нетология

Микросервисы: масштабирование

Михаил
Триполитов

Михаил Триполитов

Technical Lead

Михаил Триполитов

План занятия

1. Scale Cube
2. Балансировка
3. Кэширование
4. Автомасштабирование
5. Service Discovery
6. Service Mesh
7. Итоги
8. Домашнее задание

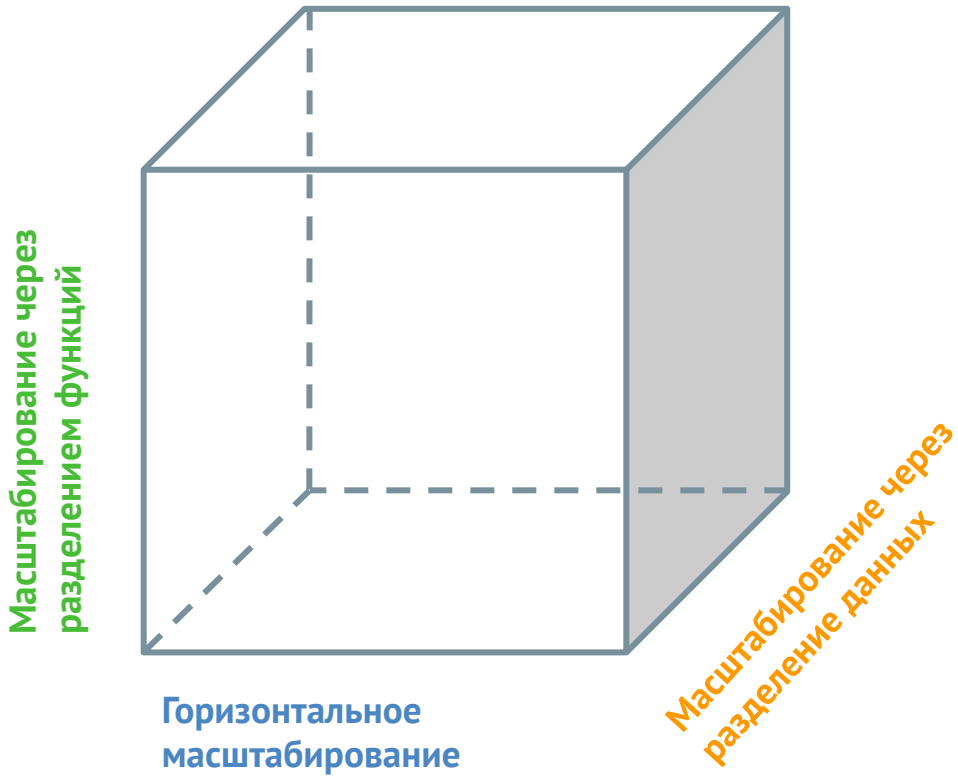
Scale Cube

Scale Cube

Ось X - горизонтальное масштабирование: запуск нескольких копий приложения

Ось Y - масштабирование разделением приложения: функции выделяются в отдельные сервисы и запускают независимо друг от друга

Ось Z - масштабирование через разделение данных: каждая запущенная копия сервиса отвечает за своё подмножество данных



Хрупкость системы

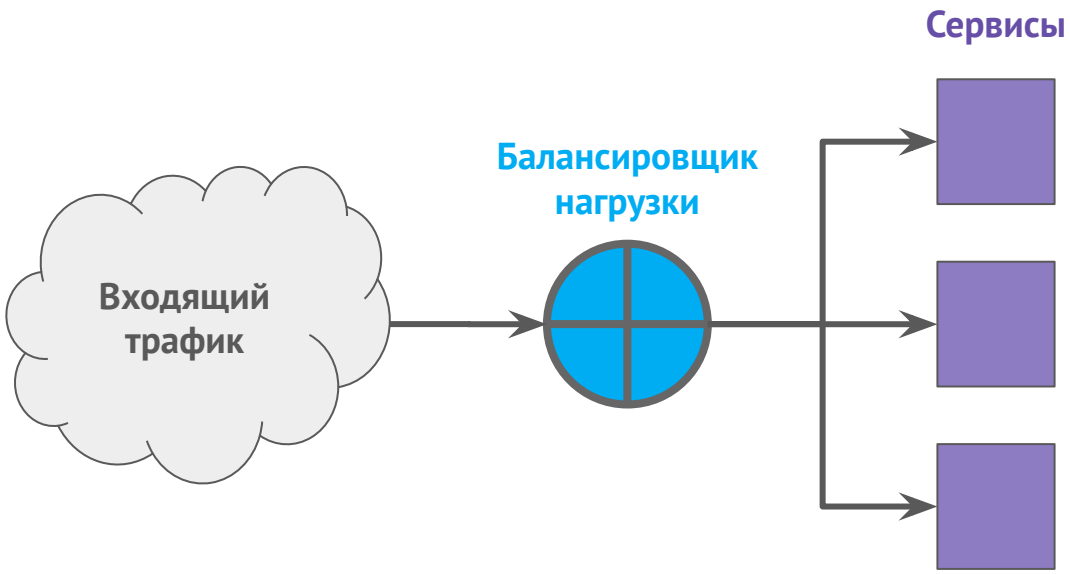
Способы защиты от хрупкости

- Максимальное время ожидания (Timeout)
- Повтор запроса (Retry)
- Прерыватели цепи (Circuit Breaker)
- Перегородки (Bulkhead)
- Ограничитель количества запросов (Rate Limiter)
- Изоляция
- Идемпотентность

