

# Лекция 17: Основы облачных вычислений. Где "живут" наши Kubernetes-кластеры

## Введение: Своя серверная или аренда?

Здравствуйте! На протяжении последних лекций мы говорили о контейнерах и оркестрации. Мы проектировали сложные системы, работающие на "кластере серверов". Но откуда берутся эти серверы?

Исторически компании следовали подходу **On-Premise**: они самостоятельно закупали физические серверы, устанавливали их в стойки в своем дата-центре (или арендованном), настраивали сеть, охлаждение и нанимали команду системных администраторов для их обслуживания. Этот подход требует огромных начальных вложений (CapEx), он медленный (поставка "железа" занимает месяцы) и негибкий (нельзя быстро добавить или убрать мощности).

Сегодня доминирует другой подход — **Cloud Computing (облачные вычисления)**. Вместо покупки "железа" мы **арендуем** вычислительные ресурсы у крупных провайдеров (таких как Amazon, Google, Яндекс) через интернет. Мы платим только за то, что реально используем, как за электричество, и можем получить новый сервер за пару минут.

Сегодня мы разберемся, из каких "кубиков" состоят облачные платформы, какие модели сервисов они предоставляют, и как они упрощают нашу жизнь, особенно в контексте работы с Kubernetes.

## 1. Модели облачных сервисов: IaaS, PaaS, SaaS

Облачные сервисы предоставляют разный уровень "абстракции". Понимание этих трех моделей — ключ к пониманию всего облачного мира.

### Аналогия: приготовление пиццы

- **On-Premise (Свои серверы):** Готовить пиццу **полностью с нуля**. Вы сами выращиваете пшеницу, мелете муку, доите корову для сыра, строите печь.
- **IaaS (Infrastructure as a Service — Инфраструктура как сервис):** Аренда кухни с **печью**. Провайдер дает вам базовые "строительные блоки" (виртуальные серверы, диски, сети), а вы сами устанавливаете на них ОС и все ПО.
- **PaaS (Platform as a Service — Платформа как сервис):** Доставка замороженной **пиццы**. Провайдер предоставляет вам готовую платформу. Вам нужно только "загрузить" свой код (или контейнер) и нажать "Запустить".
- **SaaS (Software as a Service — ПО как сервис):** Заказ **готовой, горячей пиццы с доставкой**. Вы просто пользуетесь готовым приложением (Gmail, Jira, 1C:Fresh).

Как разработчики, мы чаще всего работаем с **IaaS** и **PaaS**.

## 2. Ключевые "строительные блоки" облака

Любой облачный провайдер предоставляет набор стандартных сервисов, из которых, как из LEGO, собирается инфраструктура.

- **Compute (Вычисления):** "Мозги" вашей системы.
  - **Виртуальные машины (IaaS):** Основа основ. (Amazon EC2, Yandex **Compute Cloud**).
  - **Управляемый Kubernetes (PaaS):** Самый популярный сервис. Провайдер берет на себя всю сложную работу по управлению **Control Plane** кластера. Вы управляете только рабочими узлами. (Amazon **EKS**, Google **GKE**, Yandex **Managed Service for Kubernetes**).
  - **Serverless / FaaS (PaaS):** Вы вообще не думаете о серверах, а просто загружаете код функции. Облако само ее запускает и масштабирует (AWS **Lambda**, Yandex **Cloud Functions**).
- **Storage (Хранилища):** "Память" вашей системы.
  - **Блочное хранилище:** Виртуальные "жесткие диски", которые прикрепляются к виртуальным машинам (Amazon **EBS**, Yandex **Cloud Disks**).
  - **Объектное хранилище (S3):** Бесконечно масштабируемое хранилище для файлов (аватарки, видео, бэкапы). Стандарт де-факто — Amazon **S3**, все остальные провайдеры делают S3-совместимые аналоги.
- **Databases (Базы данных):** Специализированные **PaaS**-сервисы.
  - **Проблема:** Самостоятельно администрировать кластер базы данных (настраивать репликацию, бэкапы, обновления) — очень сложно.
  - **Решение: Управляемые базы данных (Managed Databases).** Вы просто говорите облаку: "Дай мне отказоустойчивый кластер PostgreSQL", и оно делает всю работу за вас. Вы получаете только строку подключения. (Amazon **RDS**, Yandex **Managed Service for PostgreSQL**).

Современное облачное приложение — это конструктор: код запускаем в **Управляемом Kubernetes**, базу данных берем **Управляемую**, а файлы пользователей храним в **Объектном хранилище**.

### 3. Обзор рынка: Мировые гиганты и российские лидеры

На мировом рынке доминирует "большая тройка" американских IT-гигантов: **Amazon Web Services (AWS)** — безоговорочный лидер и пионер, **Microsoft Azure** — силен в Enterprise-сегменте, и **Google Cloud Platform (GCP)** — технологический лидер в области Kubernetes и Big Data.

В России сформировался свой собственный сильный рынок, что особенно актуально в текущих условиях. Ключевые игроки:

- **Yandex.Cloud (Yandex Cloud):** Технологический лидер в РФ с очень широким набором PaaS-сервисов.
- **VK Cloud (ранее Mail.ru Cloud Solutions):** Один из крупнейших и старейших игроков, силен в IaaS и геймдеве.
- **SberCloud (Cloud.ru), Selectel** и другие.

Знание принципов работы одного облака позволяет легко адаптироваться к любому другому, так как набор базовых сервисов у всех очень похож.

## 4. Практика: Создаем инфраструктуру в Yandex.Cloud

*(Здесь в устной лекции следует переход к демонстрации экрана с консолью Yandex.Cloud)*

Давайте посмотрим, как за несколько минут можно "поднять" инфраструктуру для нашего приложения, на ручную настройку которой раньше уходили недели.

### 1. Создаем Управляемый кластер PostgreSQL:

- Заходим в сервис "Managed Service for PostgreSQL".
- Нажимаем "Создать кластер".
- Выбираем минимальную конфигурацию, задаем имя базы, пользователя и пароль.
- Через 5-10 минут облако само создает для нас отказоустойчивый кластер, настраивает репликацию и бэкапы. Мы получаем готовую к использованию строку подключения.

### 2. Создаем Управляемый кластер Kubernetes:

- Заходим в сервис "Managed Service for Kubernetes".
- Нажимаем "Создать кластер".
- Создаем как минимум одну группу рабочих узлов (например, 2 машины типа t2.medium).
- Через 10-15 минут мы получаем полностью рабочий, настроенный и защищенный Kubernetes-кластер. Control Plane управляется Яндексом, мы его даже не видим.

### 3. Развертываем приложение:

- Теперь мы можем настроить kubectl для работы с этим новым облачным кластером.
- И выполнить те же самые команды kubectl apply -f ., которые мы использовали на локальной машине.
- Наш Service типа LoadBalancer будет автоматически подхвачен облаком, и Яндекс создаст для него настоящий, публичный балансировщик нагрузки с "белым" IP-адресом.

## Заключение

Сегодня мы увидели, в чем заключается магия облачных вычислений. Они избавляют нас от сложного и дорогого процесса управления физической инфраструктурой и предоставляют мощные **управляемые сервисы (PaaS)**, которые позволяют разворачивать сложные системы, такие как Kubernetes, за считанные минуты.

Мы собрали почти все "кубики" современного backend-разработчика и DevOps-инженера. Нам осталось разобраться с последним, но не менее важным аспектом: **"Наблюдаемостью"**

**(Observability).** На следующей лекции мы поговорим о **Мониторинге и Логировании** и научимся понимать, что происходит с нашим приложением после того, как мы его развернули в облаке.