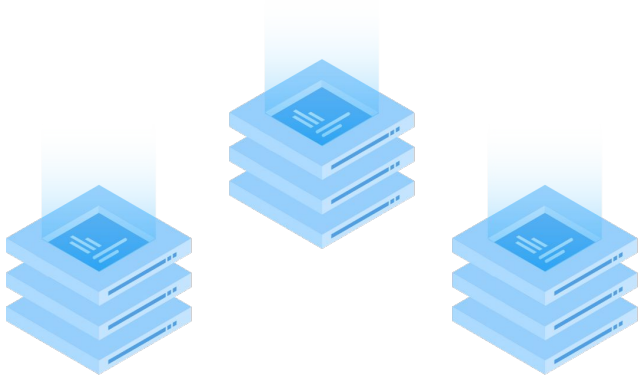
“监”听风云

Presented by Atman Zheng

PingCAP TiDB



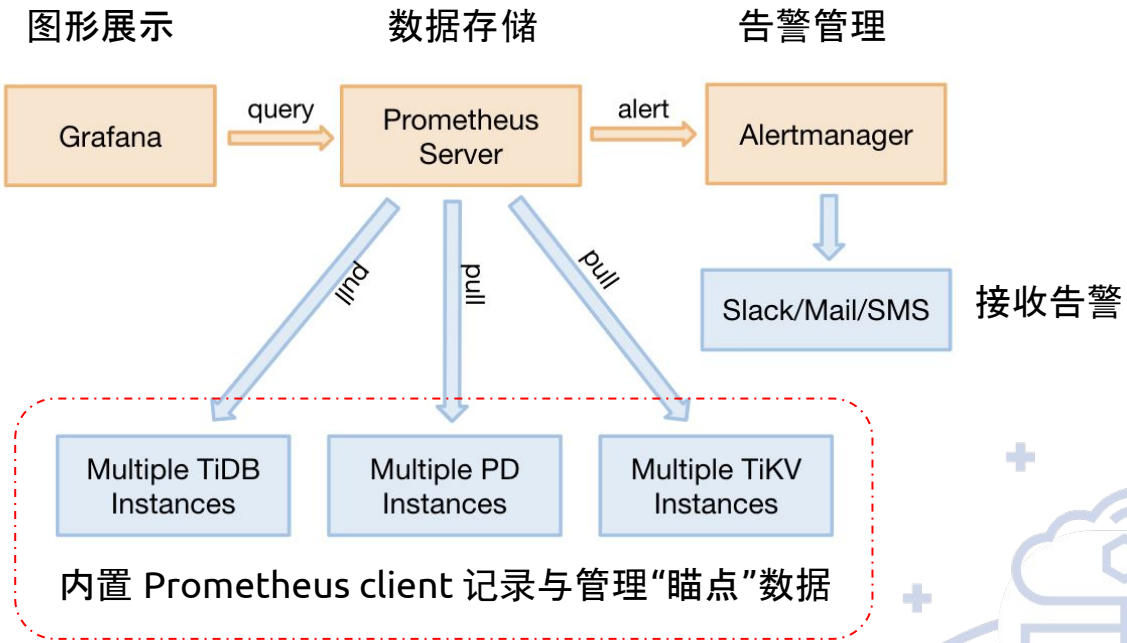
题纲

- 构建
  - 监控架构（监控组件成员介绍）
    - 铁三角
  - 监控时那些事（监控运维场景问题）
    - TiKV 下线后数据怎么还在
    - Grafana 看不到数据了
    - Grafana 与公司业务平台数据量对不上
- 话剧
  - 来自心中的疑问（这么多监控指标看些啥）
    - 监控数据怎么来的 / --
    - TiDB-Map & TiDB-Book 解剖“事物”逻辑
  - 后台解读某个场景
    - TiDB slow query / 踏雪寻梅
      - PD、TiDB、TiKV 在事物过程中处理了什么？
      - 事物逻辑与监控

