DOI: 10.58168/MoInSyTe2024_61-67
УДК 004.9

ТРЕБОВАНИЯ К БЭКЕНДУ, ПРЕДЪЯВЛЯЕМЫЕ В ОБЛАЧНЫХ СЕРВИСАХ, ОПЕРИРУЮЩИХ БОЛЬШИМИ ДАННЫМИ

Е.А. Злобин[1], Е.В. Чернышова[1], Т.В. Скворцова[1]

[1]ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация. В статье рассматриваются основные требования, предъявляемые к бэкенд-составляющей облачных сервисов, работающих с большими объемами данных. Проанализированы ключевые аспекты, такие как масштабируемость, отказоустойчивость, высокая производительность и безопасность. Выделены основные подходы и технологии, позволяющие удовлетворить данные требования, включая использование распределенных систем хранения и обработки данных, контейнеризацию и оркестрацию микросервисов, а также применение современных методов обеспечения информационной безопасности. Приведены примеры практического использования указанных подходов в реальных проектах. На основе проведенного анализа сформулированы рекомендации по проектированию и реализации бэкенд-систем для облачных big data сервисов.

Ключевые слова: облачные вычисления, большие данные, бэкенд, масштабируемость, отказоустойчивость, производительность, безопасность.

REQUIREMENTS FOR THE BACKEND IN CLOUD SERVICES DEALING WITH BIG DATA

E.A. Zlobin[1], E.V. Chernyshova[1], T.V. Skvortsova[1]

[1]Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract. The article discusses the main requirements for the backend component of cloud services that deal with large volumes of data. The key aspects such as scalability, fault tolerance, high performance, and security are analyzed. The main approaches and technologies that allow satisfying these requirements are highlighted, including the use of distributed data storage and processing systems, containerization and orchestration of microservices, as well as the application of modern methods of ensuring information security. Examples of practical use of these approaches in real projects are provided. Based on the analysis, recommendations for designing and implementing backend systems for cloud big data services are formulated.

Keywords: cloud computing, big data, backend, scalability, fault tolerance, performance, security.

ВВЕДЕНИЕ

Облачные сервисы, оперирующие большими данными, становятся все более востребованными в различных сферах, таких как электронная коммерция интернет вещей, социальные сети, здравоохранение и многие другие. Эффективная работа подобных сервисов напрямую зависит от возможностей их бэкенд-составляющей, отвечающей за хранение, обработку и предоставление данных. В связи с этим, к бэкенду облачных big data сервисов предъявляются повышенные требования в отношении масштабируемости, отказоустойчивости, производительности и безопасности [1].

В данной статье мы подробно рассмотрим ключевые требования, предъявляемые к бэкенду облачных сервисов, работающих с большими данными, а также проанализируем основные подходы и технологии, позволяющие удовлетворить эти требования. Мы разберем такие аспекты, как обеспечение горизонтальной масштабируемости, отказоустойчивости, высокой производительности и безопасности, а также приведем примеры практического применения соответствующих решений в реальных проектах.

МАСШТАБИРУЕМОСТЬ

Одной из ключевых особенностей облачных систем, работающих с большими данными, является необходимость обеспечения горизонтальной масштабируемости. Это означает возможность линейного увеличения производительности системы путем добавления новых вычислительных узлов без внесения изменений в архитектуру и программный код.

Для реализации данного требования применяются различные подходы, такие как использование распределенных файловых систем (HDFS, Ceph), NoSQL баз данных (Cassandra, MongoDB) и фреймворков распределенной обработки (Hadoop, Spark) [2].

Распределенные файловые системы, такие как HDFS (Hadoop Distributed File System) и Ceph, позволяют хранить огромные объемы данных на множестве узлов кластера, обеспечивая при этом высокую доступность и отказоустойчивость. Данные автоматически реплицируются между узлами, что позволяет продолжать работу даже в случае выхода из строя отдельных серверов.

NoSQL базы данных, такие как Cassandra и MongoDB, предназначены для эффективной работы с неструктурированными и полуструктурированными данными в распределенной среде. Они обеспечивают автоматическое горизонталь-

ное масштабирование путем добавления новых узлов в кластер, а также поддерживают репликацию и распределение данных для повышения доступности и производительности.

Фреймворки распределенной обработки, такие как Hadoop и Spark, позволяют выполнять параллельные вычисления на большом количестве узлов кластера. Они автоматически распределяют задачи между узлами и обеспечивают отказоустойчивость путем перезапуска неудачных задач на других узлах.

Применение указанных технологий и подходов позволяет строить горизонтально масштабируемые бэкенд-системы для облачных big data сервисов, способные обрабатывать петабайты данных и обслуживать миллионы пользователей.

ОТКАЗОУСТОЙЧИВОСТЬ

Другим важным аспектом является обеспечение отказоустойчивости бэкенд-системы. Учитывая большое количество узлов и компонентов, из которых состоит типичная облачная инфраструктура для работы с большими данными, вероятность сбоев и отказов отдельных элементов достаточно высока. В связи с этим, архитектура бэкенда должна предусматривать механизмы автоматического обнаружения сбоев, переключения на резервные компоненты и восстановления после отказов.

