Свойства облачной инфраструктуры

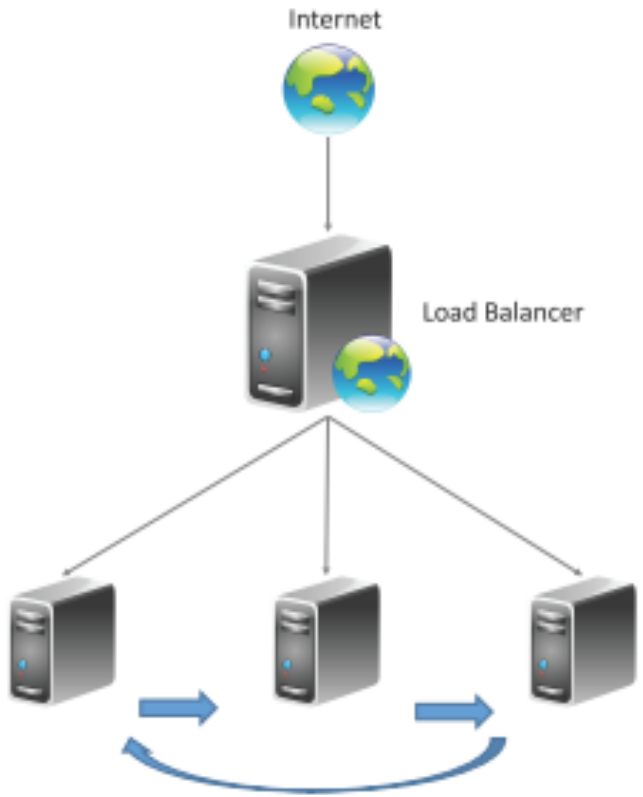
Антоненко Виталий

Балансировка нагрузки

- Ресурсы облака могут быть по запросу масштабированы (вверх) для удовлетворения требований облачного приложения.

- Балансировка нагрузки осуществляется для перераспределения нагрузки между серверами, которые обеспечивают работу приложения.

- Балансировка нагрузки преследует следующие цели:
  - Достижения максимального использования пула ресурсов
  - Минимизации времени отклика приложения
  - Максимизация пропускной способности



Алгоритмы балансировки нагрузки

- **Справедливость:** нужно гарантировать, чтобы на обработку каждого запроса выделялись системные ресурсы и не допустить возникновения ситуаций, когда один запрос обрабатывается, а все остальные ждут своей очереди;
- **Эффективность:** все серверы, которые обрабатывают запросы, должны быть заняты на 100%; желательно не допускать ситуации, когда один из серверов простаивает в ожидании запросов на обработку (сразу же оговоримся, что в реальной практике эта цель достигается далеко не всегда);
- **Сокращение времени выполнения запроса:** нужно обеспечить минимальное время между началом обработки запроса (или его постановкой в очередь на обработку) и его завершения;
- **Сокращение времени отклика:** нужно минимизировать время ответа на запрос пользователя.

- **Предсказуемость:** нужно чётко понимать, в каких ситуациях и при каких нагрузках алгоритм будет эффективным для решения поставленных задач;
- **Равномерная загрузка ресурсов системы;**
- **Масштабируемость:** алгоритм должен сохранять работоспособность при увеличении нагрузки

Алгоритмы балансировки нагрузки

- Процедура балансировки осуществляется при помощи целого комплекса алгоритмов и методов, соответствующим следующим уровням модели OSI:
  - сетевому;
  - транспортному;
  - прикладному.

Алгоритмы балансировки нагрузки

- **Round Robin**, или алгоритм кругового обслуживания, представляет собой перебор по круговому циклу: первый запрос передаётся одному серверу, затем следующий запрос передаётся другому и так до достижения последнего сервера, а затем всё начинается сначала.

- **Weighted Round Robin** — усовершенствованная версия алгоритма Round Robin. Суть усовершенствований заключается в следующем: каждому серверу присваивается весовой коэффициент в соответствии с его производительностью и мощностью.

Алгоритмы балансировки нагрузки

- **Least Connections** – учитывает количество подключений, поддерживаемых серверами в текущий момент времени. Каждый следующий вопрос передаётся серверу с наименьшим количеством активных подключений.