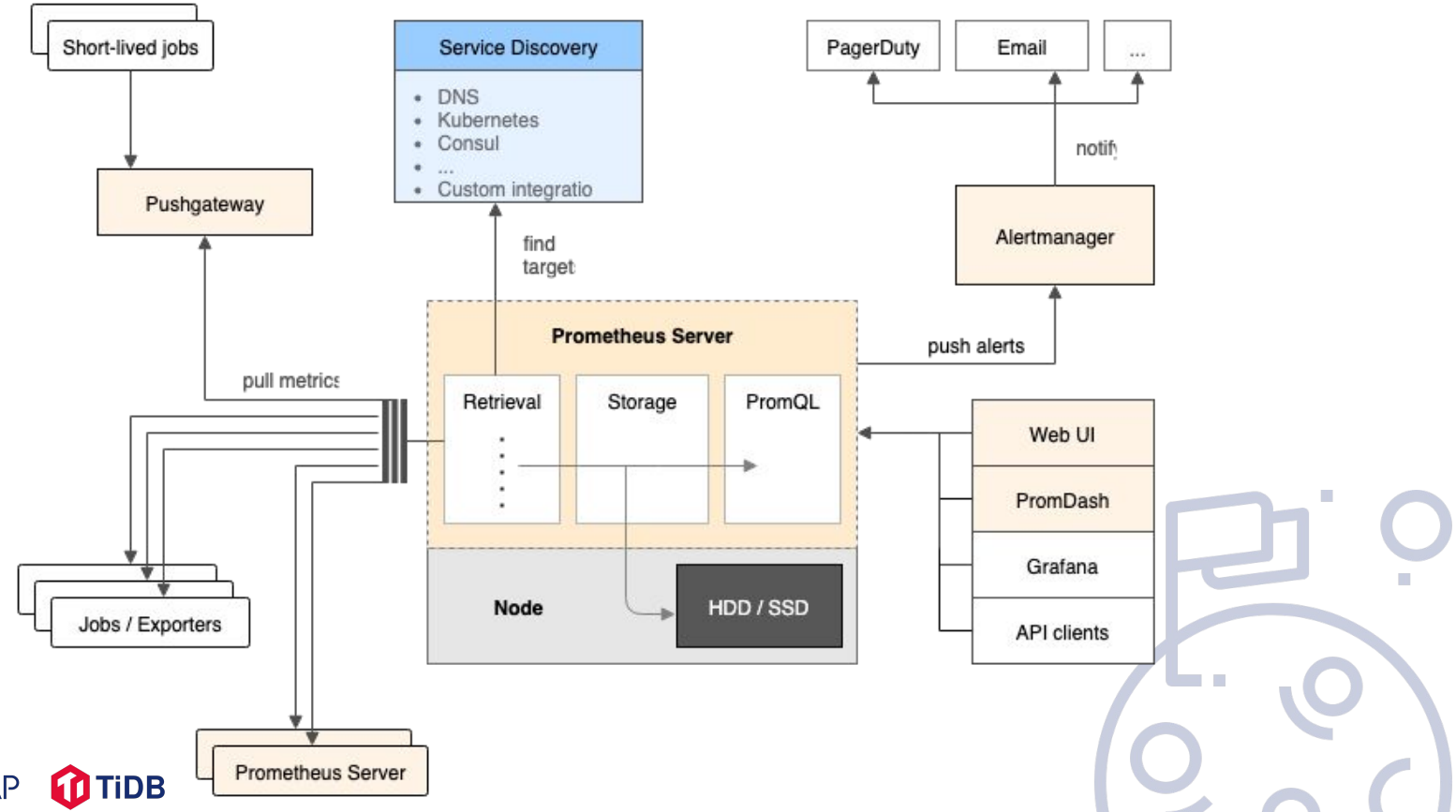
PingCAP

Part 1️⃣ - TiDB 监控架构

铁三角


图形展示
数据存储
告警管理
Grafana
query
Prometheus Server
alert
Alertmanager
pull
pull
pull
Slack/Mail/SMS
接收告警
Multiple TiDB Instances
Multiple PD Instances
Multiple TiKV Instances
内置 Prometheus client 记录与管理“瞄点”数据