Балансировка

Балансировщик нагрузки

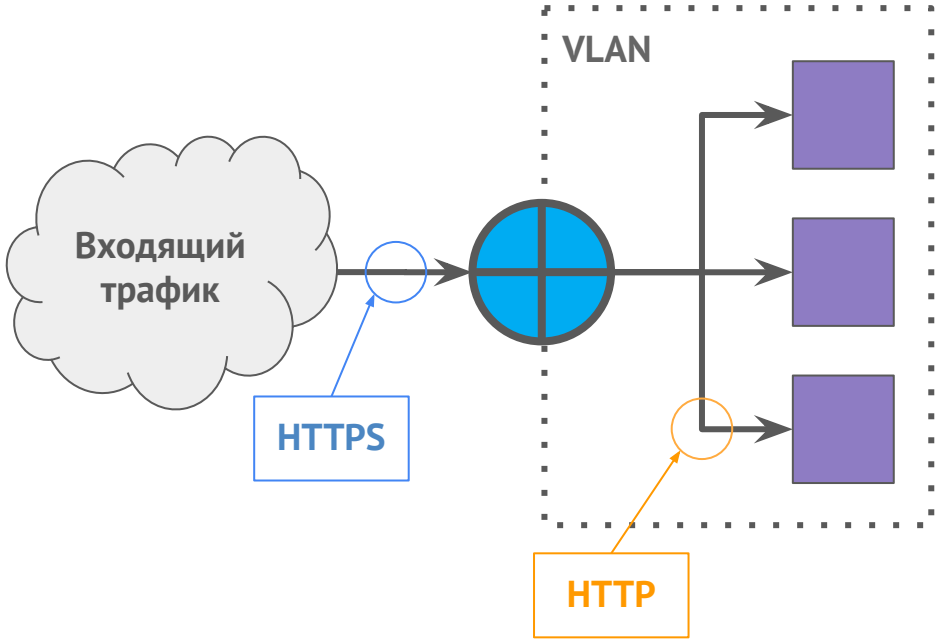
Основная функция: распределение входящих запросов по нескольким серверам



Балансировка

Дополнительные функции

- Терминация SSL
- Активация резервных серверов
- Ассиметричное распределение запросов
- Защита от DDOS атак
- Gzip сжатие HTTP
- Проверка работоспособности
- Кэширование HTTP
- Контентно зависимое распределение запросов
- Аутентификация клиентов
- Фильтрация контента



Балансировщики

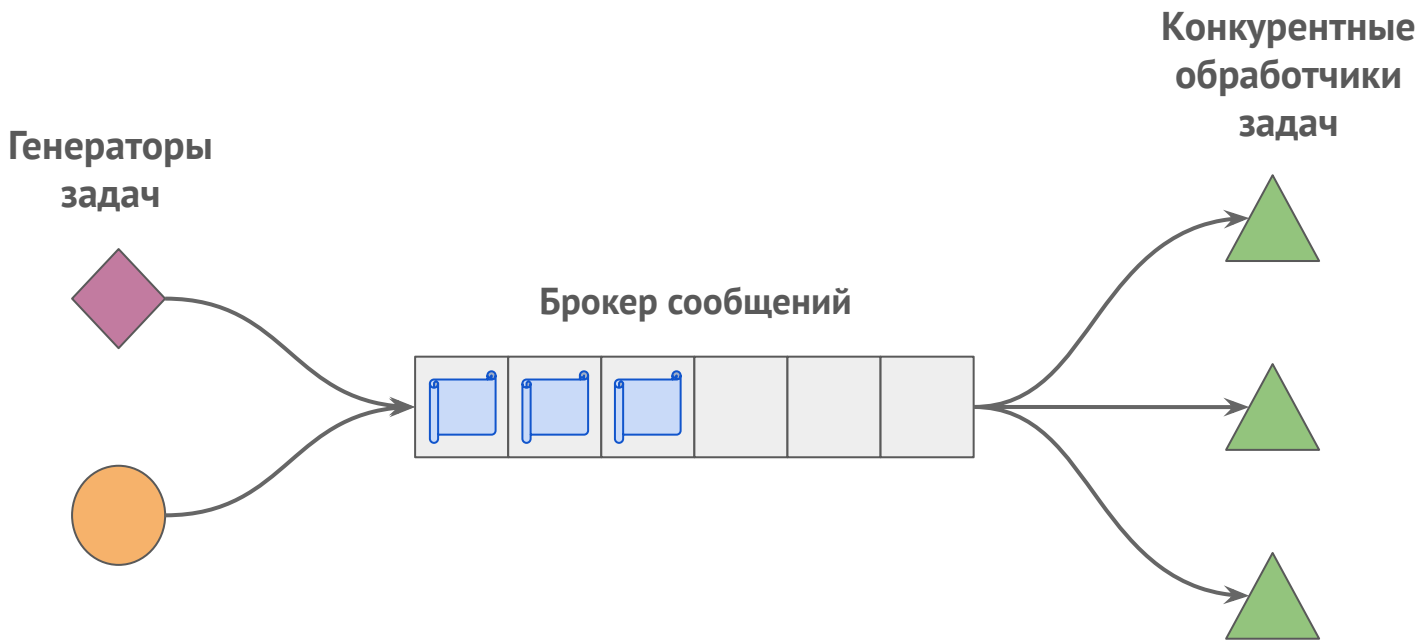
Балансировщики

- Nginx
- F5
- Kemp
- HAProxy
- Envoy

Облачные балансировщики

- AWS Elastic Load Balancer
- Google Cloud Load Balancing
- Azure Load Balancer
- Yandex Cloud Network Load Balancer

