Выпускная квалификационная работа

Инфраструктура распределённой трассировки для ClickHouse

Выполнил студент группы 166, 4 курса,

Кожихов Александр Олегович

Руководитель ВКР:

Руководитель группы разработки ClickHouse,

Миловидов Алексей Николаевич

Введение

- ClickHouse - столбцовая СУБД, применяемая преимущественно для интерактивной аналитической обработки данных (OLAP).

Введение

- ClickHouse - столбцовая СУБД, применяемая преимущественно для интерактивной аналитической обработки данных (OLAP).
- Распределенная трассировка запросов
- Распределенный запрос

Введение

- ClickHouse - столбцовая СУБД, применяемая преимущественно для интерактивной аналитической обработки данных (OLAP).
- Распределенная трассировка запросов
- Распределенный запрос
- OpenTracing API - общепринятый интерфейс для распределенной трассировки.
- Jaeger Tracing - система для агрегации и визуализации данных о распределенной трассировке.

Distributed request tracing

- Уникальный идентификатор trace id на каждый запрос.
- Уникальный идентификатор span id на каждую логическую единицу запроса.
- Иерархия родитель-потомок у подзапросов.

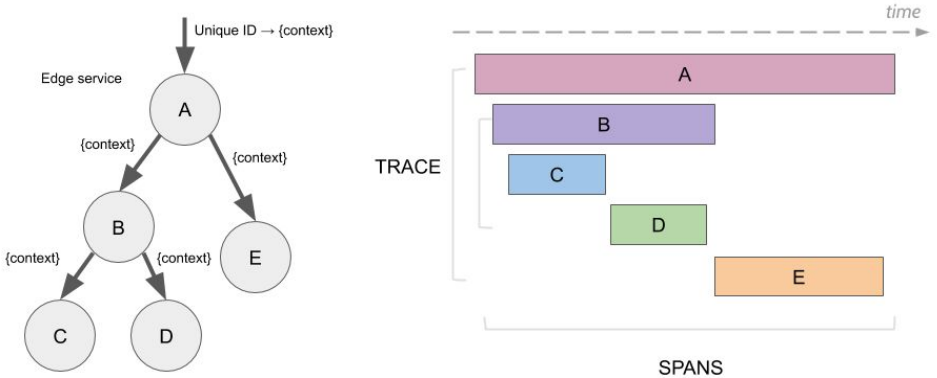
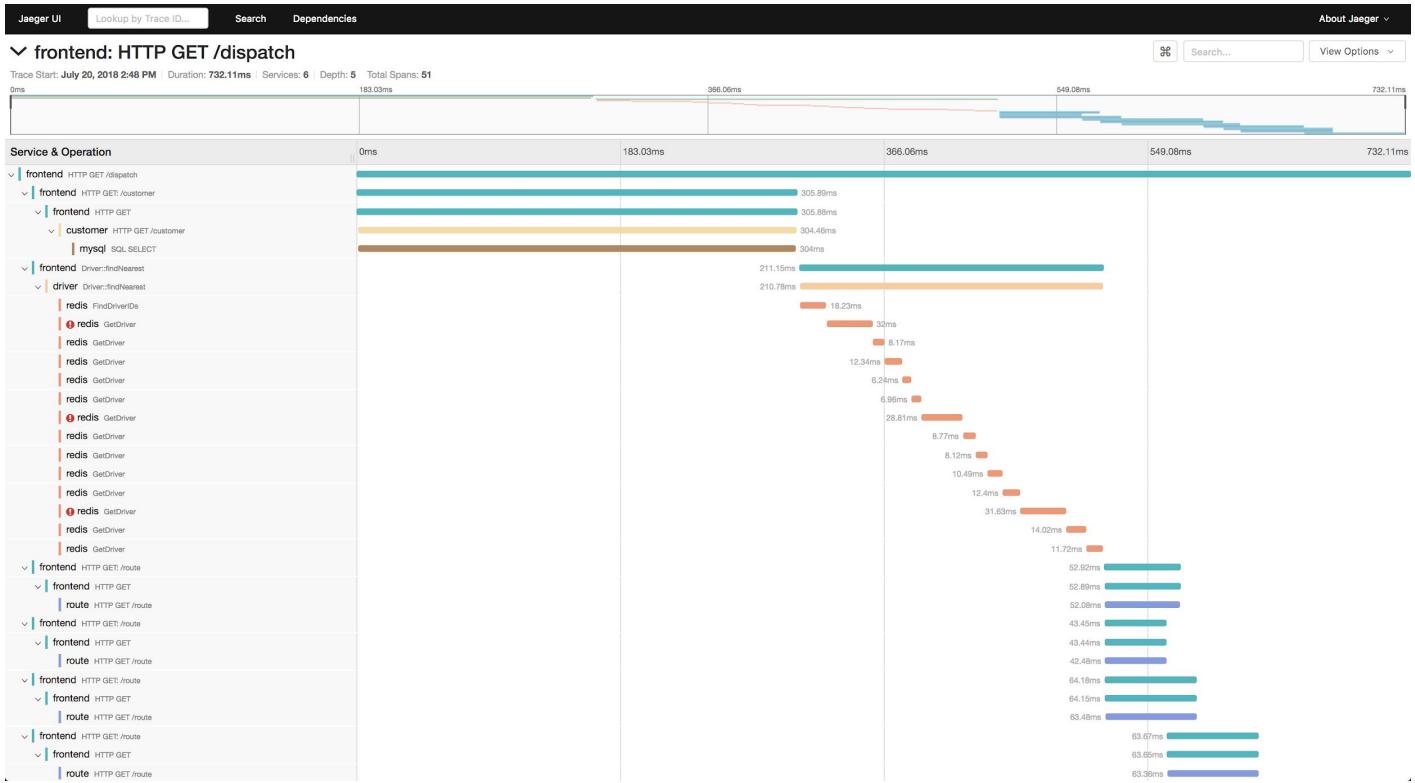
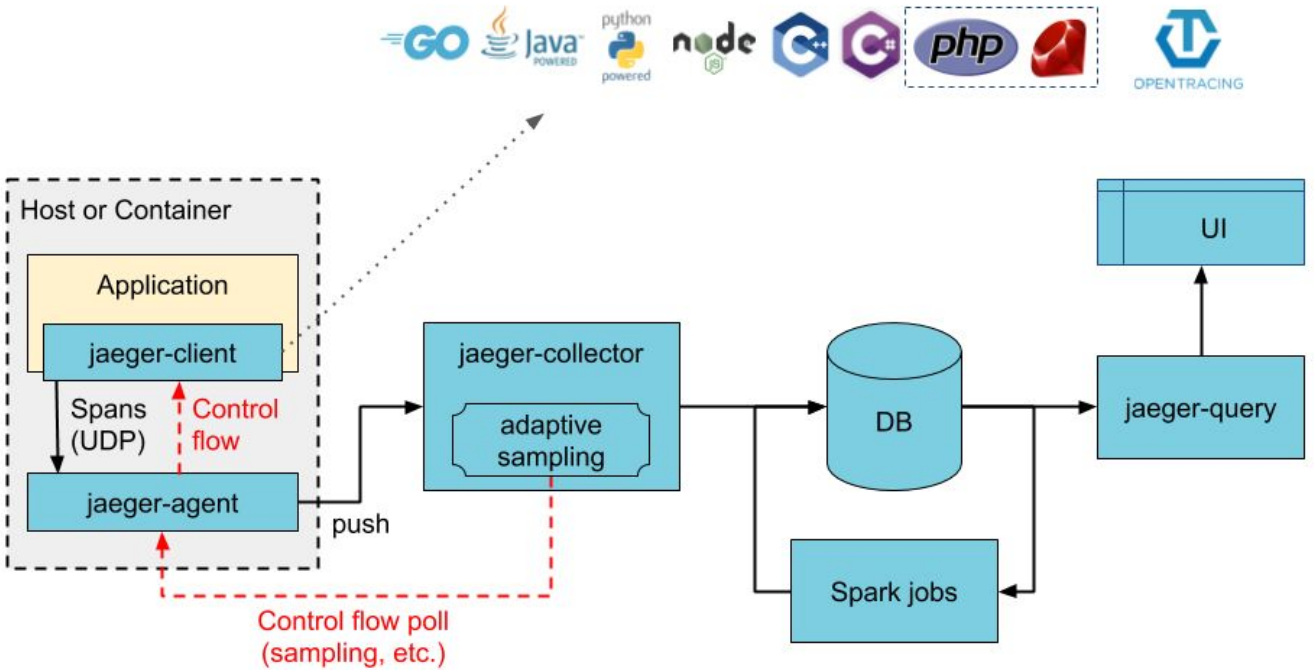


Рис. 1. Представление распределенной трассировки

Реальная визуализация в Jaeger



Архитектура Jaeger



Цель работы

Получить в интерфейсе Jaeger подобный трейс распределенного запроса к ClickHouse, который бы обладал достаточной выразительностью, чтобы можно было увидеть выполнение логических компонент запроса.

Задачи работы

1. Исследовать существующие подходы.
2. Реализовать механизм для передачи данных о распределенном запросе по сетевому протоколу.
3. Поддержать трассировку для запросов, выполняющихся многопоточно.
4. Добавить возможность передавать в ClickHouse внешние трейсы.
5. Детализировать трассировку, используя CurrentMetrics.
6. Научиться передавать собранные данные в систему Jaeger.
7. Детализировать трассировку, позволив отражать логические компоненты выполнения запроса, т.н. процессоры.