Prometheus 数据 1️⃣



PingCAP TiDB

Prometheus 数据 2️⃣

histogram_quantile(0.999, sum(rate(tidb_server_handle_query_duration_seconds_bucket[1m])) by (le))

直方图函数（数据采样比，数据运算（数据窗口范围（数据 metrics bucket [时间范围]）by（分组））

该直方图计算后得知 TiDB query duration 信息，比如 P999、P99、P95、P80，在不同数据采样比情况下会有不同的误差；此时根据业务对 query durtion 的延迟要求，比如 5000 tps/sec 并发情况下 duration 50ms，可以从 80 > 99 > 999 逐级分析

引用资料

- Prometheus
  - 官网版本
  - 第三方中文翻译
- Grafana
  - 官网文档很给力，图文并茂（目前 TiDB 使用 Grafana 6.0 版本）
- TiKV in Prometheus
  - https://pingcap.com/blog-cn/tikv-source-code-reading-3/
  - https://pingcap.com/blog-cn/tikv-source-code-reading-4/
  - https://pingcap.com/blog-cn/use-grafana-to-monitor-and-analyze-tidb-metrics/

PingCAP

Part 2️⃣ - 监控那些事

Grafana 看不到数据了

- metrics 本身没有数据
  - 事物运行时没有触发到 metrics 逻辑
- 网络防火墙阻断（端口列表）
  - 云虚拟机检查 **安全组** 规则
  - 物理机环境检查 **硬件防火墙** 规则
- 配置文件不正确
  - 依次检查 tidb metrics-status port、prometheus 配置文件、grafana 配置参数
- 目标组件进程退出
  - 进入 tidb-ansible 目录
  - ansible tidb_servers -m shell -a “ps aux | grep tidb-server” 依次查看 tidb pd tikv 组件
- 目标组件端口冲突
  - TiDB、TiKV metrics-status 端口冲突不影响启 动，会影响监控数据展示