Системы обработки задач

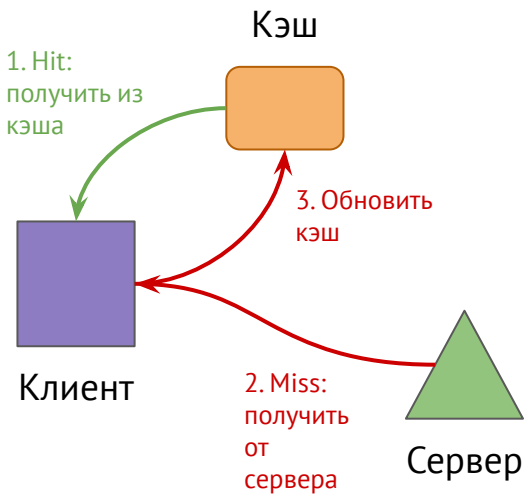


Кэширование

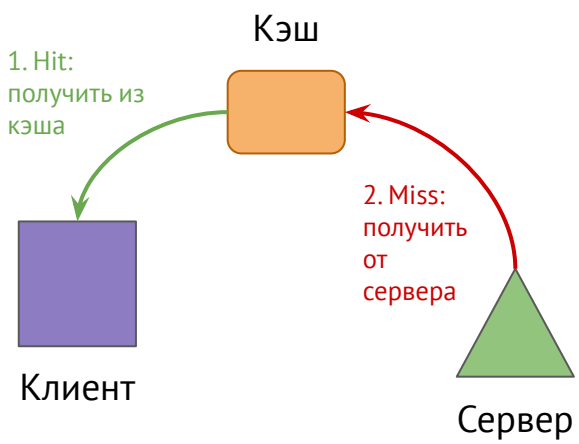
Кэширование

Кэширование - наиболее распространенный способ повышения производительности

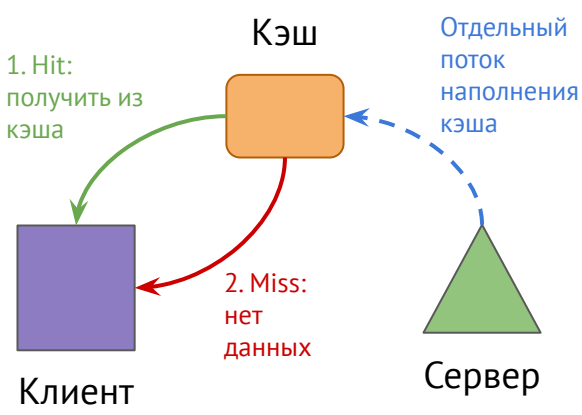
Клиентский кэш



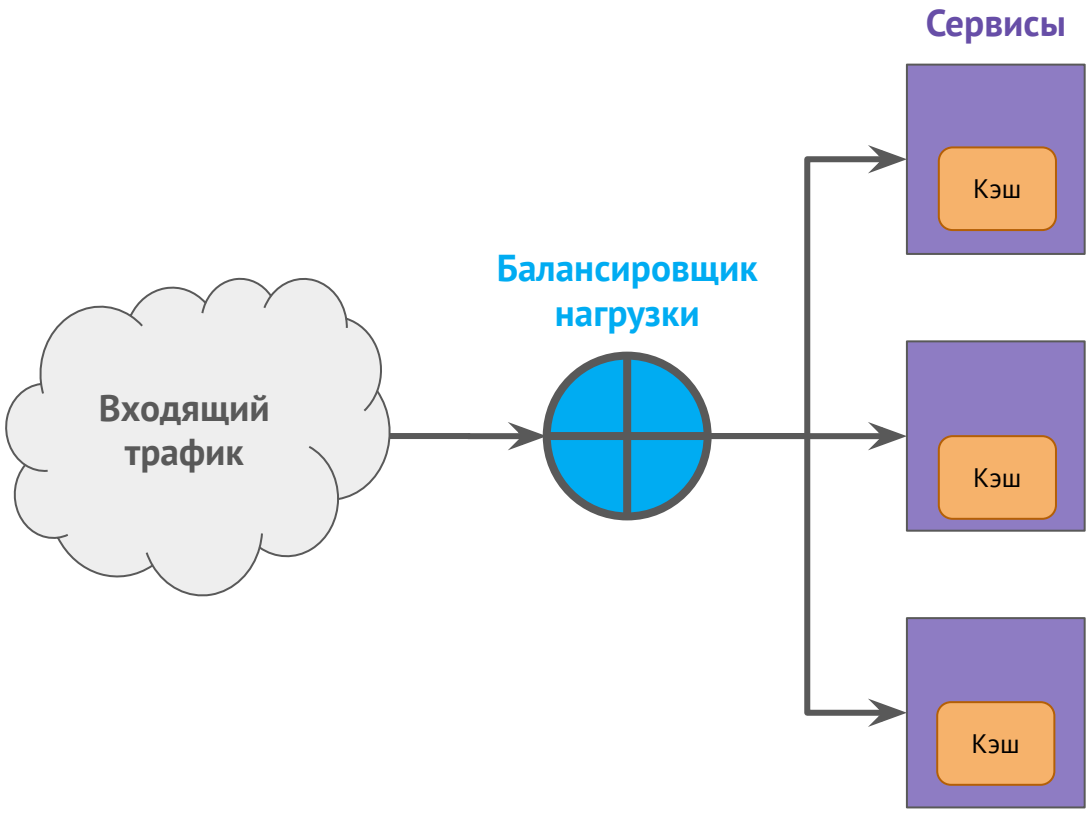
Сквозной кэш



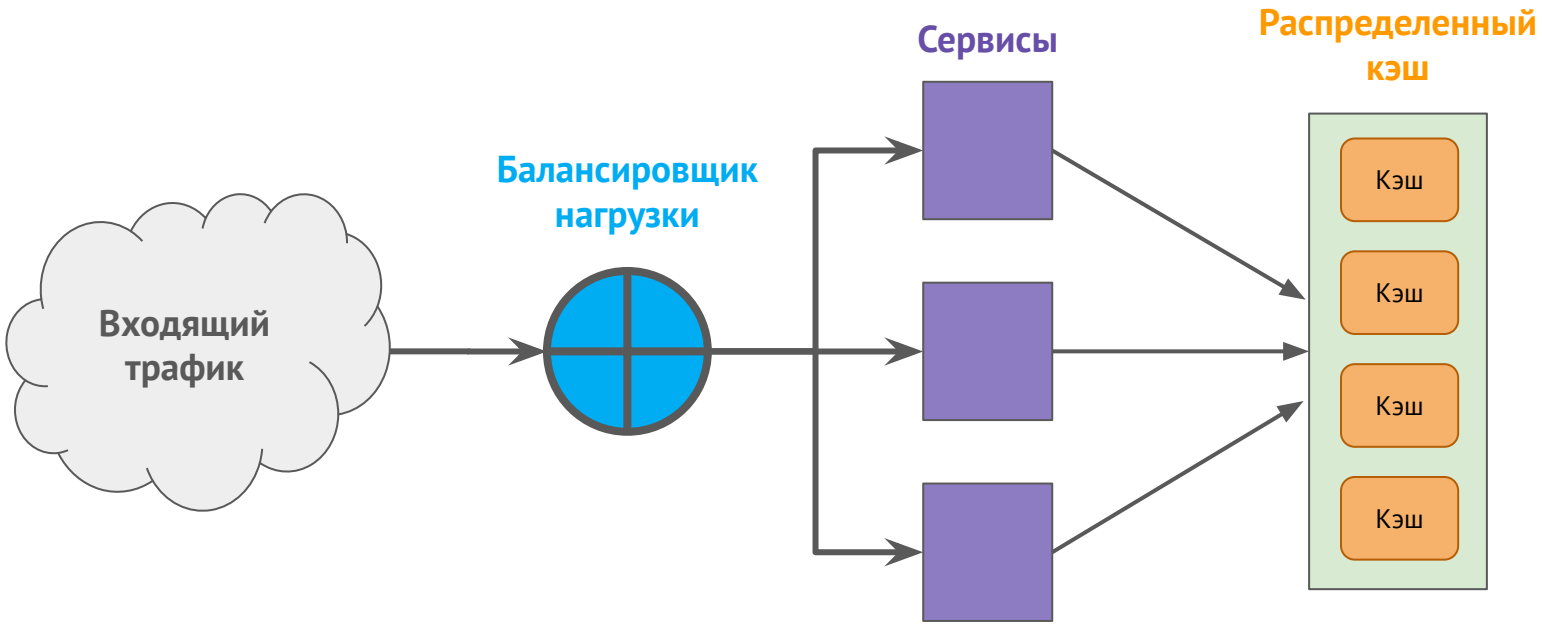
Прогретый кэш



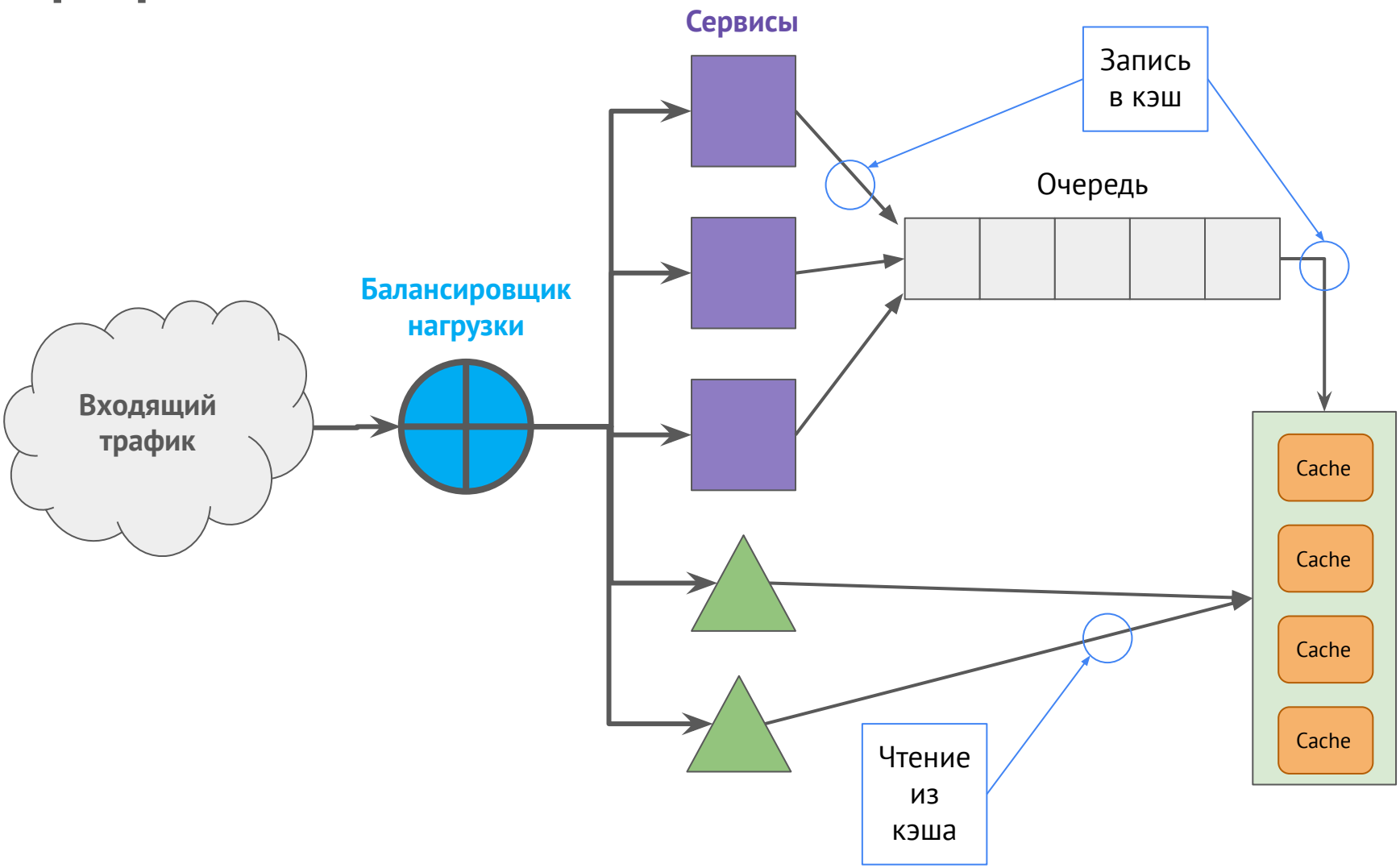
Кэш встроен в сервис



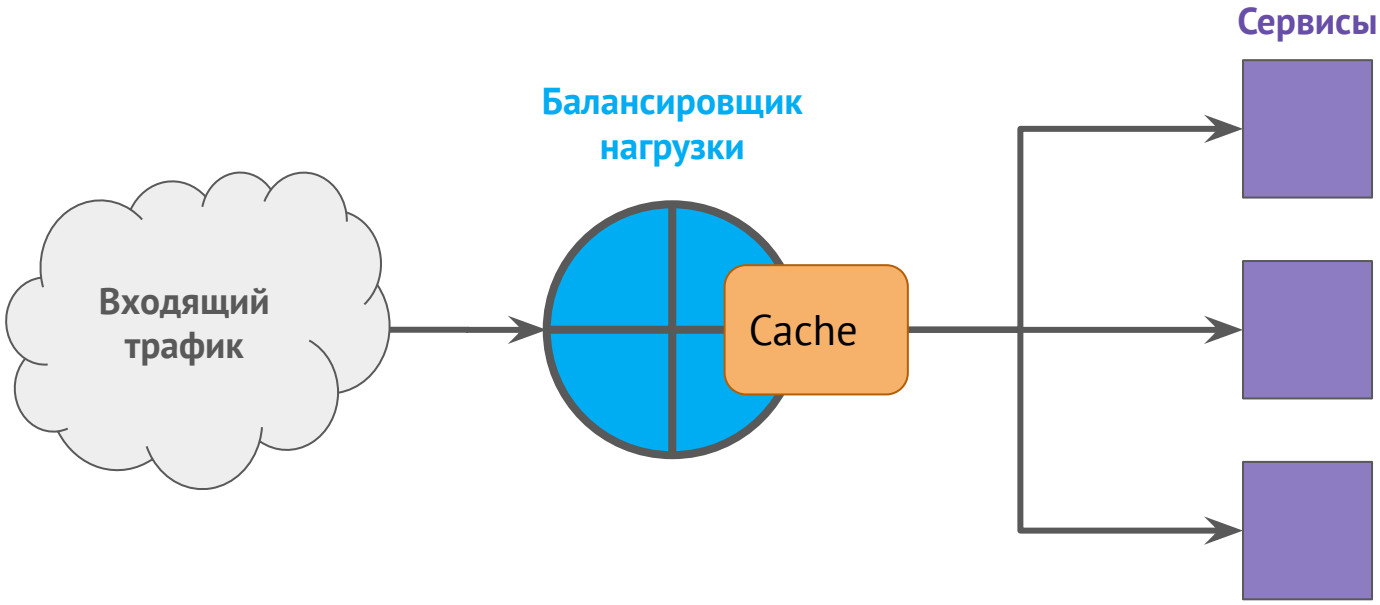
Распределенный кэш



Прогретый кэш



