

Применение функций NATS Core в распределённых системах

Бабашов Даниэль Эльманович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассматривается набор функций NATS Core системы брокера сообщений NATS для использования в микросервисной системе. Для демонстрации работы методов используются NATS сервер и набор инструментов NATS cli на стороне клиента. В результате исследования будут продемонстрированы преимущества инструмента NATS Core добавленного как посредника сообщений в микросервисной архитектуре.

Ключевые слова: NATS, брокер сообщений, балансировщик нагрузки, распределённая система.

Application of NATS Core functions in distributed systems

Babashov Daniel Elmanovich

Sholom-Aleichem Priamursky State University

Student

Abstract

This article discusses the NATS Core feature set of the NATS message broker system for use in a microservice system. To demonstrate how the methods work, a NATS server and a set of NATS cli tools for the client are used. As a result of the research, the advantages of the NATS Core tool added as a message mediator in the microservice architecture will be demonstrated.

Keywords: NATS, message broker, load balancer, distributed system.

1 Введение

1.1 Актуальность

Современные приложения, работающие в микросервисной архитектуре или облачных средах, нуждаются в эффективных механизмах коммуникации между компонентами. NATS Core предоставляет высокопроизводительное, минималистичное и отказоустойчивое решение, способное удовлетворить потребности масштабируемости, надежности и производительности распределенных систем. Особенности NATS Core заключаются в его простой архитектуре, ориентированной на паттерн публикации и подписки, а также на запросы и ответы. Такая архитектура позволяет легко интегрировать различные компоненты системы, сохраняя при этом низкую сложность настройки и администрирования. Протокол NATS, используемый для взаимодействия, оптимизирован для высокой скорости передачи данных, что

делает его особенно подходящим для приложений, требующих низкой латентности, таких как системы реального времени, IoT-решения и торговые платформы.

1.2 Обзор исследований

Д.А.Яблонский в своей статье написал обзор современных систем обмена сообщениями - Kafka, RabbitMQ и NATS streaming [1]. Н.А.Иванова, С.В.Антоненко, А.М.Сотченков провели исследования брокера сетевых сообщений для организации взаимодействия компонентов распределённых систем [2]. Е.А.Тришин исследовала функции и возможности различных брокеров сообщений в приложениях с микросервисной архитектурой [3]. В.Д.Лукин, М.В.Верховский, Н.И.Барсуков провели сравнительный анализ брокеров сообщений, описали преимущества использования каждого в отдельных случаях [4]. Р.В.Воячек в своей работе описал брокер сообщений, как способ коммуникации между элементами информационной системы [5].

1.3 Цель исследования

Цель исследования – продемонстрировать функции NATS Core помогающие упростить реализацию распределённых систем.

2 Материалы и методы

Для реализации используются образы Docker. Контейнеры на основе образов nats и natsio/nats-box применяются в качестве сервера и клиента соответственно.

3 Результаты и обсуждения

NATS Core является базовым функционалом в системе NATS. Работа осуществляется по модели «издатель-подписчик», используя адресацию на основе субъекта/темы. Обусловлен независимостью от местоположения и представляет шаблон связи «многие ко многим». Позволяет создавать решения для разработки микросервисов, без необходимости использования дополнительных технологий, таких как балансировщики нагрузки, шлюзы API или конфигурация DNS.

В NATS используется текстовый протокол, основанный на простых командах. Клиенты и сервер обмениваются командами и сообщениями. Клиенты взаимодействуют с сервером NATS через постоянное TCP соединение. Используется один порт (обычно 4222 по умолчанию) для обмена данными. Соединение поддерживается живым с помощью сообщений типа "PING/PONG".

Основные команды:

- CONNECT – клиент инициирует соединение, передавая параметры, такие как авторизация, версия клиента и другие;
- PING/PONG – используются для проверки доступности соединения между клиентом и сервером;

- SUB – клиент подписывается на определённую тему;
- UNSUB – клиент отписывается от темы;
- PUB – клиент отправляет сообщение на определённую тему;
- MSG – сервер отправляет сообщение клиенту.

В NATS сообщения маршрутизируются по текстовым темам. Они поддерживают иерархии с использованием разделителей (.) и wildcard-символов. Символы (>) и (*) означают подписку на все темы в поддереве и подписку на один уровень иерархии соответственно (рис.1).