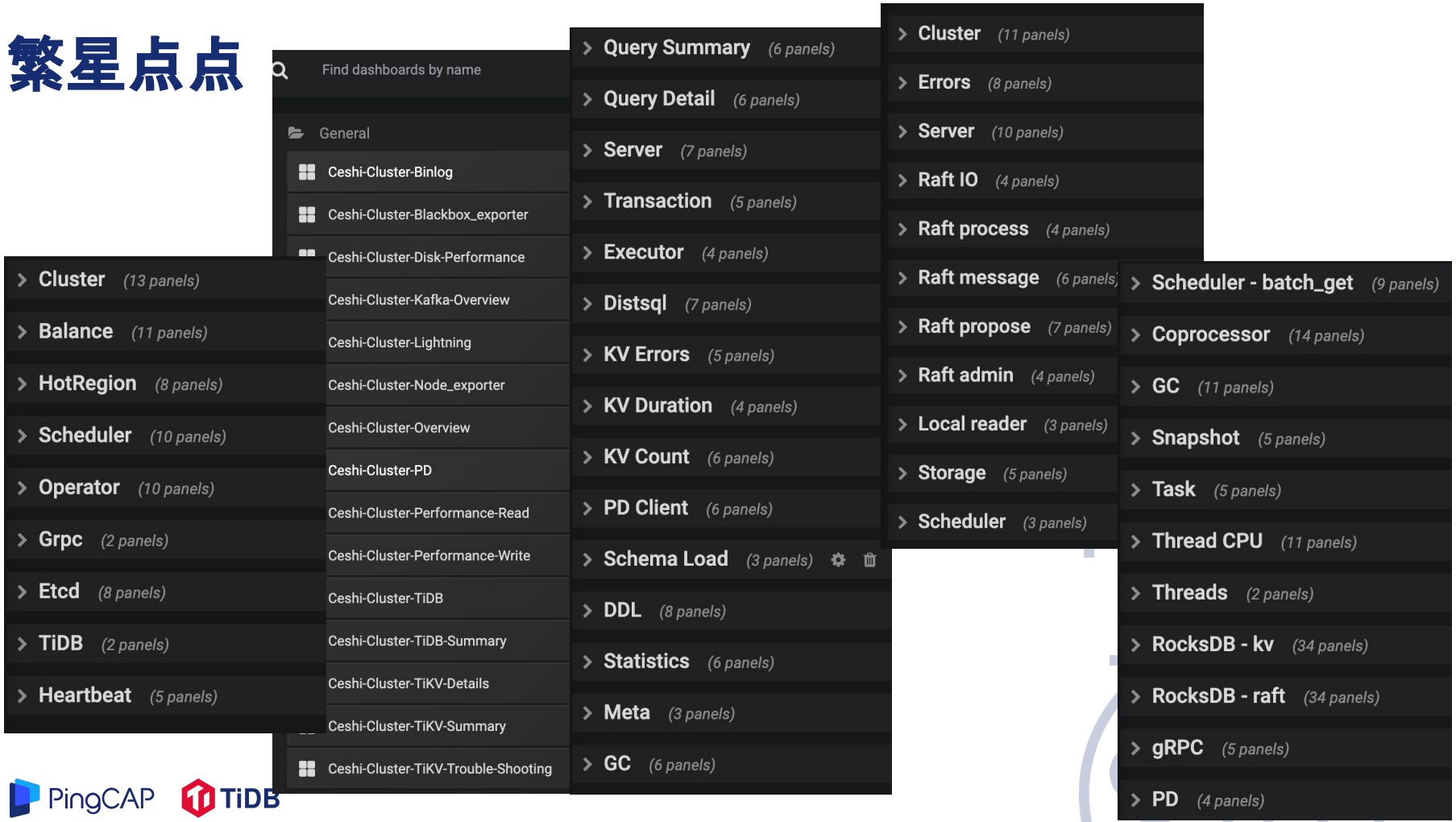
残留的 TiKV 数据

- 清理 Prometheus 数据
  - 使用 Prometheus 2.1 以上版本
  - 打开 --web.enable-admin-api 参数
  - 详情看官网 Prometheus Admin api
  - curl -X POST -g 'http://127.0.0.1:9090/api/v1/admin/tsdb/delete_series?match[]={instance="10.0.1.4:20181",job="tikv"}'

PingCAP

Part 3️⃣ - 来自心中的疑问

繁星点点



PingCAP TiDB

它是怎么来的 1

- tidb_server_handle_query_duration_seconds_bucket

2 code results in pingcap/tidb

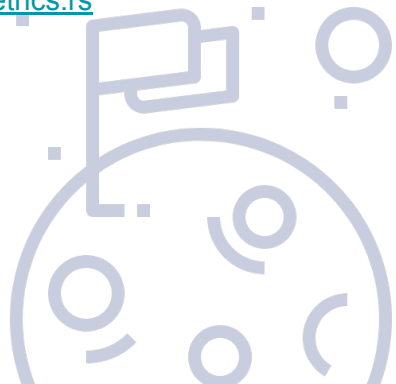
```
// Metrics
var (
    QueryDurationHistogram = prometheus.NewHistogramVec(
        prometheus.HistogramOpts{
            Namespace: "tidb",
            Subsystem: "server",
            Name:      "handle_query_duration_seconds",
            Help:      "Bucketed histogram of processing time (s) of handled queries.",
            Buckets:   prometheus.ExponentialBuckets(0.0005, 2, 22), // 500us ~ 2097s
        }, []string{LblSQLType})
```

```
35                      Name:      "handle_query_duration_seconds",
```

