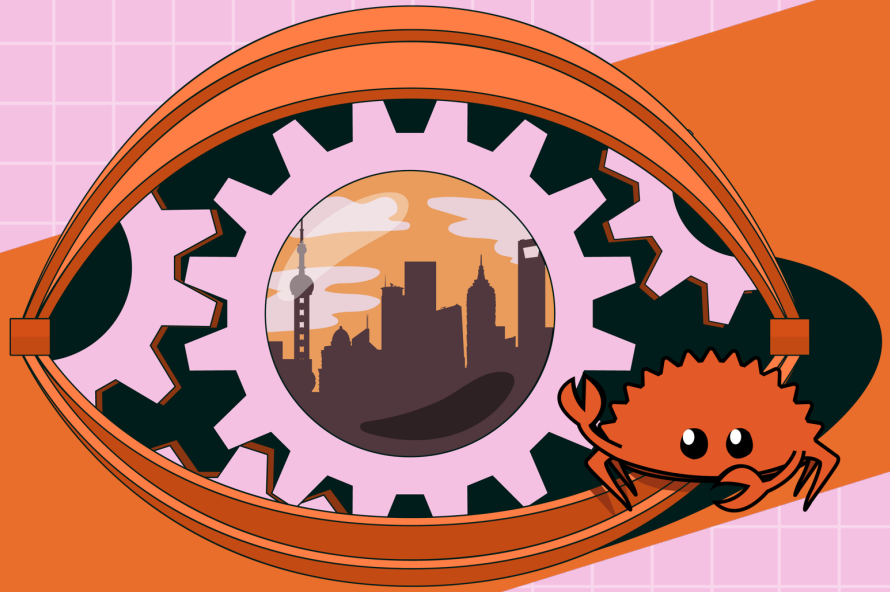
RUST CHINA CONF 2023

第三届中国Rust开发者大会



6.17-6.18 @Shanghai

使用Rust与ClickHouse构建高效可靠的日志系统

刘炜
腾讯云（专有云）

大纲

- 自我介绍
- 系统介绍
- 整体架构
- 系统实现
- 遇到问题

自我介绍

- 大龄码农
- 做过嵌入式/CDN/数据库开发
- 从C/C++到Rust
- 现在在腾讯云（专有云）从事日志系统的开发

系统介绍

- 属于腾讯专有云PaaS平台(TCS)
- 承接TCS底座日志
- 从Loki=>Menicus
- 提供日志的搜索/报警/处理等功能

系统介绍

- 为什么放弃 Loki
  - 资源占用过大
  - 统计/计算能力比较弱
  - 组件过多，排查问题比较困难
  - 商业使用不友好的开源协议
- 为什么选择Mencius+ClickHouse
  - 存储计算与业务分离
  - 计算/统计能力更强
  - 资源占用更小，性能更好
  - 更友好的开源协议

系统介绍

- 多种接入方式
  - Agent
  - Client
- 查询
  - LogQL
  - OpenTSDB
- 其他功能
  - 多租户
  - 自定义索引
  - 多维度统计
  - 鉴权
  - ...