Передача данных о трейсе по сети

- Абсолютный идентификатор спана - SpanContext - по существу пара (trace_id, span_id)
- На основе структуры ClientInfo в TCPHandler.cpp и HTTPHandler.cpp

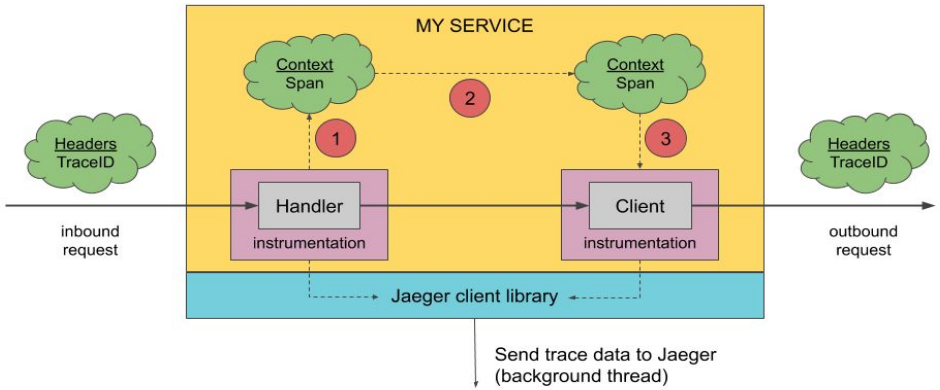


Рис.3. Передача данных о трейсе по сети

Работа с трейсом в многопоточном окружении

- Thread local переменная типа ThreadStatus хранит контекст запроса.
- Этот контекст общий для всех потоков, исполняющих запрос.
- Корневой спан подзапроса хранится в этом контексте.
- Другие спаны хранятся непосредственно в ThreadStatus.

Трейс, начавшийся вне ClickHouse

HTTP-заголовок “Traceparent”

- 128 бит на trace_id
- 64 бита на span_id

CurrentMetrics для детализации трассировки

- Переиспользуем CurrentMetrics которые соответствуют, например, чтению с диска или записи на диск.
- В коде реализовано в виде guard-объектов.
- На время жизни guard-а перекладываем в thread-local переменную новый, дочерний, спан.
- Таким образом получаем неявный стек спанов.

Поддержка Jaeger

- Client to Agent: Apache Thrift over UDP
- Agent to Collector: gRPC
- Client to Collector: Apache Thrift over HTTP