Go Showing the top match Last indexed 3 days ago

它是怎么来的 2

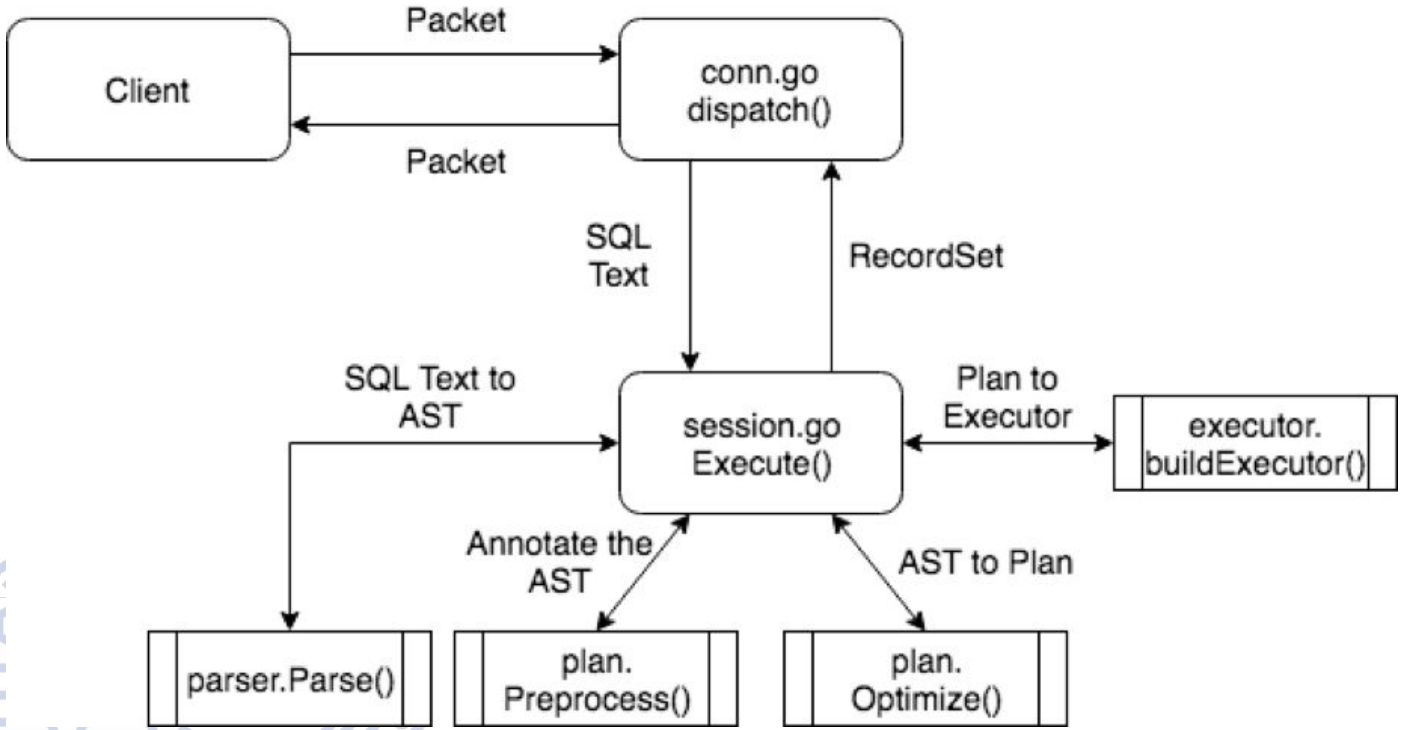
- PD & TiKV 的方式与 TiDB 的搜索方式不同，但命名格式相等
- pd_cluster_status
  - 本条 metrics 信息来自 https://github.com/pingcap/pd/tree/master/server/statistics
- tikv_pd_heartbeat_tick_total（统计 region & leader 数量）
  - tikv_pd_ TiKV 与 PD 之间的交互
  - heartbeat_tick_total 心跳统计
- tikv_engine_size_bytes
  - TiKV 存储大小，显示单位 bytes
  - https://github.com/tikv/tikv/blob/master/components/engine/src/rocks/util/engine_metrics.rs



