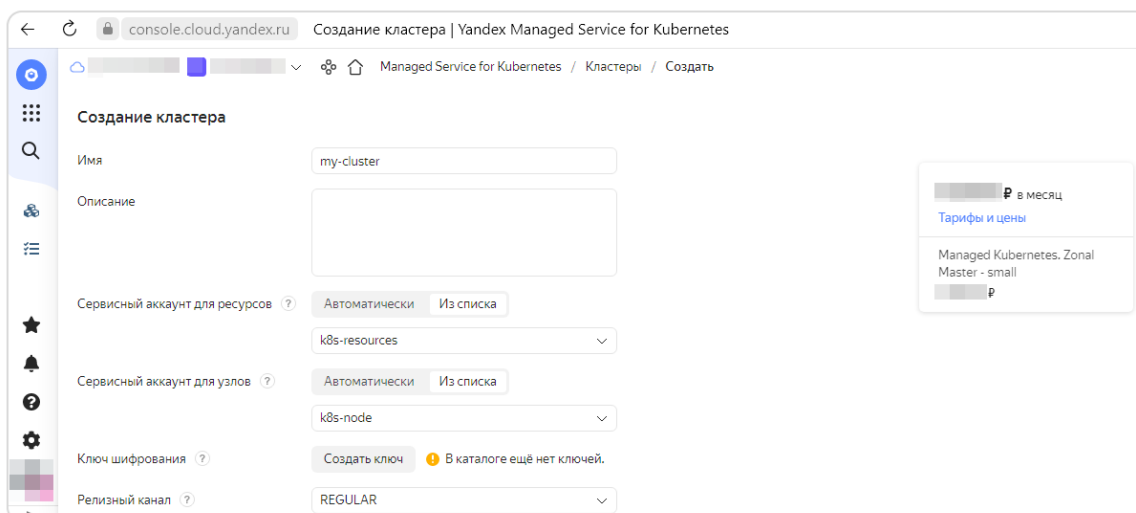


# Практическая работа. Создание кластера

В этой практической работе вы создадите кластер Kubernetes и группу узлов в нём.

1. Выберите каталог для кластера.
2. Выберите сервис **Managed Service for Kubernetes**. Нажмите кнопку **Создать кластер**. Дальше заполним настройки кластера:



The screenshot shows the 'Создание кластера' (Create Cluster) page in the Yandex Cloud console. The page is titled 'Создание кластера | Yandex Managed Service for Kubernetes'. The left sidebar contains navigation icons. The main form has the following fields:

- Имя** (Name): my-cluster
- Описание** (Description): (empty text area)
- Сервисный аккаунт для ресурсов** (Service account for resources): Automatically (selected) / From list
- Сервисный аккаунт для узлов** (Service account for nodes): Automatically (selected) / From list
- Ключ шифрования** (Encryption key): Create key (button) / In catalog there are no keys (note)
- Релизный канал** (Release channel): REGULAR

On the right, there is a pricing box showing 'Managed Kubernetes, Zonal Master - small' with a price of 'P в месяц' (P per month) and a link to 'Тарифы и цены' (Rates and prices).

3. Для Kubernetes необходим сервисный аккаунт для ресурсов и узлов.

**Сервисный аккаунт для ресурсов** — это аккаунт, под которым сервисы Kubernetes будут выделяться ресурсы в нашем облаке.

**Сервисный аккаунт для узлов** необходим уже созданным узлам самого кластера Kubernetes для доступа к другим ресурсам. Например, чтобы получить Docker-образы из Container Registry.

Этим аккаунтам нужны разные права, и поэтому у них бывают [разные роли](#). В общем случае вы можете использовать один и тот же сервисный аккаунт.

4. Ключ шифрования [Yandex Key Management Service](#) позволяет защитить конфиденциальную информацию (пароли, OAuth-токены и SSH-ключи) и повысить безопасность. Это необязательно — кластер запустится и без ключа. Для этой практической работы не создавайте его.
5. [Релизные каналы](#) `RAPID`, `REGULAR` и `STABLE` отличаются процессом обновления и доступными вам версиями Kubernetes.

`RAPID` и `REGULAR` содержат все версии, включая минорные. `STABLE` — только стабильные версии. `RAPID` обновляется автоматически, а в `REGULAR` и `STABLE` обновление можно отключить. Когда появляется обновление, информация о нём отображается в консоли управления.

Выберите `REGULAR`.

