

Разработка распределенной системы онлайн-судьи Veritas

Артём Башкирев

07 Ноября 2023

Содержание

1	Введение	3
1.1	Мотивация	3
1.2	Цели и задачи	3
1.3	Целевая аудитория	4
2	Техническая архитектура	5
2.1	Обзор проектирования системы	5
2.2	Контейнеризация	6
2.2.1	Выделенный реестр контейнеров	6
3	Взаимосвязь архитектуры	8
3.1	Распределенный кэш	8
3.2	Балансировка нагрузки	8
3.3	Мониторинг	8
3.4	Обзор пути обработки решений	8
3.4.1	Выбор контейнера и распределенное выполнение	8
3.4.2	Выполнение тестов и возврат вердиктов	8
3.4.3	Агрегация и хранение вердиктов	8
3.4.4	Трейсинг в реальном времени	8
4	Оценка	9
4.1	Ожидаемые преимущества	9
4.2	План дальнейшего развития	9
5	Заключение	10
6	Приложение	11
6.1	Глоссарий	11
6.2	Подробные технические спецификации	11
6.3	Литература	11

1 Введение

1.1 Мотивация

Нынешние платформы для проведения соревнований по спортивному программированию, также используемые для вступительных экзаменов, могут уходить в простой в периоды высокой пользовательской нагрузки, что приводит к значительным прерываниям пользователей и потенциально влияет на достоверность оценок. Такие сбои могут привести к потере времени на соревнованиях, усилению беспокойства и упущенным возможностям для участников.

1.2 Цели и задачи

Цель 1: Разработать систему онлайн-судьи.

Задачи:

- Определить техническую архитектуру системы, включая конкретные технологии, которые будут использоваться.
- Разработать и внедрить различные компоненты системы.
- Провести интеграцию различных компонентов в целостную систему и протестировать в различных условиях.

Цель 2: Оценить производительность и масштабируемость системы.

Задачи:

- Разработать способы измерения производительности и масштабируемости.
- Провести эксперименты по сравнению производительности с существующими системами онлайн-судейства.
- Проанализировать результаты экспериментов и определить области для улучшения.

Цель 3: Развертывание в производственной среде.

Задачи:

- Выбрать облачного провайдера или службу хостинга для развертывания.
- Настроить систему для использования в производственной среде, включая меры безопасности и средства мониторинга.
- Выпустить Veritas для ограниченной группы пользователей и собрать отзывы.

1.3 Целевая аудитория

Пользователи

- Получают преимущества от бесперебойной работы и быстрого получения результатов.
- Организаторы получают повышенную надежность и поддержку большого количества участников.

Разработчики и администраторы

- Ценители технологий с открытым исходным кодом, модульной архитектуры и распределенных технологий.
- Используют хорошо документированный код и ресурсы для легкого развертывания и настройки.

2 Техническая архитектура

2.1 Обзор проектирования системы

Подход к микросервисам:

Veritas использует архитектуру микросервисов, в которой отдельные службы выполняют отдельные функции:

- **Служба судьи:** Получает материалы, проверяет их формат, выбирает подходящий контейнер и инициирует распределенное выполнение.
- **Служба прогонщиков:** Выполняет тестовые примеры в изолированных контейнерах и генерирует вердикты.
- **Агрегатор вердиктов:** Собирает вердикты от прогонщиков, выполняет дополнительный анализ и сохраняет окончательные результаты.
- **Сервис пользовательского интерфейса:** Взаимодействует с пользователями, обрабатывает аутентификацию, отображает статус отправки и предоставляет обратную связь.
- **Сервис управления данными:** Управляет доступом и взаимодействием с распределенными базами данных (Redis, PostgreSQL, MongoDB).

Каждый сервис работает независимо, взаимодействуя с другими через определенные API. Это позволяет:

- **Масштабируемость:** Отдельные сервисы можно масштабировать горизонтально, добавляя новые экземпляры в зависимости от спроса.
- **Управляемость:** Маленькие, сфокусированные сервисы легче разрабатывать, тестировать и обновлять.
- **Устойчивость к сбоям:** Сбои в работе сервисов изолированы, что минимизирует влияние на всю систему.

Механизмы связи: Veritas использует различные механизмы связи в зависимости от контекста:

- **RESTful API:** Используется для обмена структурированными данными между службами, например для получения представлений или хранения вердиктов.

- **Websockets:** Смогут обеспечить двунаправленную связь между клиентами и сервером в реальном времени, предоставляя оперативные обновления статуса подачи.
- **Redis Pub/Sub:** Обеспечивает асинхронную передачу вердиктов между сервисами: прогонщик публикует вердикт по завершении работы, а агрегатор вердиктов подписывается на его получение.
- **Очереди сообщений:** (Планируется) Для сценариев, требующих буферизации сообщений, будут реализованы с помощью RabbitMQ.

Эти механизмы обеспечивают эффективную и масштабируемую связь в распределенной системе.

Дополнительные соображения:

- **Обнаружение сервисов:** Такие механизмы, как Consul или Kubernetes, могут быть использованы для динамического определения использования сервисов.
- **Управление конфигурацией:** Такие инструменты, как Ansible, позволяют управлять конфигурацией распределенных сервисов.
- **Логгинг и мониторинг:** Распределенные сервисы протоколирования, такие как Atlassian Statuspage, обеспечивают централизованное понимание поведения системы и потенциальных проблем.

2.2 Контейнеризация

Система предполагает использование контейнеров. Это обеспечивает:

- **Согласованность:** Тесты и код выполняются на любой системе, независимо её расположения.
- **Изоляцию:** Потенциально деструктивный для системы прогонщик код предполагает невозможность выхода за пределы песочницы.
- **Скорость:** Предварительно созданные образы контейнеров исключают длительную сборку и направлены на быстрое получение результатов.

2.2.1 Выделенный реестр контейнеров

Представляемый Container Registry для Docker предполагает автоматическую сборку и обновления образов с инструментами CI/CD. Ключевым в представимом

решении станут системы контроля версий утилит сборки и языков программирования внутри контейнеров в целях обеспечить честность соревнований.

3 Взаимосвязь архитектуры

3.1 Распределенный кэш

3.2 Балансировка нагрузки

3.3 Мониторинг

3.4 Обзор пути обработки решений

3.4.1 Выбор контейнера и распределенное выполнение

3.4.2 Выполнение тестов и возврат вердиктов

3.4.3 Агрегация и хранение вердиктов

3.4.4 Трейсинг в реальном времени

4 Оценка

4.1 Ожидаемые преимущества

4.2 План дальнейшего развития

5 Заключение

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

6 Приложение

6.1 Глоссарий

6.2 Подробные технические спецификации

6.3 Литература