PingCAP

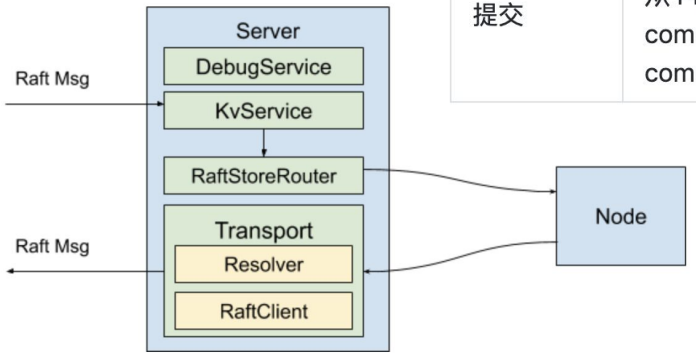
Part 3️⃣ - SQL 的一生

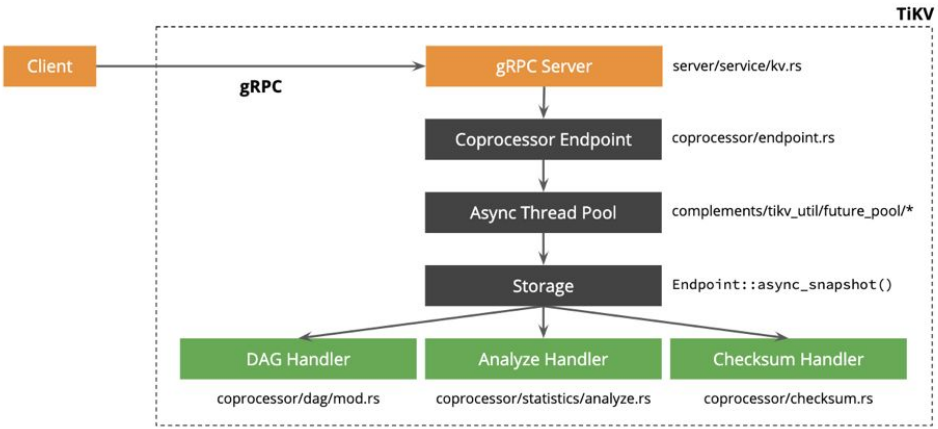
TiDB 事物执行框架



TiKV 事物处理

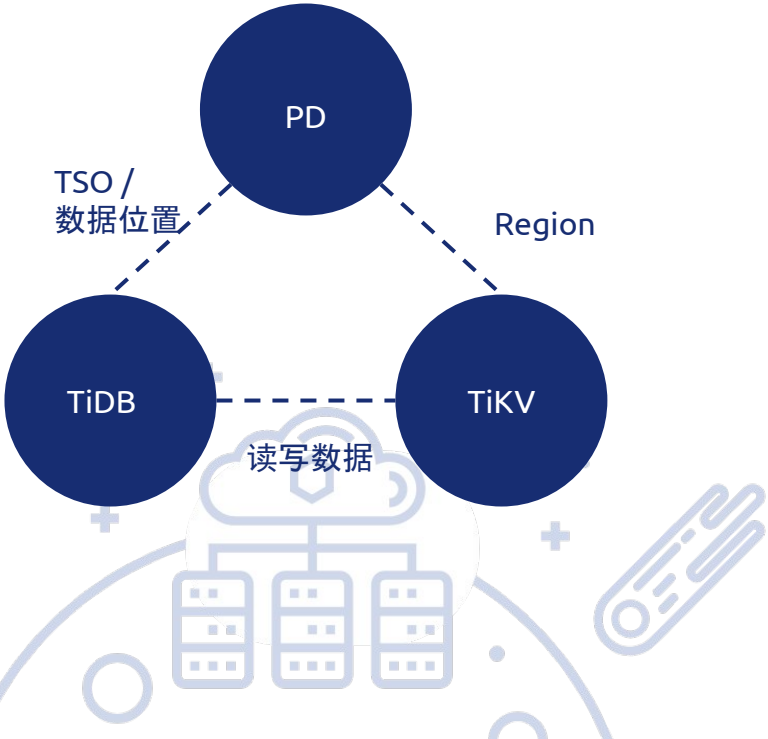
用户操作	tikv client	TiKV
开始事务	从 PD 取 `start_ts`	
进行读操作	使用 `start_ts` 向 TiKV 发送读请求	处理读请求
进行写操作	写入到 buffer 中	
进行读操作	如果被写到了 buffer 中，从 buffer 中读；否则向 TiKV 发送读请求	处理读请求
提交	prewrite 写入到 buffer 中的所有 key 从 PD 取 `commit_ts` commit primary key commit 其它 key	处理 prewrite 请求 处理 commit 请求 处理 commit 请求