Внимательно выбирайте релизный канал! Изменить его после создания кластера Kubernetes нельзя.

## Конфигурация мастера

Мастер — ведущая нода группы узлов кластера — следит за состоянием Kubernetes и запускает управляющие процессы. Сконфигурируем мастер:

The screenshot shows the 'Создание кластера' (Create Cluster) page in the Yandex Cloud console. The left sidebar contains navigation icons. The main area is titled 'Конфигурация мастера' (Master Configuration). The settings are as follows:

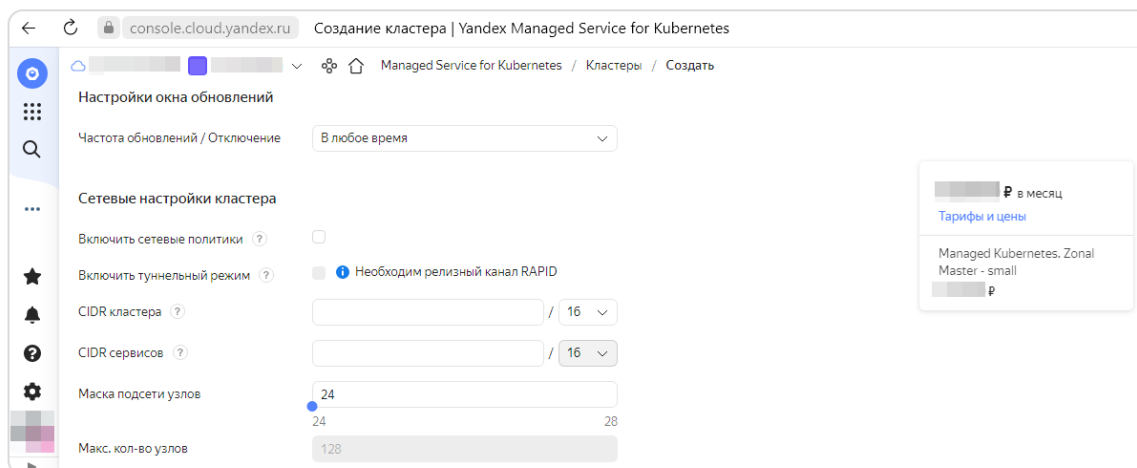
- Версия Kubernetes: 1.20
- Публичный адрес: Автоматически
- Тип мастера: Зональный
- Зона доступности: ru-central1-a
- Облачная сеть: intro2
- Подсеть: intro2-ru-central1-a
- Группы безопасности: Без групп

On the right, a price tag indicates the cost: 'P в месяц' (P per month) for 'Managed Kubernetes. Zonal Master - small'.

6. Выберите версию Kubernetes. Их набор зависит от релизного канала. Версии мастера и других нод могут не совпадать, но это достаточно тонкая настройка, могут возникнуть проблемы совместимости, которые повлияют на работу всего кластера.

7. Кластеру может назначаться публичный IP-адрес. Выберите вариант **Автоматически**. В этом случае IP выбирается из пула свободных IP-адресов. Если вы не используете Cloud Interconnect или VPN для подключения к облаку, то без автоматического назначения IP-адресов вы не сможете подключиться к кластеру: он будет доступен только во внутренней сети вашего облака.
8. Тип мастера влияет на отказоустойчивость. **Зональный** работает только в одной зоне доступности, а **региональный** — в трёх подсетях в каждой зоне доступности.  
Выберите зональный тип. В будущем для рабочей среды используйте региональные кластеры, а для разработки и тестирования — более дешёвые зональные.
9. Выбор типа мастера также влияет на подсети, в которых будет развёрнут кластер. У вас уже есть подсети, созданные по умолчанию для функционирования облака. Выберите их.

## Настройки окна обновлений



10. Режимов обновления четыре: **Отключено**, **В любое время**, **Ежедневно** и **В выбранные дни**. Региональный мастер во время обновления остаётся доступен, зональный — нет. Группа узлов кластера обновляется с выделением дополнительных ресурсов, так как при обновлении создаются узлы с обновлённой конфигурацией. При обновлении поды с

контейнерами будут переезжать с одного узла на другой.  
По умолчанию выставлен пункт **В любое время**. Оставьте его.

## Сетевые настройки кластера

11. **Сетевые политики** для кластера Kubernetes необязательны.

Эта опция включает сетевой контроллер [Calico](#), который позволяет применять тонкие настройки политик доступа для кластера.