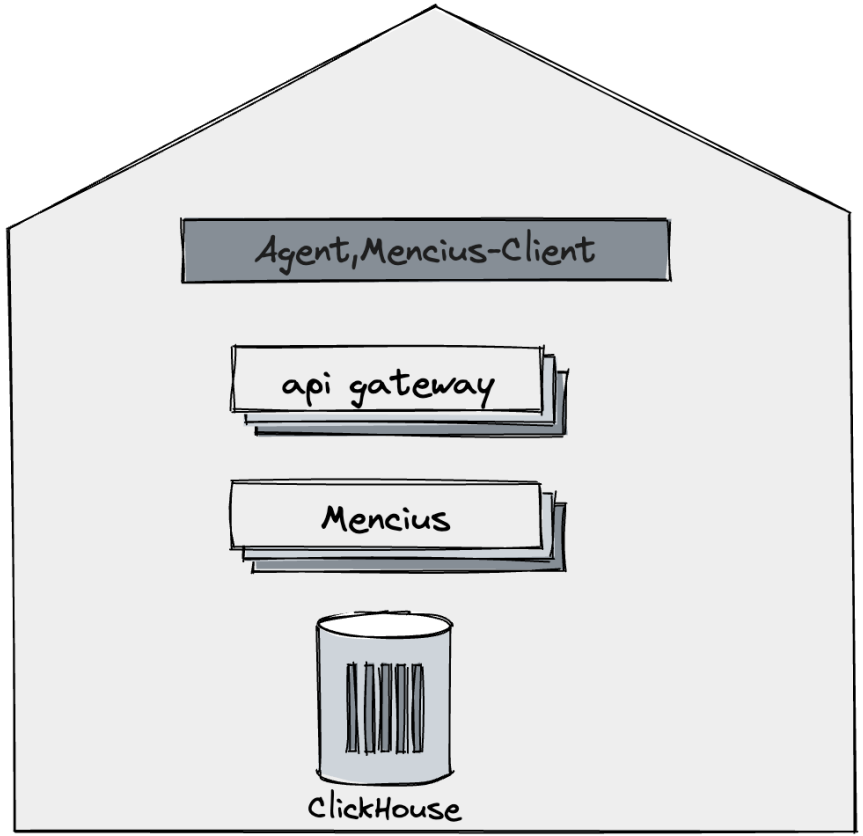
系统介绍

- 写入
  - 每天 100G
- 磁盘
  - 压缩比 1:13
- 内存
  - Mencius
    - 200M左右
  - ClickHouse
    - 2G以下

```
SELECT
    partition,
    formatReadableSize(sum(bytes_on_disk)),
    sum(data_uncompressed_bytes) / sum(data_compressed_bytes)
FROM system.parts
WHERE partition >= '20220806'
GROUP BY partition

Query id: cee8d8d0-91fb-4f08-b393-ac93b192463f

┌─partition─┬─formatReadableSize(sum(bytes_on_disk))─┬─divide(sum(data_uncompressed_bytes), sum(data_compressed_bytes))─┐
│ 20220807  │ 13.77 GiB                              │                                                  12.898084095101234 │
│ 20220806  │ 13.13 GiB                              │                                                   13.50751067635851 │
│ 20220808  │ 4.33 GiB                               │                                                   14.53635756025943 │
│ tuple()   │ 3.38 MiB                               │                                                   6.923569579029528 │
└───────────┴────────────────────────────────────────┴──────────────────────────────────────────────────────────────────────┘

4 rows in set. Elapsed: 0.520 sec. Processed 1.65 thousand rows, 69.90 KB (3.17 thousand rows/s., 134.37 KB/s.)
```

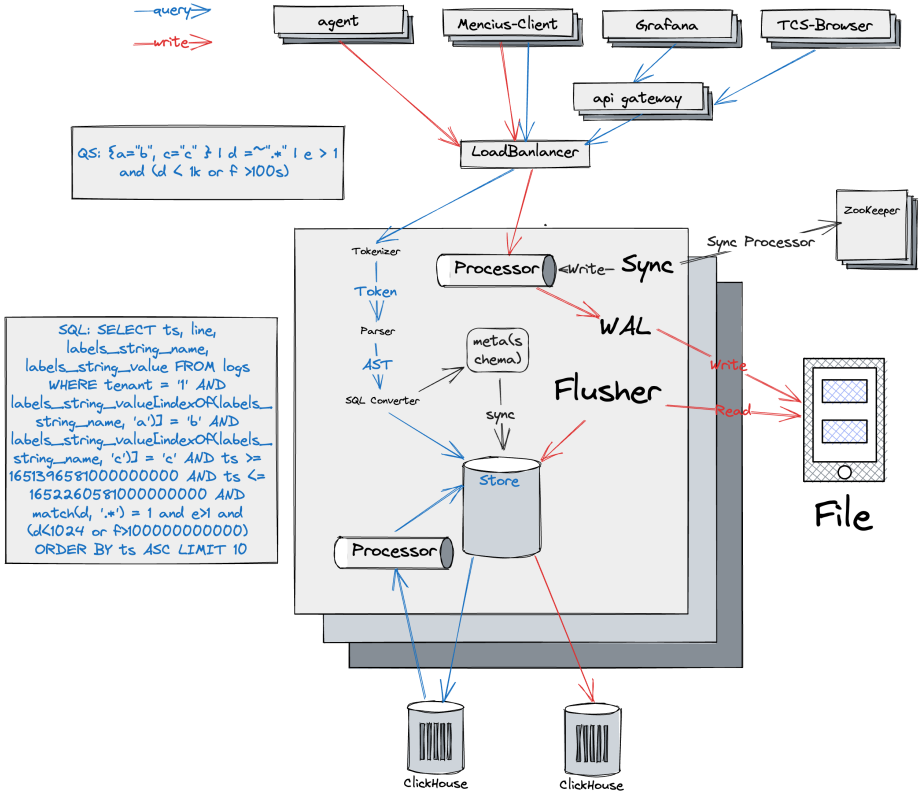
整体架构

- 接入端
- API Gateway
- 日志服务(Mencius)
- 存储(ClickHouse)