- Существует усовершенствованный вариант этого алгоритма, предназначенный в первую очередь для использования в кластерах, состоящих из серверов с разными техническими характеристиками и разной производительностью. Он называется **Weighted Least Connections** и учитывает при распределении нагрузки не только количество активных подключений, но и весовой коэффициент серверов.

Алгоритмы балансировки нагрузки

- Алгоритм **Destination Hash Scheduling** был создан для работы с кластером кэширующих прокси-серверов, но он часто используется и в других случаях. В этом алгоритме сервер, обрабатывающий запрос, выбирается из статической таблицы по IP-адресу получателя.

- Алгоритм **Source Hash Scheduling** основывается на тех же самых принципах, что и предыдущий, только сервер, который будет обрабатывать запрос, выбирается из таблицы по IP-адресу отправителя.

Алгоритмы балансировки нагрузки

- **Sticky Sessions** — алгоритм распределения входящих запросов, при котором соединения передаются на один и тот же сервер группы. Он используется, например, в веб-сервере Nginx. Сессии пользователя могут быть закреплены за конкретным сервером с помощью метода IP hash.

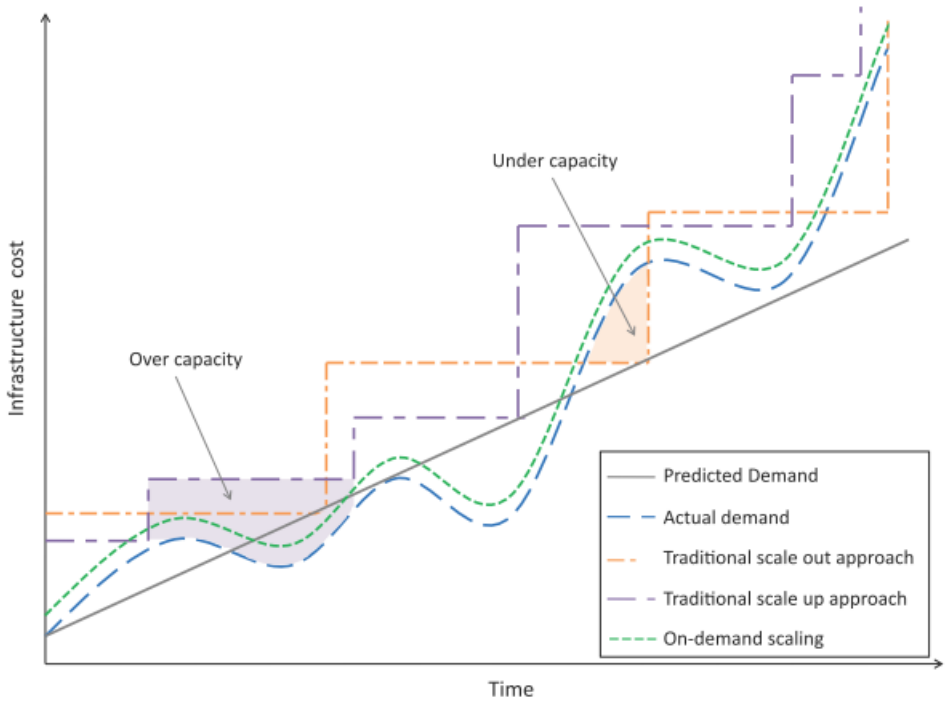
Балансировка нагрузки – устоявшиеся походы

- Для успешной балансировки нагрузки между множейством серверов важным аспектов является управления состоянием сетевых сессий.

- Устоявшиеся подходы
  - Прикрепленные сессии (Sticky sessions)
  - БД сессий (Session Database)
  - «Печенки» в браузере (Browser cookies)
  - Перезапись URL (URL re-writing)

Масштабируемость & Эластичность

- Многоуровневые приложения, например, электронная коммерция, социальные сети, B2B и т.д. могут быстро изменять динамику сетевого трафика.
- Планирование ресурсов включает грамотное, эффективное размещение всех видом ресурсов на каждой стадии жизненного цикла приложения.
- Планирование ресурсов включает планирование вычислительных и сетевых ресурсов, ресурсов хранения и памяти.