Кэширующий балансировщик



In-Memory Кеш

- Hazelcast
- Redis
- Memchached
- Tarantool

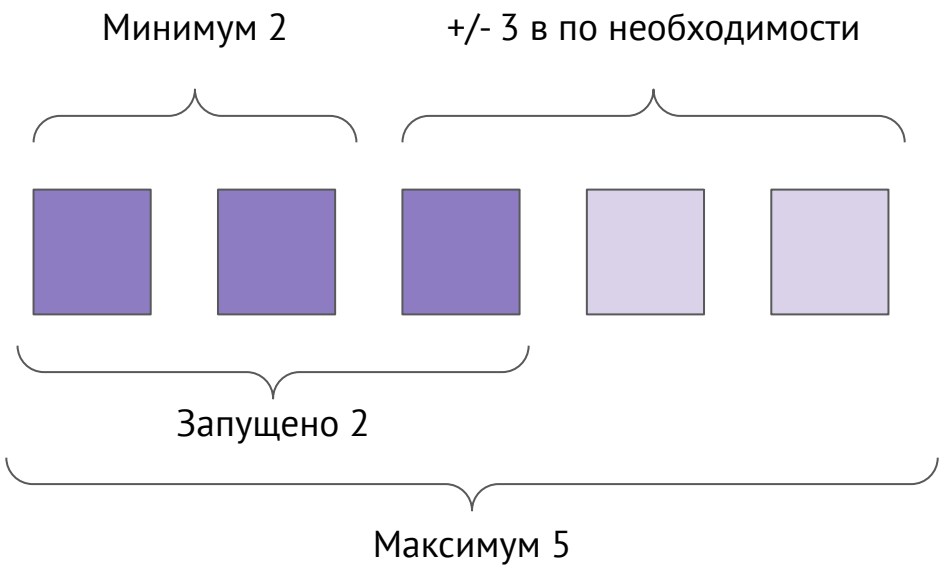
Автомасштабирование

Автоматическое масштабирование

Процесс динамического выделения ресурсов для удовлетворения требований производительности:

При увеличении нагрузки
выделяются дополнительные ресурсы для обеспечения соглашения об уровне обслуживания (SLA)

При снижении нагрузки
освобождаются ресурсы для экономии средств



Необходимые компоненты

- Мониторинг приложений и инфраструктуры
- Логика принятия решений
- Управление ресурсами
- Тестирование процедуры автомасштабирования

Подходы

- Динамическое масштабирование на основе метрик приложения или инфраструктуры:
  - Среднее значение CPU
  - Общее количество запросов
  - Длительность обработки задачи
  - Количество запросов завершившихся ошибкой
- Масштабирование по расписанию
- Предсказательное масштабирование

Автомасштабирование

- Kubernetes Horizontal Pod Autoscaler
- Amazon EC2 Auto Scaling
- Azure Monitor
- Google Compute Engine Autoscaling
- Yandex Cloud Cluster-Autoscaler