系统实现

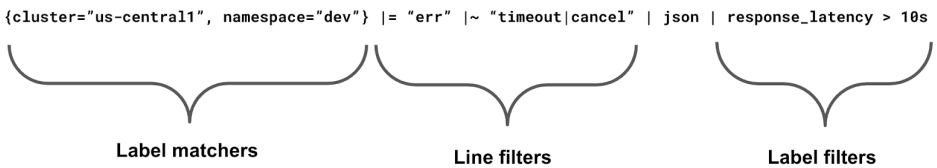
- 协议层
- 处理层
- 计算层
- 存储层



协议层

- 支持协议
  - Loki
    - 写入
      - Json/ProtoBuf
    - 查询
      - LogQL
        - Log Queries
        - Metric Queries
  - OpenTSDB
    - 统计/计算

```
{
  "streams": [
    {
      "stream": {
        "label": "value"
      },
      "values": [
          [ "<unix epoch in nanoseconds>", "<log line>" ],
          [ "<unix epoch in nanoseconds>", "<log line>" ]
      ]
    }
  ]
}
```



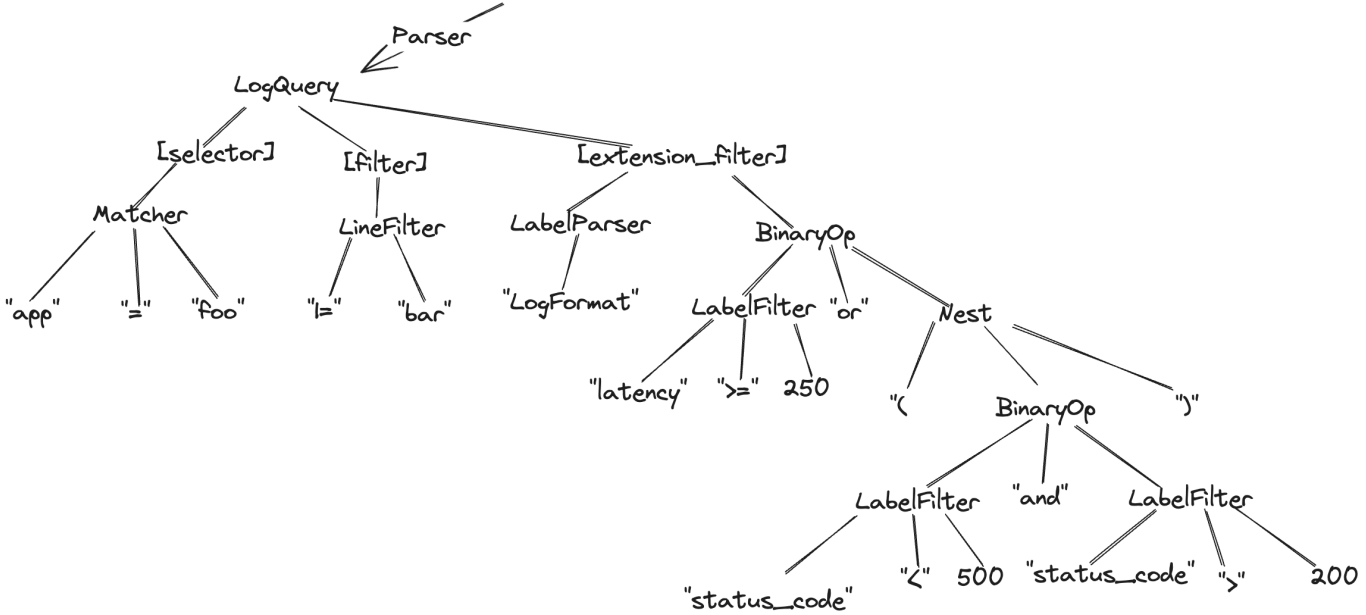
协议层

- Parser
  - 手写
    - 为什么？
    - LogQL=>Token=>AST=>SQL
  - 测试
    - Fuzz testing