Подходы к масштабированию

- Вертикальное масштабирование/ Масштабирование вверх:
  - Включает обновление количества физических ресурсов виртуальной сущности (добавление дополнительной памяти, диска или ядер ЦП).

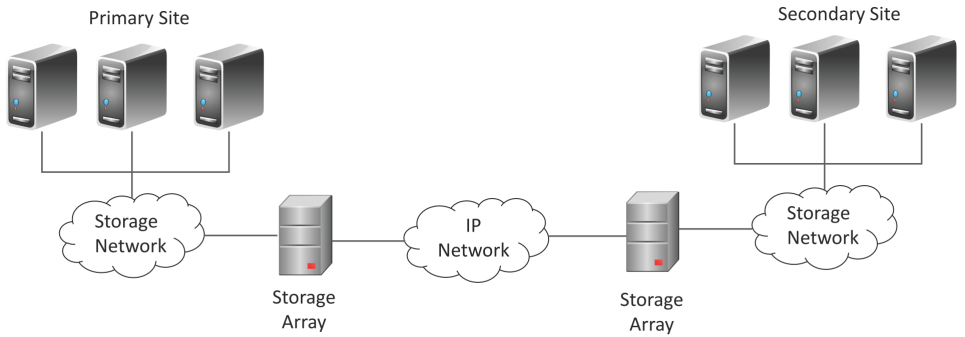
- Горизонтальное масштабирование/ Масштабирование вширь
  - Включает размещение дополнительных виртуальных сущностей того же типа (ВМ, контейнеры).

Размещение

- Размещение облачного приложения это итеративный процесс, который включает:
  - Планирование ресурсов (Deployment Design)
    - В качестве входные параметров на данной стадии используются количество серверов, размеры жесткий дисков, ОП, количество ядер ЦП, скорость сети, стратегии балансировки нагрузки и резервного копирования.
  - Анализ производительности (Performance Evaluation)
    - Верификация удовлетворяет ли выделенная физическая инфраструктура требования приложения.
    - Включает мониторинг рабочей нагрузки, оценка ключевых параметров рабочей нагрузки, напмриер, время задержки и пропускной способности.
    - Мониторинг использования (Utilization) физической инфраструктуры (ЦП, ОП, диск и т.д.).
  - Уточнения размещения (Deployment Refinement)
    - Изменения характеристик размещения с учетом вертикального или горизонтального масштабирования, альтернативных сетевых подключений, альтернативных стратегий балансировки и резервного копирования для каждой виртуальной сущности (ВМ, контейнер).

Резервное копирование

- Резервное копирование использует для создание множества копий виртуальной сущности в облаке.
- Облако представляет быстрый механизм резервного копирования для обеспечения аварийного восстановление после сбоев.
- Используя облачную инфраструктуру нет необходимости держать дублирующую физическую инфраструктуру для обеспечения аварийного восстановления.



Array-based Replication

Мониторинг

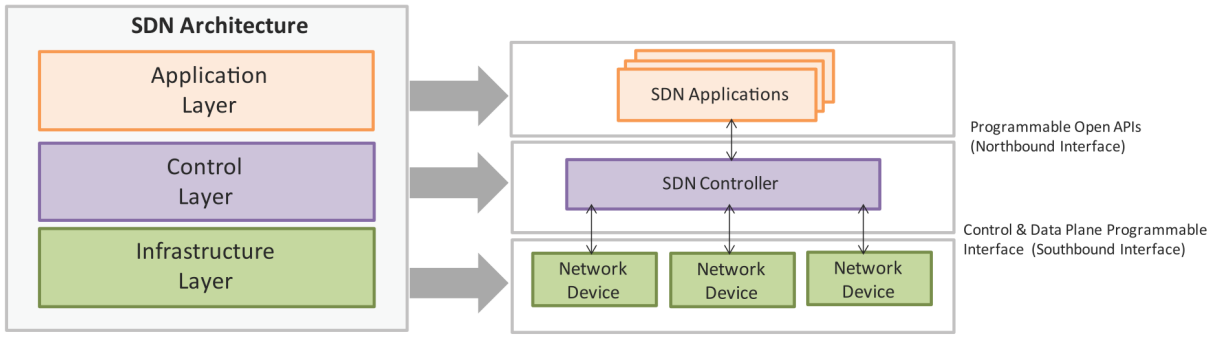
- Мониторинг позволяет пользователям облака собирать и анализировать данные, использую различные метрики.
- Мониторинг собирает данные о работе как облачных приложение, так и физической инфраструктуры облака.
- Мониторинг позволяет пользователям оценить текущее состоянием приложения и принимать решения и восстановлении/ масштабировании/ репликации.