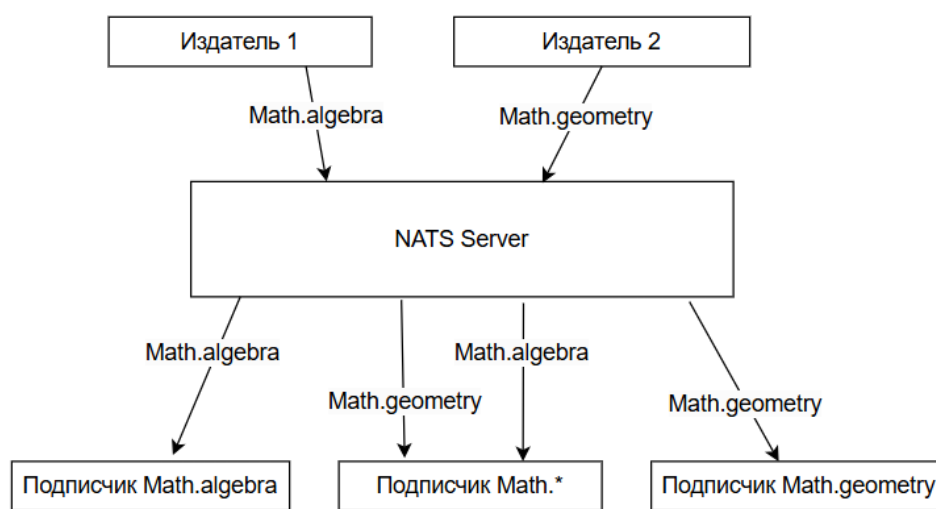


Рис. 1. Схема взаимодействия издателей и подписчиков через NATS

Используя Docker контейнеры можно быстро загрузить и установить клиента и сервера NATS. Для взаимодействия используется адресация по имени сервиса, но для наружного клиента можно сопоставить порты. Сервер NATS открывает 3 порта для взаимодействия. Основной клиентский порт 4222. Этот порт используется клиентами для подключения к серверу. Через него проходят команды CONNECT, PUB, SUB и другие. Порт маршрутизации (Cluster Port) 6222 используется для обмена данными между серверами в кластере. Серверы делятся информацией о подписках и маршрутах для доставки сообщений между узлами. Порт для мониторинга (HTTP/Management) 8222 предоставляет интерфейс для наблюдения за состоянием сервера. Через этот порт можно получить данные в формате JSON или просмотреть состояние через браузер, например, по ссылке (<http://localhost:8222/>) (рис.2).

```

version: "3.8"

services:
  # NATS сервер
  service-nats:
    container_name: service-nats
    image: nats:latest
    #ports:
    # - "4222:4222" # Порт для клиентов
    # - "8222:8222" # HTTP порт для управления и отчетов
    # - "6222:6222" # Порт для маршрутизации в кластерах

  # NATS клиент
  nats-client:
    container_name: nats-client
    image: natsio/nats-box
    depends_on:
      - service-nats
    environment:
      - NATS_URL=nats://service-nats:4222
    entrypoint: ["/bin/sh", "-c"]
    command: ["while true; do sleep 30; done;"] # Контейнер остается активным

```

Рис. 2. Инструкция docker-compose для запуска клиента и сервера NATS

Чтобы упростить ввод команд, в переменные окружения для клиента указан адрес сервера. На рисунке 3 приведены результаты взаимодействия двух клиентов в разных терминалах. Подписчик подписался на тему msg.test, после чего он находится в режиме прослушивания. Когда другой клиент отправит сообщение с указанием той же темы, подписчик получит сообщение.

```

~ # nats sub msg.test
03:31:16 Subscribing on msg.test

~ # nats pub msg.test "NATS MESSAGE"
03:31:41 Published 12 bytes to "msg.test"

~ # nats sub msg.test
03:31:16 Subscribing on msg.test
[#1] Received on "msg.test"
NATS MESSAGE

```

Рис. 3. Получение подписчиком сообщения от издателя

NATS поддерживает шаблон «Запрос-Ответ». Запрос публикуется по заданной теме с использованием темы ответа. Подписчики слушают тему и динамически направляют ответ обратно запрашивающей стороне, независимо от местонахождения любой из сторон. Для этого подписчик должен использовать команду reply название темы и сообщение. Отправитель указывает request название темы и сообщение. После получения сообщения подписчик автоматически вернёт свой ответ (рис.4).

```

~ # nats reply help 'first answer'
03:37:24 Listening on "help" in group "NATS-RPLY-22"
03:38:52 [#0] Received on subject "help":

I need help!

~ # nats request help 'I need help!'
03:38:51 Sending request on "help"
03:38:52 Received with rtt 1.751422ms
first answer

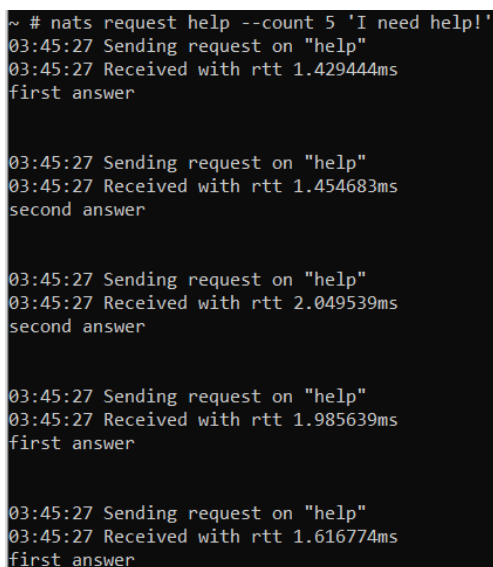
```