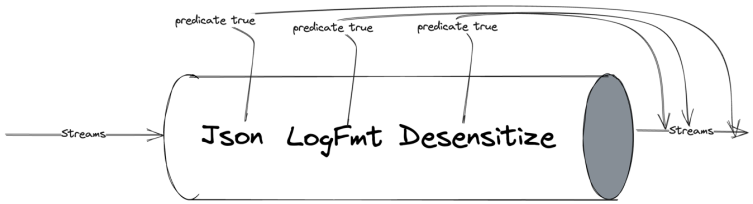
```
{app="foo"} |= "bar"  | logfmt| latency >= 250ms or ( status_code < 500 and status_code > 200)
```

Tokeninzer

```
[OpenBrace, Identifier(Word { value: "app", keyword: None }), EQ, String("foo"),
CloseBrace, Whitespace(Space), PipeExact, Whitespace(Space), ........
 Whitespace(Space), Gt, Whitespace(Space), Number(Number("200")), CloseParenthesis]
```



处理层



- Processor
  - 初始化
    - 静态/动态
  - 执行
    - 读/写
  - 结构
    - prediccate
    - process(&self, streams: Streams) -> Streams
  - 类型
    - 修改原始数据
    - 抽取原始数据字段
      - Json/LogFmt

计算层

- 大部分计算交给ClickHouse
  - Aggregation operators
- Vector 的计算
  - 报警使用
  - Binary operator
    - >, == …
  - Vector matching
    - 匹配两个Vector

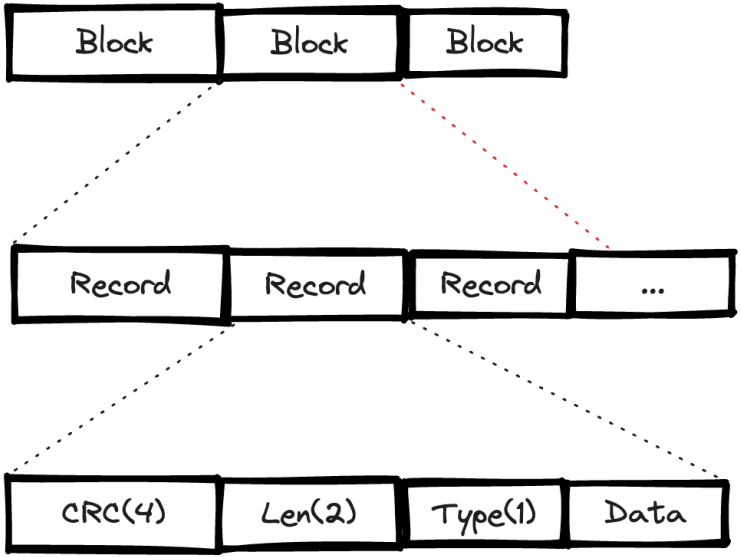
```
count_over_time({job="nginx"}[10s])
```



```
SELECT count() AS __agg_values, __start_ts,
       labels_string_name,
        labels_string_value,
         FROM logs_5 WHERE
         AND job = 'nginx'
         AND ts ≥ (toInt64(1640172995323027203) AS __start_ts)
         AND ts ≤ (1640173005323027203 AS __end_ts)
          GROUP BY labels_id
```

```
sum(rate({app="foo", env="production"} ⊨ "error" [5m])) by (job)
              /
sum(rate({app="foo", env="production"}[5m])) by (job)
            > 0.05
```

存储层



- WAL
  - 移植LevelDB的 WAL
    - 定长的 Block
  - Batch 写入 ClickHouse
- Flush Worker
  - 异步任务定时刷新 WAL
  - 清理策略
- Schema同步
  - 转换SQL

存储层

- ClickHouse
  - 使用Array来保存 Labels
  - 物化列
    - Lowcardinality(String)
      - 加速查询
  - Skipping Indexes
    - 日志全文索引

```
{
  "streams":
  [
   {
     "stream": { "name": "bar2"},
     "values": [
                  [ "1570818238000000000", "fizzbuzz" ] ,
                  [ "1570818238000000000", "fizzbuzz" ]
                ]
   }
   ]
 }
```

```
Create table logs {
 module String,
 name Lowcardinality(String) MATERIALIZED labels_string_value[indexOf(labels_string_name, 'name')] ,
.......
lables_string_name Array(String),
labels_string_value Array(String),
// log line
line String,
.....
index_line line TYPE ngrambf_v1(10, 256, 2, 0),

}
```

问题

- Rust
  - Lifetime侵入性比较强
  - 库质量参差不齐
- ClickHouse
  - 可运维性弱
  - 强Schema带来不够灵活的缺点
  - 全文索引支持比较弱

Thank you !