Примеры метрик для мониторинга

Тип	Метрика
CPU	CPU-Usage, CPU-Idle
Disk	Disk-Usage, Bytes/sec (read/write), Operations/sec
Memory	Memory-Used, Memory-Free, Page-Cache
Interface	Packets/sec (incoming/outgoing), Octets/sec(incoming/outgoing)

Программно-конфигурируемые сети

- Программно-конфигурируемые сети (ПКС) для сетевая архитектура, которая разделяет уровень управления сетью с уровнем передачи данных, предоставляя централизованное управления с помощью ПКС контроллера.
- Традиционная сетевая архитектура
  - Уровень управления и уровень данные объединены. Уровень управления это часть сетевой инфраструктуры, которая содержит и анализирует сигнальную и маршрутизирующую информацию, уровень данных это полезная нагрузка (Payload) сетевого трафика.
- ПКС архитектура
  - Уровень управления и уровень данных разведены. Управление осуществляется централизованным ПКС контроллером.



ПКС- Ключевые элементы

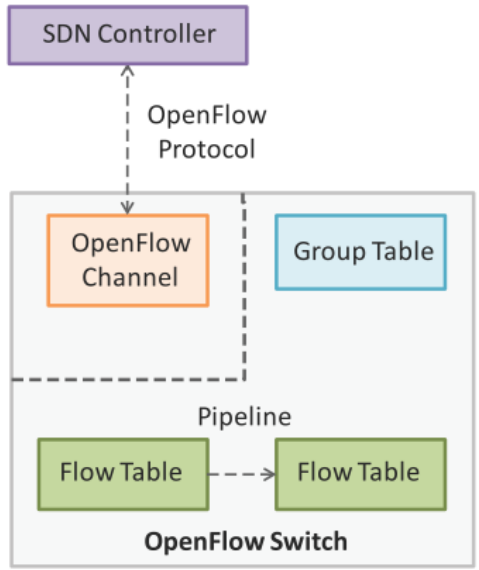
- Централизованный Сетевой Контроллер
  - Элемент позволяющих задать логику управления сетевыми устройствами.

- Программируемые открытые API
  - ПКС архитектура поддерживарое программируемые открытый API для взаимодействия приложений и уровня управления (Северный интерфейс). Данные открытые API позволяют реализовыват различные сетевые сервисы: маршрутизация, QoS, контроль доступа (ACL) и т.д.

- Стандартный интерфейс взаимодействия (OpenFlow)
  - ПКС архитектура использует стандартный интерфейс взаимодействия между контроллером и ПКС коммутаторами (Южный интерфейс). Протокол OpenFlow, которые стандартизован и поддерживается Open Networking Foundation (ONF) один из широко распространённых ПКС протоколов для Южного интерфейса.

OpenFlow

- Протокол для связи ПКС контроллера и ПКС коммутатора.
- С OpenFlow, маршрутизация потоков трафика задается непосредственно на ПКС коммутаторе через таблицу потоков.
- OpenFlow использует концепцию потоков трафика, чтобы идентифицировать сетевой трафик при помощи специализированные правил.
- Потоки могут программироваться статически или динамический в зависимости от запущенный приложений на ПКС контроллере.
- Протокол OpenFlow специфицирует интерфейс как для ПКС контроллера, так и для ПКС коммутатора.



ПКС (OpenFlow) коммутатор содержит одну или несколько таблиц потоков и групповых таблиц, которые обеспечивают обработку и маршрутизацию сетевых пакетов.
Так же каждый коммутатор имеет OpenFlow канал для связи с ПКС контроллером.