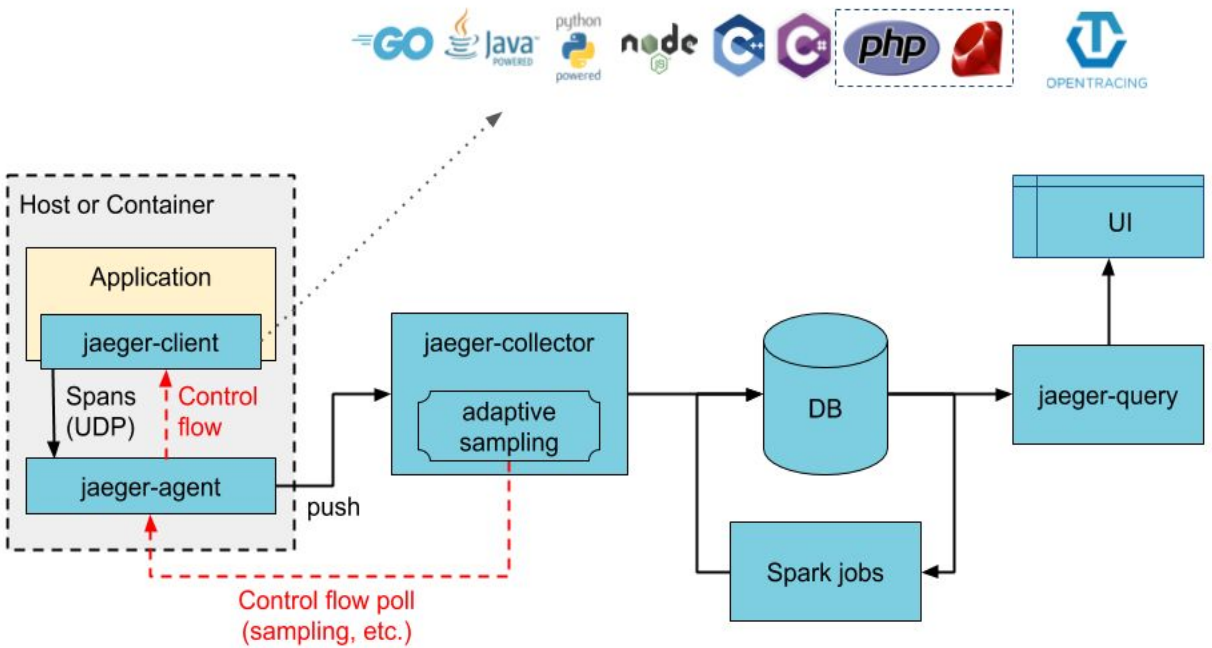


Рис.4. Архитектура системы Jaeger

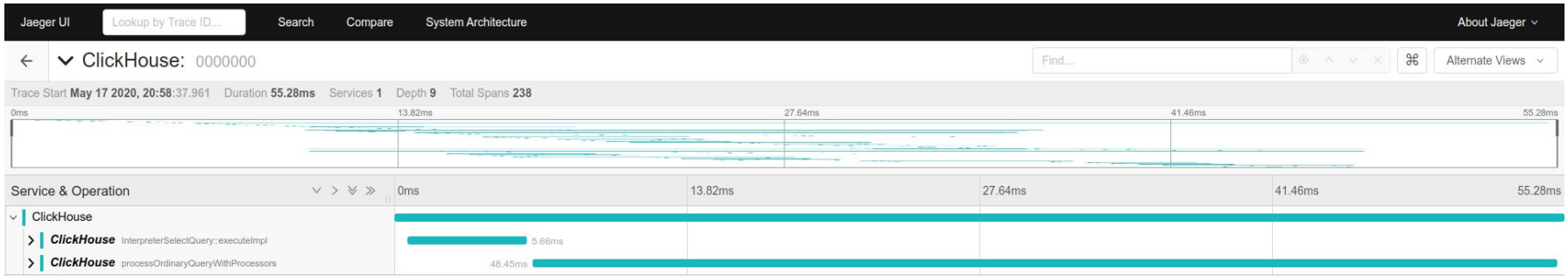
Processors для детализации трассировки

- Интерпретатор запроса (например InterpreterSelectQuery.cpp) анализирует abstract syntax tree запроса.
- Так называемые процессоры - логически обособленные компоненты запроса.
- Зависимости процессоров друг от друга описываются в коде некоторым графом.
- Шедулинг процессоров - в файле PipelineExecutor.cpp

Processors для детализации трассировки

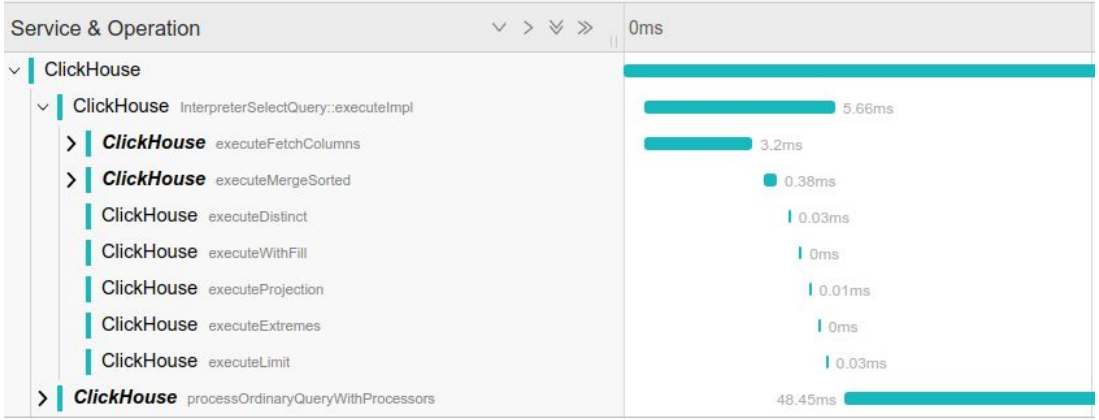
- Необходимо следить за состояниями процессора (Idle, Finished).
- У исполняющих потоков тоже есть свои спаны. Паркуем их, когда начинаем исполнять процессор.
- Внутри спанов процессоров есть спаны соотв. этапам работы процессора - work, prepare.