Рис. 4. Пример использования шаблона Запрос-Ответ

В NATS для распределения нагрузки между подписчиками используется концепция Queue Groups (группы очередей). В этой системе несколько подписчиков могут быть частью одной группы очередей, и NATS

будет отправлять сообщения в очередь одному подписчику из этой группы, распределяя нагрузку среди них. Несколько подписчиков могут подписаться на одну и ту же тему (subject) и стать частью одной группы очередей. Подписчики, которые являются частью группы очередей, обрабатывают сообщения поочередно и только один из них получит сообщение, даже если несколько подписчиков подписаны на ту же тему. Если один из подписчиков выходит из группы или не может обработать сообщение, NATS перераспределит сообщения среди оставшихся активных подписчиков. Это позволяет эффективно использовать несколько подписчиков для обработки сообщений, не перегружая одного подписчика.

После создания второго подписчика, отправляющего ответ second answer, результат с равной вероятностью вернёт один из двух слушателей. Для проверки распределения можно в запросе указать флаг (--count) с числом запросов. В примере на 5 обращений произошло распределение для первого слушателя 3 для второго 2 обращения (рис.5).



```
~ # nats request help --count 5 'I need help!'
03:45:27 Sending request on "help"
03:45:27 Received with rtt 1.429444ms
first answer

03:45:27 Sending request on "help"
03:45:27 Received with rtt 1.454683ms
second answer

03:45:27 Sending request on "help"
03:45:27 Received with rtt 2.049539ms
second answer

03:45:27 Sending request on "help"
03:45:27 Received with rtt 1.985639ms
first answer

03:45:27 Sending request on "help"
03:45:27 Received with rtt 1.616774ms
first answer
```

Рис. 5. Подключаемые модули

Такой метод позволяет распределить количество обращений с высоконагруженного сервиса, разделив его на несколько частей. В таком случае NATS будет выступать в роли балансировки нагрузки.

В результате исследования были рассмотрены возможности функционала NATS Core и представлены полезные возможности во включении данного инструмента в микросервисную архитектуру в качестве брокера сообщений. Приведены примеры использования и описана сетевая составляющая на уровне протоколов.

Библиографический список

1. Яблонский Д.А. Обзор современных систем обмена сообщениями - Kafka, RabbitMQ и NATS streaming // В сборнике: Наука и инновации -

- современные концепции. Сборник научных статей по итогам работы Международного научного форума. Москва, 2021. С. 161-168.
2. Иванова Н.А., Антоненко С.В., Сотченков А.М. Использование брокера сетевых сообщений для организации взаимодействия компонентов распределённых систем // В сборнике: Наука и образование: актуальные вопросы теории и практики. Материалы IV Международной научно-методической конференции. Самара-Оренбург, 2024. С. 281-286.
 3. Тришин Е.А. Исследование брокеров сообщений в приложениях с микро-сервисной архитектурой // Вестник науки. 2024. Т. 3. № 6 (75). С. 1469-1474.
 4. Лукин В.Д., Верховский М.В., Барсуков Н.И. Сравнительный анализ брокеров сообщений // Труды международного симпозиума "Надежность и качество". 2024. Т. 2. С. 390-392.
 5. Воячек Р.В. Брокер сообщений, как способ коммуникации между элементами информационной системы // Вестник научных конференций. 2024. № 5-2 (105). С. 34-35.
 6. Гулов Г.В. Безопасность в брокерах сообщений и как обеспечить надежную передачу данных // Научный аспект. 2024. Т. 26. № 7. С. 3251-3255.
 7. Лукин В.Д., Верховский М.В., Барсуков Н.И. Определение сферы применения брокеров сообщений // В сборнике: Математическое и компьютерное моделирование естественно-научных и социальных проблем (МКМ-2024). Сборник статей по материалам XVIII Всероссийской с международным участием научно-технической конференции молодых специалистов, аспирантов и студентов. Пенза, 2024. С. 280-285.