Виртуализация сетевых функций

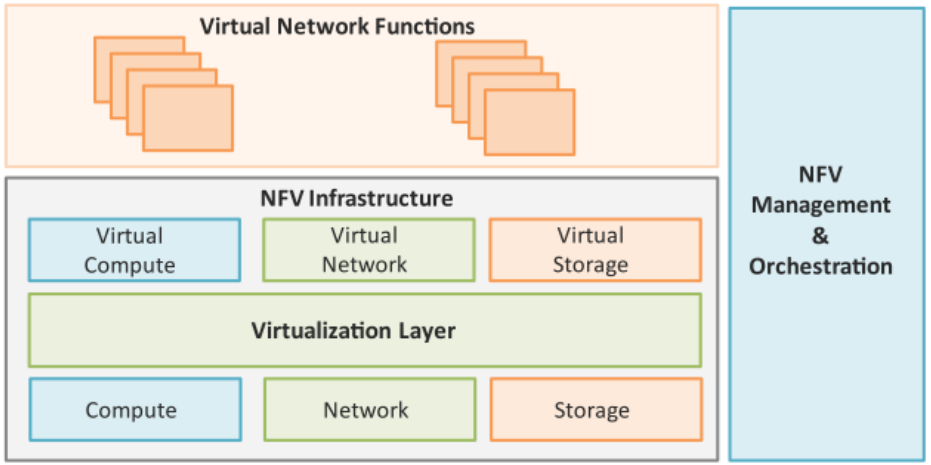
- Виртуализация сетевых функций (NFV) это технология, которая виртуализирует функциональностью физический сетевых устроств, позволяя перенести ее на архитектуру ЦОД (вычислительные серверва, сервера хранения + коммутаторы).

- Отношения к ПКС
  - NFV дружественная к SDN технология, так как NFV позволяет создавать инфраструктуру, к которой применима ПКС архитектура.
  - NFV и ПКС взаимовыгодны друг другу, но не зависят друг от друга.
  - Сетевые функции могут быть виртуализованы без ПКС, также как и ПКС может быть запущен без NFV.

- NFV программно реализует функции сетевых устройств, которые в дальнейшем запускаются на виртуальной инфраструктуре облака.
- NFV отделяет реализацию сетевых функций от непосредственного сетевого оборудования

Архитектура NFV

- Ключевыми элементами NFV архитектуры являются:
  - Виртуальная Сетевая Функция (VNF): VNF программная реализация сетевой функции, которая может запуститься на NFV инфраструктуре (NFVI).
  - NFV Инраструктура (NFVI): NFVI включает вычислительные, сетевые ресурсы, а также ресурсы хранения, которые могут быть виртуализованы.
  - NFV Управления и Оркестрами (MANO): NFV MANO фокусируется на всех задачах связанных с виртуализацией, которые возникают на всех стадия жизненного цикла VNF.



Управление идентификацией и доступом

- Управление идентификацией и доступом (IDAM) для облаков, описывает процесс аутентификации, авторизации пользователей для предоставления безопасного доступа к облачной инфраструктуре.
- Используя IDAM облачные сервисы получают многопользовательский доступ с возможностью разграничения прав каждого пользователя.
- IDAM позволяет централизовать управления правами пользователя, у также управления информационной безопасностью и ключами доступа.
- IDAM позволяет организовать ролевой доступ к ресурсам облачной инфраструктуры.
- IDAM позволяет создавать группы пользователей, где каждый член группы имеет равные права доступа к облачной инфраструктуре.
- IDAM реализует при помощи нескольких технологий, таких как OpenAuth, Role-based Access Control (RBAC), Digital Identities, Security Tokens, Identity Providers, и т.д.

Управление платежами

Провайдеры облачный услуг предлагаю множество моделей оплаты услуг, основными их них являются:

- Elastic Pricing
  - Модель – Плати за использование (pay-as-you-use), потребитель платит только за использованные ресурсы.
- Fixed Pricing
  - Модуль фиксированной цены, потребитель платить фиксированную цену в месяц за доступ к облачным ресурсам.
- Spot Pricing
  - Рыночная модели формирования цены. Чем больше желающих на конкретные ресурсы тем больше цена.

Спасибо за внимание!

Вопросы?

anvial@lvk.cs.msu.su

Антоненко Виталий