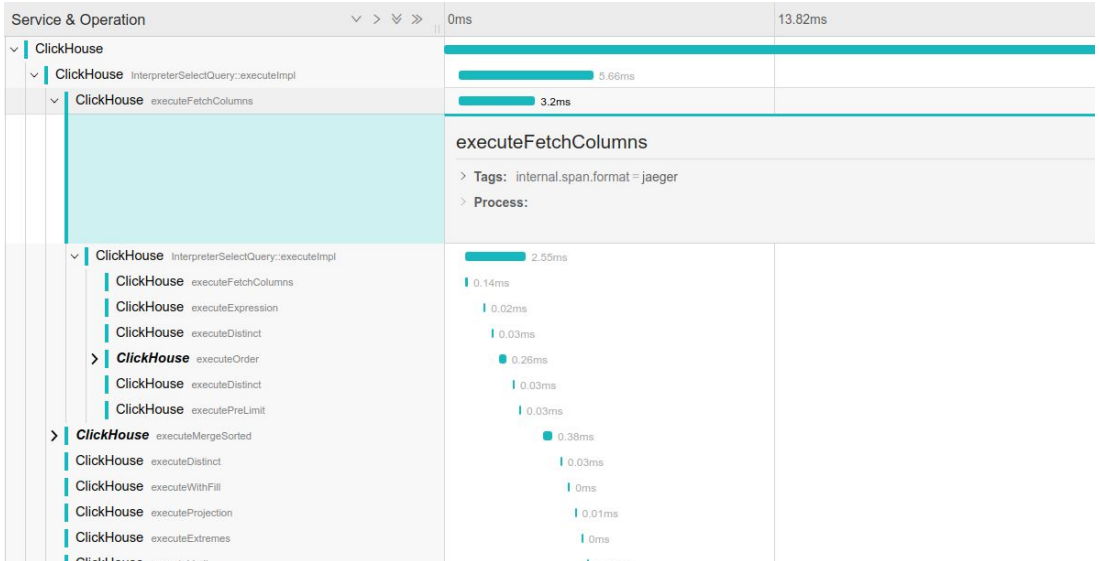
Пример работы на основе распределенного SELECT запроса



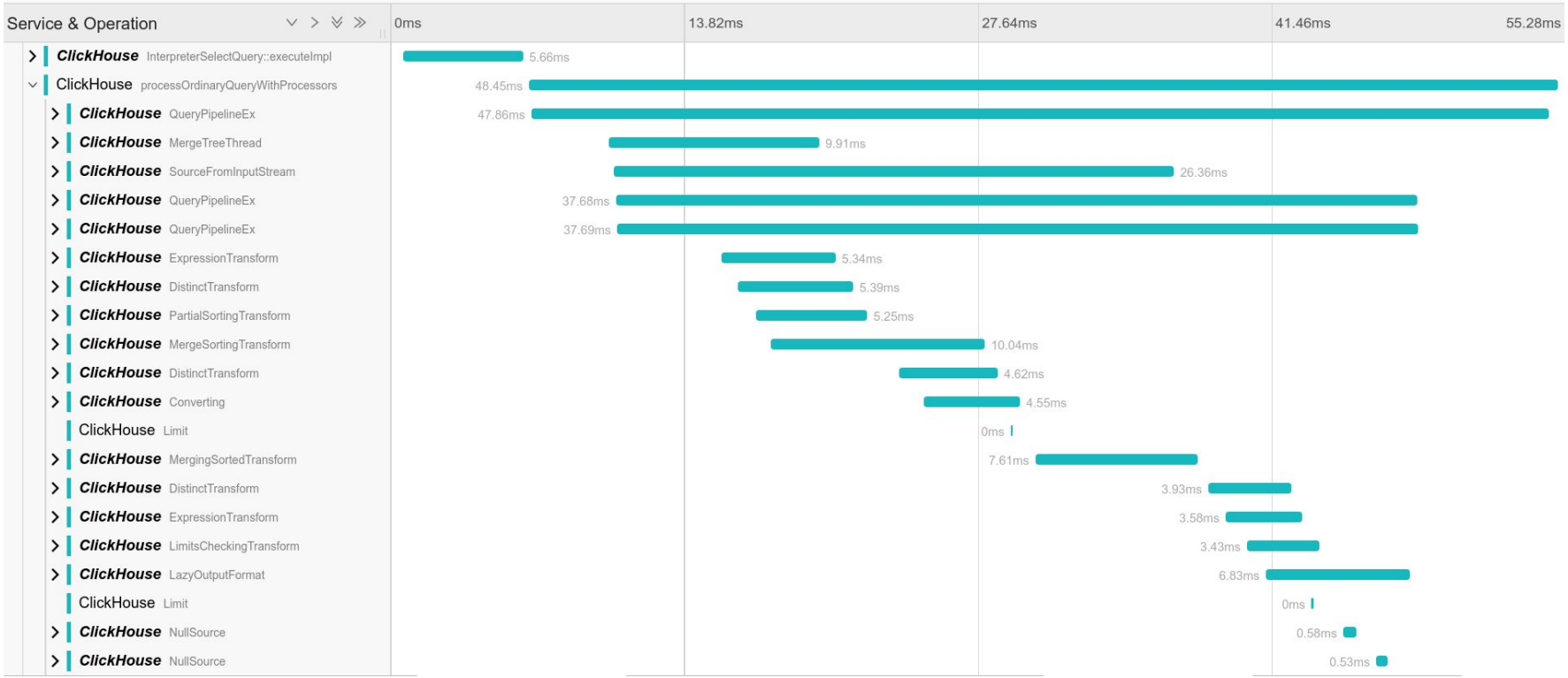
Интерфейс системы Jaeger

```
SELECT DISTINCT s FROM remote('127.0.0.{1,2}', currentDatabase(), data) ORDER BY x + y, s LIMIT 10;
```