Service Discovery

Service Discovery

Настройки

У клиентов в конфигурации IP адреса сервисов с которыми ему нужно взаимодействовать

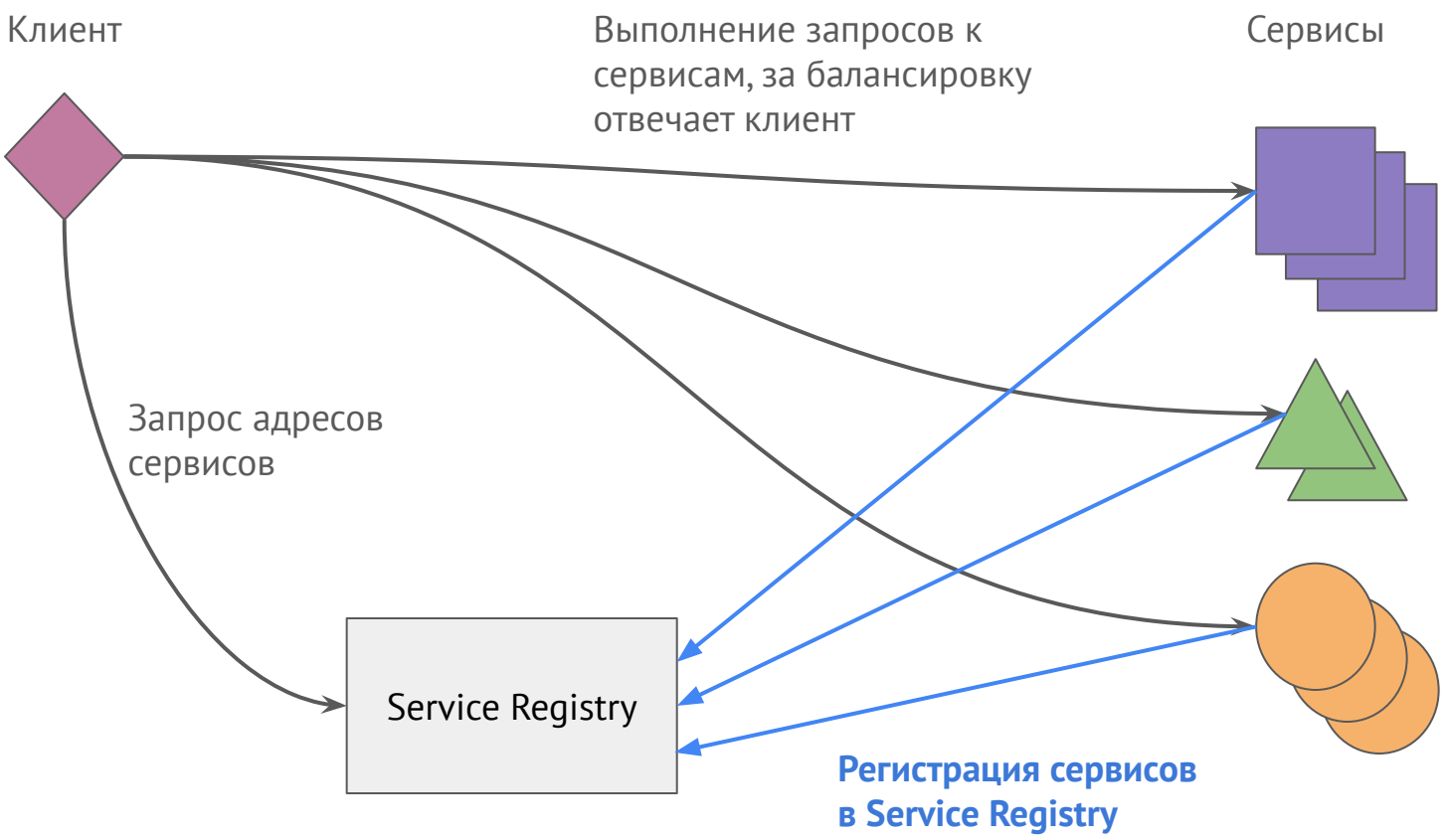
DNS

У каждого сервиса есть DNS имя за которым несколько IP адресов его запущенных экземпляров