Не выбирайте эту опцию.

12. Во время работы кластера подам с контейнерами и сервисам самого кластера Kubernetes будут автоматически присваиваться внутренние IP-адреса. Чтобы IP-адреса подов и сервисов Kubernetes не пересеклись с другими адресами в вашем облаке, задайте **CIDR** (Classless Inter-Domain Routing — бесклассовая междоменная маршрутизация). Оставьте адреса пустыми: они будут назначены автоматически.

**Маска подсети узлов** влияет на количество подов, которые могут запускаться. Если адресов не хватит, под не запустится.

Вы заполнили все настройки, теперь нажмите **Создать кластер**.

Дождитесь, пока статус кластера станет `RUNNING`, а состояние — `HEALTHY`. Это может занять около 10 минут.

## Создание группы узлов

Зайдите в созданный кластер, перейдите на вкладку **Управление узлами** и нажмите **Создать группу узлов**. Группы узлов — это группы виртуальных машин.

Введите имя и описание группы, выберите версию Kubernetes.

Выберите **Автоматический** тип масштабирования и количество узлов от 1 до 5. Укажите среду запуска контейнеров — Docker.

В сетевых настройках задайте автоматический IP-адрес и выберите зону доступности (кластер зональный, поэтому зона доступности только одна). Задайте SSH-ключ, чтобы иметь доступ к виртуальным машинам кластера. Настройки обновления идентичны настройкам мастера.

Остальные настройки группы, которые мы не упомянули (вычислительные ресурсы, хранилище и т. д.), оставьте по умолчанию. Нажмите **Создать группу узлов** и дождитесь, пока операция выполнится.

# Практическая работа. Первое приложение в кластере

1. Основное средство взаимодействия с кластером — инструмент [kubectl](#). Установите его по [инструкции](#).
2. В консоли управления войдите в созданный кластер Managed Service for Kubernetes и нажмите кнопку **Подключиться**. В открывшемся окне скопируйте команду для подключения:

Скопировать код

```
yc managed-kubernetes cluster get-credentials \  
  --id <идентификатор_кластера> \  
  --external
```

3. Чтобы проверить правильность установки и подключения, посмотрите на конфигурацию:

Скопировать код

```
kubectl config view
```

4. Ответ получится примерно таким (IP-адрес сервера и название кластера будут отличаться):

```
apiVersion: v1  
clusters:  
- cluster:  
  certificate-authority-data: DATA+OMITTED  
  server: https://178.154.206.242  
  name: yc-managed-k8s-cat2oek6hbp7mnhhhr4m  
contexts:  
...
```

## Создание манифеста

Для описания настроек приложения в кластере создадим файл `my-nginx.yaml`. Такой файл называется **манифестом**.

### Скопировать код

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-nginx-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: cr.yandex/<идентификатор_реестра>/ubuntu-nginx:latest
```

Рассмотрим, из чего он состоит.

3. Директива `apiVersion` определяет, для какой версии Kubernetes написан манифест. От версии к версии обозначение может меняться.

```
apiVersion: apps/v1
```

4. Директива `kind` описывает механизм использования. Она может принимать значения `Deployment`, `Namespace`, `Service`, `Pod`, `LoadBalancer` и т. д. Для развёртывания приложения укажите значение `Deployment`.

```
kind: Deployment
```

5. Директива `metadata` определяет метаданные приложения: имя, метки ([labels](#)), аннотации.  
С помощью **Меток** можно идентифицировать, группировать

объекты, выбирать их подмножества. Добавляйте и изменяйте метки при создании объектов или позднее, в любое время.

**Аннотации** используют, чтобы добавить собственные метаданные к объектам.

Укажем имя приложения:

```
metadata:
  name: my-nginx-deployment
```

## 6. В основном блоке `spec` содержится описание объектов Kubernetes.

Директива `replicas` определяет масштабирование. Для первого запуска укажите, что приложению нужен один под. Позже вы посмотрите, как приложения масштабируются, и сможете увеличить число подов.

Директива `selector` определяет, какими подами будет управлять контейнер (подробнее о ней можно прочитать [в документации](#)).

Поды отбираются с помощью метки (`label`).

Директива `template` определяет шаблон пода. Метка в шаблоне должна совпадать с меткой селектора — `nginx`.

В шаблоне содержится ещё одна, собственная директива `spec`. Она задаёт настройки контейнеров, которые будут развёрнуты на поде. Нам нужен один контейнер. Используйте для него образ, [созданный ранее](#) с помощью Docker и помещённый в реестр Yandex Container Registry.

