УДК 004.624

АНАЛИЗ ВЗАИМОДЕЙСТВИЯ РАСПРЕДЕЛЕННЫХ СЕРВИСОВ С ИСПОЛЬЗОВАНИЕМ ТЕЛЕМЕТРИИ

Фролов М.А. (Университет ИТМО)
Научный руководитель – доцент, PhD, Аксенов В.Е.
(Университет ИТМО)

Введение. При работе с распределенными системами большую сложность доставляет поиск проблем во взаимодействии их компонентов. Примерами таких проблем могут быть ошибки и отказы сервисов, их перегруженность и большая задержка ответов. Каждый такой сбой может нести серьезные убытки для компании, поэтому сейчас для их предотвращения и быстрого исправления требуется специальный отдел менеджеров по чрезвычайным ситуациям.

В компании Яндекс была создана система мониторинга *Juggler*, позволяющая быстро реагировать на проблемы, предупреждать сбои и получать уведомления при критическом изменении важных для данного сервиса метрик.

Для анализа логов и трассировок в России и за рубежом активно применяется сервис *Jaeger*, предоставляющий функционал работы с трассировками и логами: поиск и просмотр информации о цепочке вызовов функции, о времени, ошибках и параметрах вызова.

Однако, несмотря на использование этих решений, разработчики часто сталкиваются со сложностями в поиске проблем взаимодействия сервисов, в особенности при рассмотрении ошибок, происходящих в *межсервисном* взаимодействии. У разработчиков возникает запрос на динамический анализ работы системы, например рассмотреть ситуацию *до* и *во время* поломки.

Предлагаемое мною решение – новый подход к анализу взаимодействия различных сервисов, где они представляются в виде графа сервисов и операций с дополнительной статистикой по каждой операции, что помогает быстро находить сбои и наглядно наблюдать изменения во взаимодействии нескольких сервисов в разные промежутки времени.

Основная часть.

Построение графа взаимодействия сервисов состоит из трех основных этапов.

На первом этапе производится обработка и извлечения информации из трассировок и логов сервиса. Данные представляются в стандартном формате распределенного трейсинга и мониторинга OpenTelemetry, в котором каждый вызов функции это один спан с информацией о времени вызова, названии сервиса, информации о родительском спане, информации об успешности вызова операции. Из-за большого количества данных о трассировках они представлены в Protobuf формате, позволяющем эффективно их передавать и хранить.

Однако для того, чтобы воспользоваться данными нужны *инструменты для анализа больших данных, большой объем хранилища*, а также *методы для извлечения полезной информации и вывода ее в удобном для пользователей виде*. Данными возможностями обладает система Yandex Table (аналог Hadoop), которая позволяет эффективно обрабатывать большие объемы данных. В результате создается таблица с основной информацией об одной операции взаимодействия двух сервисов: названия сервисов, название проекта, время начала/конца операции, код ответа.

Вторым этапом является запуск задачи по подсчету статистики для данного сервиса в интервале времени. Для этого планировщик раз в пять минут запускает Map Reduce операцию, получая на выходе несколько таблиц с данными о каждой операции: количество, процент ошибок, статистику по видам ошибок, квантили для времени отклика 90/95/99, количество операций в секунду и другие. Данная таблица прекрасно подходит для подтверждения тех случаев, которые будут найдены на третьем этапе.

Третий этап – построение графического интерфейса. Представим наши данные в виде графа, где вершинами будут сервисы, а ребрами операции из статистики, посчитанной на

втором этапе. Наиболее удачным хранилищем для таких данных является графовая база данных Neo4j, которая не только позволяет эффективно хранить данные, но также осуществлять отображения данных в виде графа[2].
Графический интерфейс позволяет отображать 3 вида графов:

1) сервисы и операции для одной трассировки
2) взаимодействие сервисов по каждой операции
3) граф сервисов, где операции объединены в одно ребро

Есть классификация графов по тем данным, которые мы отображаем:

1) ошибки и их виды 4xx, 5xx.
2) квантили времени

Данное разделение позволяет структурировать тот большой объем данных, который содержится в трассировках, и представить его в читаемом виде. Так как данные хранятся в графовой базе данных, можно быстро и эффективно искать и преобразовывать данные, и у пользователей появляется возможность самим настраивать метрики для своего графа.
Основные варианты использования данного решения:

1) Построение схемы взаимодействия сервисов
2) Отслеживание проблем во взаимодействии сервисов
3) Автоматическое определение критического пути
4) Сравнительный анализ изменения взаимодействия двух сервисов в разные промежутки времени(в момент/до сбоя или до/после выхода новой версии приложения: сравнить на каких операциях процент ошибок и время стало больше)
5) Нахождение всех операций и сервисов с заданными ограничениями (например, много ошибок или большое время работы)
6) Сбор информации о взаимодействии сервисов и ее агрегация (время ответа, процент ошибок, количество байт, операций в секунду, квантили для времени отклика).

Выводы.

В данный момент система проходит апробацию. С ее помощью были найдены и проанализированы несколько проблем взаимодействия сервисов. Одним из примеров использования данного сервиса было нахождение большого количества ошибок во взаимодействии фронтового и бэкендового сервисов. Проблема заключалась в том, что фронтенд обращался к тем сервисам бэкенда, с которыми у него не должно было быть взаимодействия в данном пользовательском сценарии. В результате были заведены задачи в соответствующие команды, и проблема была устранена.

Список использованных источников:

1. Формат данных Opentelemetry // Opentelemetry URL: https://opentelemetry.io/ (дата обращения: 05.02.2023).
2. Графовая база данных Neo4j // Neo4j URL: https://neo4j.com/ (дата обращения: 01.01.2022).

Фролов И.И. (автор) Подпись

Аксенов В.Е. (научный руководитель) Подпись