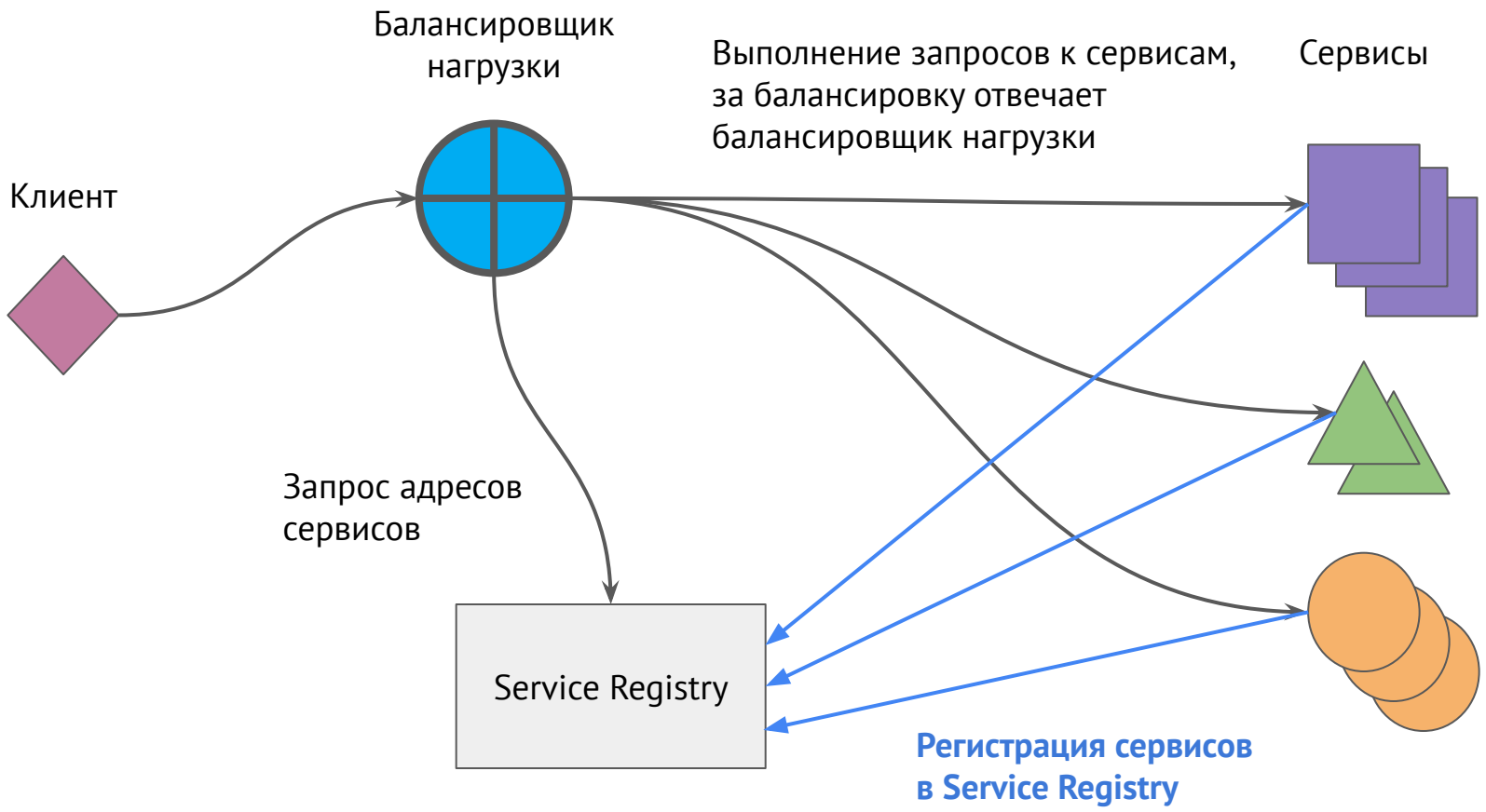
Service Registry

Специальное программное обеспечение позволяет сервисам при старте зарегистрировать адреса запускаемого экземпляра, а клиентам получить эти адреса по имени

Service Registry



Service Registry



Service Registry

- Zookeeper
- Consul
- Eureka
- Самодельный

Service Mesh

Задачи Service Mesh

- **Гибкость настройки** взаимодействия сервисов на уровне инфраструктуры, управление трафиком, балансировка
- **Унифицированный мониторинг** большой системы
- **Повышение безопасности**: отсутствие риска перехвата трафика, полный контроль над сетью, исключение угроз при работе в нескольких дата центрах или публичных облаках
- **Упрощение реализации сложных шаблонов развертывания** приложений: Blue/Green, Canary, A/B Tesing
- **Организация взаимодействия сервисов в мульти кластерной среде**