Для решения этой задачи применяются такие подходы как репликация данных, использование распределенных очередей сообщений (Kafka, RabbitMQ) и оркестрация контейнеров (Kubernetes) [3].

Репликация данных является ключевым механизмом обеспечения отказоустойчивости в распределенных системах хранения. Данные автоматически копируются на несколько узлов кластера, что позволяет продолжать работу даже в случае выхода из строя части серверов. Современные распределенные файловые системы и NoSQL базы данных поддерживают настройку коэффициента репликации, что позволяет найти оптимальный баланс между надежностью и накладными расходами на хранение.

Распределенные очереди сообщений, такие как Kafka и RabbitMQ, обеспечивают надежную доставку данных между компонентами распределенной системы. Они позволяют буферизовать сообщения и автоматически повторять доставку в случае сбоев, что повышает устойчивость системы к отказам отдельных компонентов.

Оркестрация контейнеров с помощью систем управления кластером, таких как Kubernetes, позволяет автоматизировать развертывание, масштабирование и

восстановление после сбоев для микросервисных архитектур. Kubernetes обеспечивает автоматический перезапуск контейнеров в случае их отказа, а также позволяет настраивать политики восстановления, такие как перенос контейнеров на другие узлы кластера.

Использование указанных подходов и технологий позволяет создавать отказоустойчивые бэкенд-системы для облачных big data сервисов, способные продолжать работу даже в случае сбоев и отказов отдельных компонентов.

ПРОИЗВОДИТЕЛЬНОСТЬ

Высокая производительность является одним из ключевых требований для облачных big data сервисов, поскольку от скорости обработки и выдачи результатов напрямую зависит качество пользовательского опыта и конкурентоспособность сервиса в целом.

Для обеспечения высокой производительности бэкенда применяются различные оптимизационные техники, такие как использование индексов и денормализация данных, кэширование часто запрашиваемой информации (Redis, Memcached), применение асинхронных и неблокирующих подходов в разработке (Node.js, Akka). Также большое значение имеет возможность эффективного масштабирования бэкенд-составляющей при росте нагрузки [4].

Использование индексов является одним из основных способов оптимизации производительности при работе с большими объемами данных. Индексы позволяют быстро находить нужные записи по заданным критериям, избегая полного сканирования всего набора данных. Современные NoSQL базы данных, такие как Cassandra и MongoDB, поддерживают гибкие возможности индексации, позволяющие оптимизировать выполнение запросов под конкретные паттерны доступа.

Денормализация данных, то есть хранение избыточной информации для ускорения часто выполняемых запросов, является еще одним эффективным подходом к оптимизации производительности. Вместо выполнения сложных соединений таблиц или агрегаций в момент запроса, данные заранее подготавливаются в денормализованном виде, что позволяет получать результаты за константное время.

Кэширование часто запрашиваемых данных позволяет разгрузить backend-системы и повысить скорость обработки запросов. Такие решения, как Redis и Memcached, предоставляют высокопроизводительное хранилище ключ-значение

в оперативной памяти, что позволяет на порядки ускорить доступ к часто используемым данным по сравнению с обращением к дисковым хранилищам.

Применение асинхронных и неблокирующих подходов в разработке бэкенда позволяет эффективно обрабатывать большое количество одновременных запросов. Технологии, такие как Node.js и Akka, основаны на событийно-ориентированной модели и неблокирующем вводе-выводе, что позволяет обслуживать тысячи клиентов с помощью небольшого количества потоков выполнения.

Наконец, возможность эффективного масштабирования бэкенд-системы играет ключевую роль в обеспечении высокой производительности при росте нагрузки. Использование распределенных архитектур, контейнеризации и оркестрации позволяет быстро наращивать вычислительные мощности путем добавления новых узлов в кластер без необходимости изменения кода приложений.

БЕЗОПАСНОСТЬ

Обеспечение безопасности является критически важным аспектом для облачных сервисов, работающих с чувствительными пользовательскими данными, такими как персональная информация, платежные данные, коммерческая тайна и т.д.

Для защиты данных и предотвращения несанкционированного доступа в бэкенде облачных систем применяются различные методы, включая шифрование данных при передаче и хранении, использование токенов и протоколов авторизации (OAuth, JWT), регулярное обновление программного обеспечения и мониторинг безопасности. Также важным является соответствие бэкенд-системы отраслевым стандартам безопасности, таким как PCI DSS для платежных сервисов или HIPAA для систем здравоохранения [5].

Шифрование данных при передаче и хранении является фундаментальным требованием для защиты конфиденциальности информации. Для шифрования данных при передаче по сети используются протоколы SSL/TLS, которые обеспечивают безопасное соединение между клиентом и сервером. При хранении данных применяются алгоритмы симметричного и асимметричного шифрования, такие как AES и RSA, для защиты от несанкционированного доступа в случае компрометации системы хранения.