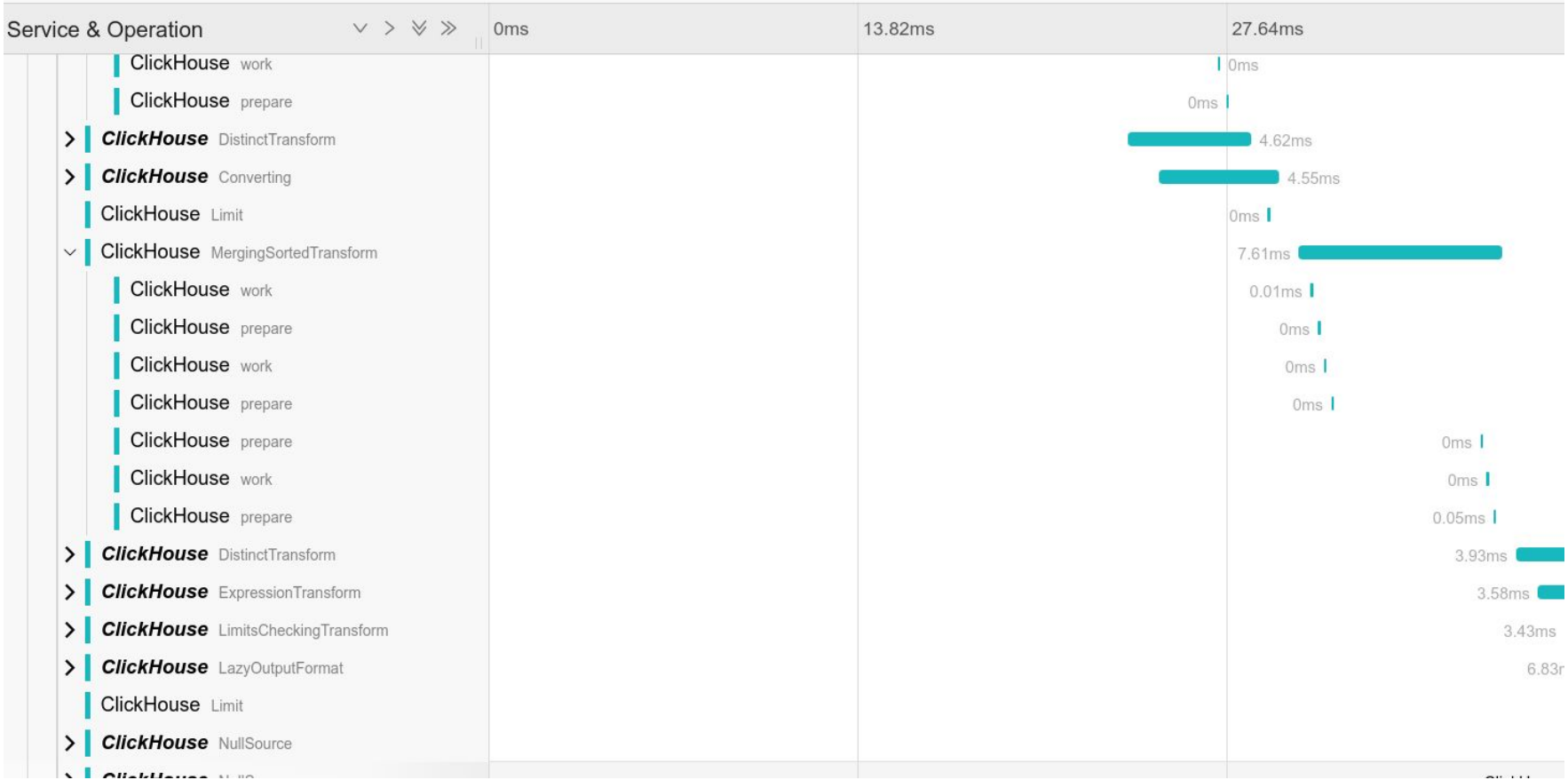




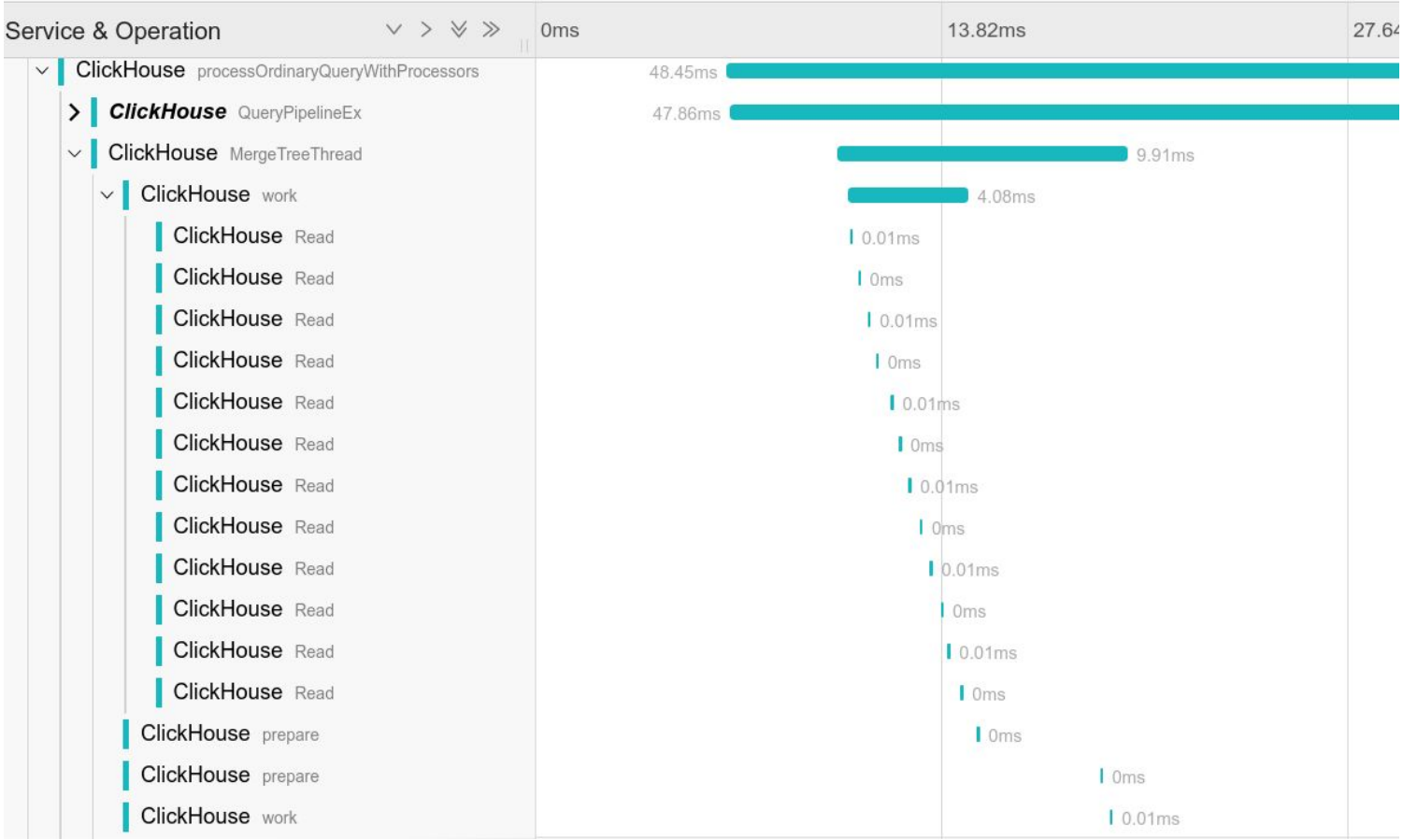
Трассировка интерпретатора запроса



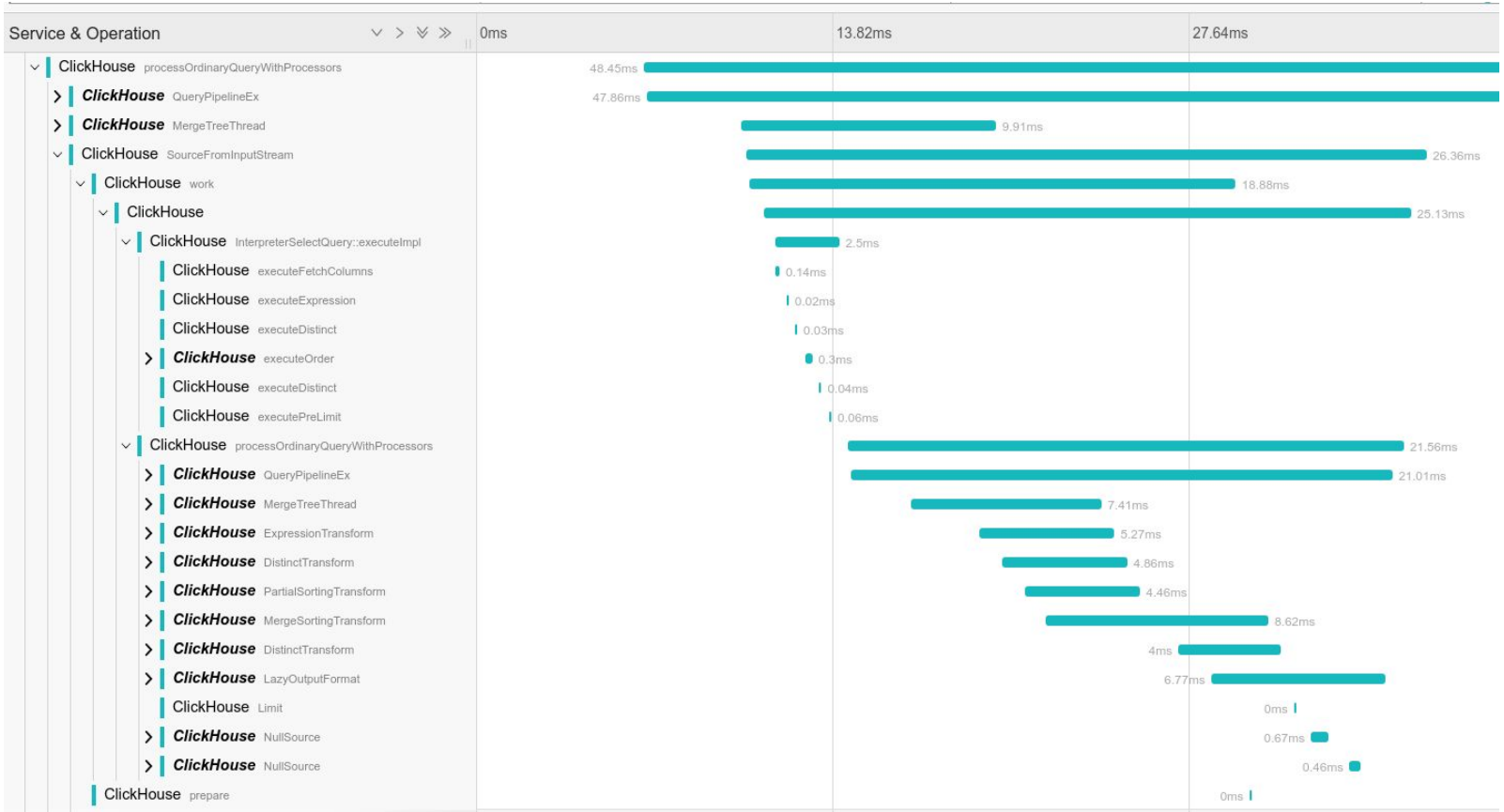
Исполнение запроса на основе Processors



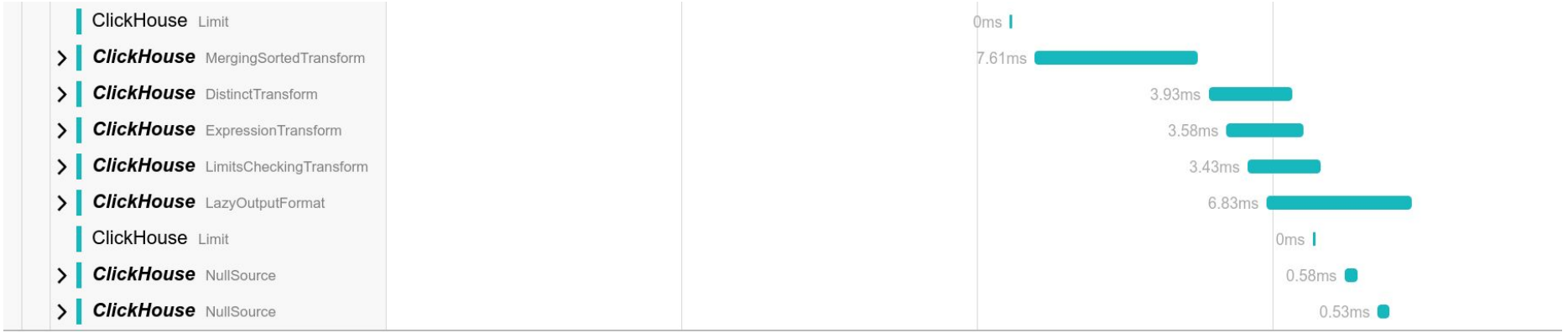
Стадии исполнения процессора MergingSortedTransform



Чтения внутри MergeTreeThread (из таблицы с движком MergeTree)



Распределенный запрос к другому шарду таблице



Слияние результатов с двух шардов и вывод результата запроса

Результаты работы

1. Удалось поддержать OpenTracing API в ClickHouse и подключиться к системе Jaeger Tracing.
2. Удалось добиться подробной трассировки с выделением логики запроса.
3. Трассировка действительно является распределенной.
4. Есть возможность использовать ClickHouse как подсистему другой системы, поддерживающей OpenTracing API.

Таким образом, все поставленные в данной работе задачи были успешно выполнены.

А также удалось получить подтверждение целесообразности внедрения OpenTracing в ClickHouse.