Принцип работы Service Mesh на примере Istio

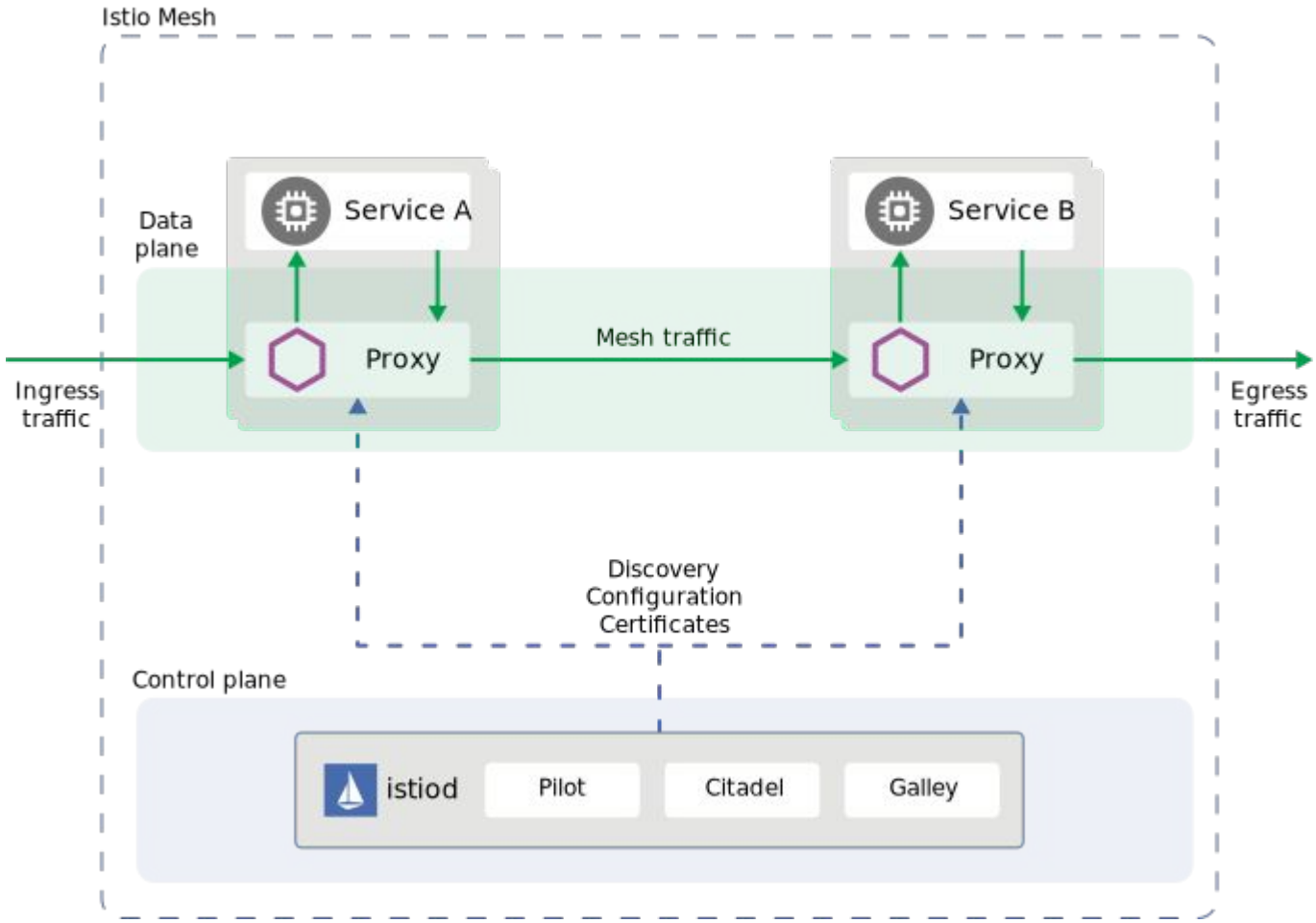


Схема с сайта https://istio.io/latest/docs/concepts/what-is-istio

Service Mesh

- Istio
- Linkerd
- Consul Connect
- Kuma
- Maesh
- OpenShift Service Mesh
- AWS App Mesh

Итоги

- Узнали важность непрерывной поставки для микросервисной архитектуры
- Познакомились со способами развертывания микросервисов
- Узнали про разные виды тестирования и изучили влияние пирамиды тестирования на результат
- Разобрали разные способы обеспечения аутентификации и авторизации
- Познакомились со способами мониторинга: метрики, логи, трассировка
- Обсудили масштабирование и кэш

Домашнее задание

Давайте посмотрим ваше домашнее задание.

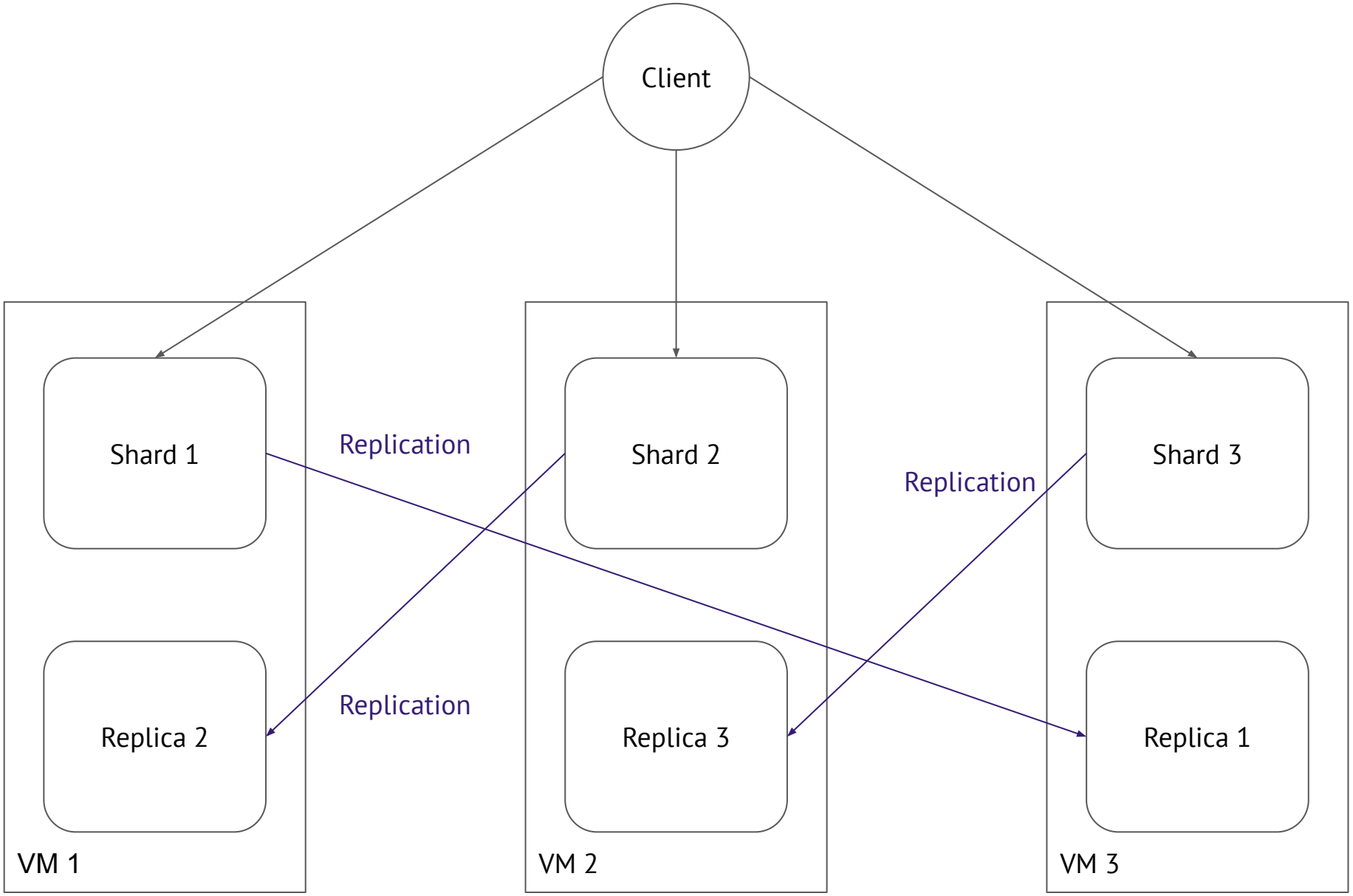
- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

нетология

Задавайте вопросы и пишите отзыв о лекции!

Михаил Триполитов

Михаил Триполитов


Client
Shard 1
Shard 2
Shard 3
Replication
Replication
Replication
Replica 2
Replica 3
Replica 1
VM 1
VM 2
VM 3