Использование токенов и протоколов авторизации, таких как OAuth и JWT (JSON Web Tokens), позволяет реализовать надежную аутентификацию и контроль доступа в распределенных системах. Вместо передачи учетных данных пользователя при каждом запросе, используются временные токены, которые

выдаются после успешной аутентификации и содержат информацию о правах доступа. Это снижает риски, связанные с передачей и хранением конфиденциальных учетных данных.

ЗАКЛЮЧЕНИЕ

При работе с облачными сервисами, обрабатывающими большие объемы данных, я считаю, крайне важно уделять особое внимание разработке и поддержке их базовой инфраструктуры. Необходимо обеспечить масштабируемость, надежность, высокую производительность и безопасность. Для этого применяются различные подходы и технологии, такие как распределенные системы хранения и обработки данных, контейнеризация и управление микросервисами, оптимизация производительности с использованием кэширования и асинхронной обработки, а также современные методы обеспечения информационной безопасности. Учитывая эти требования и применяя соответствующие решения, можно создать эффективные и надежные бэкенд-системы для облачных сервисов обработки больших данных.

Список литературы

1. Аксютина, Е.М. Использование облачных технологий для обработки больших данных / Е.М. Аксютина, Ю.С. Белов // Московский экономический журнал. – 2020. – №6. – URL: https://elibrary.ru/item.asp?edn=wpncrz (дата обращения: 20.03.2024).

2. Клеппман М. Высоконагруженные приложения. Программирование, масштабирование, поддержка. – СПб.: Питер, 2018. – 640 с.

3. Таненбаум, Э. Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. ван Стеен ; – СПб.: Питер, 2003. – 877 с.

4. Аудит информационной безопасности // Астрал Безопасность. – URL: https://is.astral.ru/services/zashchita-informatsii/audit-informatsionnoy-bezopasnosti/ (дата обращения: 20.03.2024).

5. Киреева, К.А Разработка искусственной нейронной сети для классификации ЭКГ / К.А. Киреева, Л.А. Коробова, Д.В. Арапов // Моделирование систем и процессов. – 2023. – Т. 16, №3. – С. 42-54. – DOI: 10.12737/2219-0767-2023-16-3-42-54.

6. Классификация последствий воздействия ИИ КП на РЭА / А.Е. Козюков, П.А. Чубунов, К.В. Зольников [и др.] // Моделирование систем и процессов. – 2021. – Т. 14, № 3. – С. 22-28. – DOI: 10.12737/2219-0767-2021-14-3-22-28.

7. Полуэктов А.В., Макаренко Ф.В., Ягодкин А.С. Использование сторонних библиотек при написании программ для обработки статистических данных // Моделирование систем и процессов. – 2022. – Т. 15, № 2. – С. 33-41.

8. Зольников, В.К. Моделирование и анализ производительности алгоритмов балансировки нагрузки облачных вычислений / В.К. Зольников, О.В. Оксюта, Н.Ф. Даюб // Моделирование систем и процессов. – 2020. – Т. 13, №1. – С. 32-39.

References

1. Aksyutina, E.M. The use of cloud technologies for big data processing / E.M. Aksyutina, Y.S. Belov // Moscow Economic Journal. – 2020. – No.6. – URL: https://elibrary.ru/item.asp?edn=wpncrz (date of application: 03/20/2024).

2. Clement M. Highly loaded applications. Programming, scaling, support. – St. Petersburg, 2018. – 640 p.

3. Tanenbaum, E. Distributed systems. Principles and paradigms / E. Tanenbaum, M. van Steen. – St. Petersburg: Peter, 2003. – 877 p.

4. Information security audit // Astral Security. – URL: https://is.astral.ru/services/zashchita-informatsii/audit-informatsionnoy-bezopasnosti/. (date of application: 03/20/2024).

5. Kireeva, K.A. Development of an artificial neural network for ECG classification / K.A. Kireeva, L.A. Korobova, D.V. Arapov // Modeling of systems and processes. - 2023. – Vol. 16, No.3. – pp. 42-54. – DOI: 10.12737/2219-0767-2023-16-3-42-54.

6. Classification of the effects of AI KP on REA / A.E. Kozyukov, P.A. Chugunov, K.V. Zolnikov [et al.] // Modeling of systems and processes. - 2021. – Vol. 14, No. 3. – pp. 22-28. – DOI: 10.12737/2219-0767-2021-14-3-22-28.

7. Poluektov A.V., Makarenko F.V., Yagodkin A.S. The use of third-party libraries when writing programs for processing statistical data // Modeling of systems and processes. - 2022. – vol. 15, No. 2. – pp. 33-41.

8. Zolnikov, V.K. Modeling and performance analysis of cloud computing load balancing algorithms / V.K. Zolnikov, O.V. Oxyuta, N.F. Dayub // Modeling of systems and processes. - 2020. – vol. 13, No. 1. – pp. 32-39.