```
spec:
  matchLabels:
    app: nginx
  replicas: 1
  selector: ~
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image:
            "cr.yandex/<идентификатор_реестра>/ubuntu-nginx:latest"
```



## Выполнение манифеста

7. Для создания или обновления ресурсов в кластере используется команда `apply`. Файл манифеста указывается после флага `-f`.

### Скопировать код

```
kubectl apply -f <путь_к_файлу_my-nginx.yaml>
```

8. Если результат будет успешным, вы увидите сообщение:

```
deployment.apps/my-nginx-deployment created
```

9. Чтобы убедиться, что приложение создано, посмотрите список подов:

### Скопировать код

```
kubectl get pods
```

10. Дождитесь статуса `Running`:

NAME	READY	STATUS	RESTARTS	AGE
my-nginx-deployment-65b9b678b6-zmfww	1/1	Running	0	5m27s

11. Теперь получите более подробную информацию, выполнив ту же команду с флагом `-o wide`:

### Скопировать код

```
kubectl get pods -o wide
```

12. Вы увидите внутренний IP-адрес, который присвоен поду. Это пригодится, если нужно узнать, где именно развёрнуто приложение.

Чтобы получить максимально подробную информацию о запущенном приложении, используйте команду `describe`:

### Скопировать код

```
kubectl describe deployment/my-nginx-deployment
```

## Масштабирование

9. Теперь увеличьте количество подов. Вручную это можно сделать двумя способами:

- изменить файл манифеста, указав в директиве `replicas` нужное число подов, и снова выполнить команду `apply`;
- если файла манифеста нет под рукой — использовать команду `scale`:

### Скопировать код

```
kubectl scale --replicas=3 deployment/my-nginx-deployment
```

Если всё получится, в выводе команды `kubectl get pods` вы увидите сообщение:

NAME	READY	STATUS	RESTARTS	AGE
my-nginx-deployment-65b9b678b6-6whpp	1/1	Running	0	
117s				
my-nginx-deployment-65b9b678b6-wtph9	1/1	Running	0	
117s				
my-nginx-deployment-65b9b678b6-zmfww	1/1	Running	0	14m

## Кластер как код

Как видите, управление кластерами Kubernetes отлично вписывается в концепцию Infrastructure as Code: вы можете описать конфигурацию кластера в текстовом файле — манифесте. Вы также можете разворачивать кластеры Kubernetes с помощью [Terraform](#).

# Практическая работа. Балансировка нагрузки

Большинство веб-приложений созданы, чтобы взаимодействовать через интернет. Вы развернули в кластере приложение, но у вас пока нет к нему доступа из интернета. Чтобы исправить эту проблему, воспользуемся [сервисом LoadBalancer](#).

У созданного пода есть внутренний IP-адрес.

Помните, мы говорили о том, что в кластере есть собственный сервис DNS? Он работает с внутренними IP-адресами объектов кластера, чтобы те могли взаимодействовать.

Однако внутренний IP-адрес может меняться, когда ресурсы группы узлов обновляются. Чтобы обращаться к приложению извне, требуется неизменный публичный IP-адрес — это и будет IP-адрес балансировщика.

## 1. Создайте файл-манифест `load-balancer.yaml`:

Скопировать код

```
apiVersion: v1
kind: Service
metadata:
  name: my-loadbalancer
spec:
  selector:
    app: nginx
  ports:
    - port: 80
      targetPort: 80
  type: LoadBalancer
```

Где:

`port` — порт сетевого балансировщика, на котором будут обслуживаться пользовательские запросы;

`targetPort` — порт контейнера, на котором доступно приложение;

`selector` — метка селектора из шаблона подов в манифесте объекта `Deployment`.

## 2. Выполните манифест:

Скопировать код

```
kubectl apply -f <путь_к_файлу_load-balancer.yaml>
```

## 3. Вы увидите сообщение:

Скопировать код

```
service/my-loadbalancer created
```

4. В консоли управления откройте раздел **Load Balancer**. Там должен появиться балансировщик нагрузки с префиксом k8s в имени и уникальным идентификатором кластера Kubernetes.
5. Скопируйте IP-адрес балансировщика в адресную строку браузера. Вы увидите приветственную страницу NGINX.