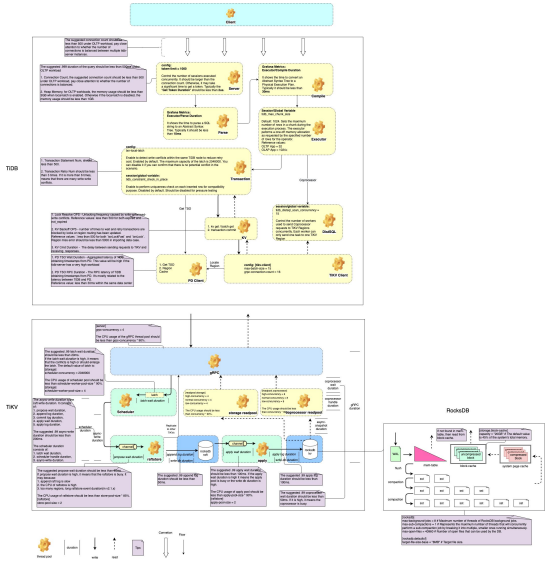


SQL 的一生

简洁版



详细版



图太大，贴个链接

PingCAP TiDB

引用资料

- TiDB SQL 的一生
  - https://pingcap.com/blog-cn/tidb-source-code-reading-3/
- TiDB PD TiKV 交互地图
  - https://github.com/pingcap/tidb-map/tree/master/maps
- 新鲜出炉的 TiDB Book
  - https://pingcap-incubator.github.io/tidb-in-action/
- TiDB 源码阅读系列
  - TiDB 源码阅读
  - TiKV 源码阅读

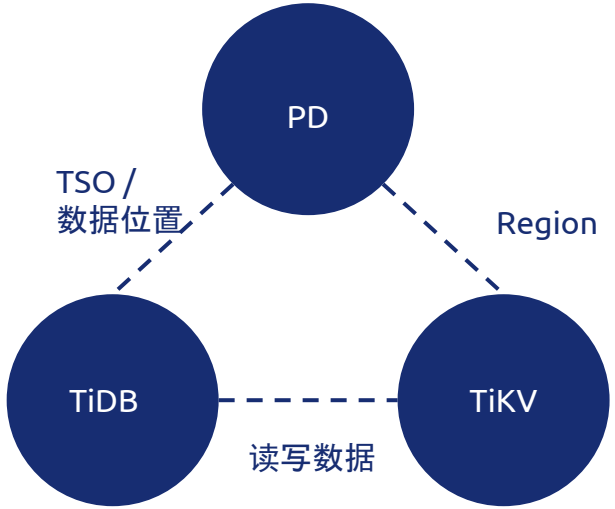
PingCAP

Part 5️⃣ - 踏血寻梅

金三角

- 题：前方业务部门说昨晚遇见一些抖动，让 DBA 帮助协查下是什么原因？

- 当我们接到以上问题时应该怎么处理呢？
  - 聊业务逻辑（根据业务思考可能出现的问题）
  - 巡检看监控（或许时间范围）
  - 查服务日志（获取物证）
  - 分析各种信息（与业务部门抓凶）
  - *本次演示是 3.0，与 4.0 版本操作略有不同*
- 根据这些步骤来分析下 TiDB slow query 场景



初步信息

- 假设场景：
- 获取 TiDB 集群配置
  - 使用 5 台 64vcpu + 256G ram 部署 15 个 TiKV 服务
  - 3 台 16vcpu + 16G ram 部署 PD 服务
  - 5 台 32vcpu + 128G ram 部署 TiDB 服务
- 业务高峰期 TPS 5w/sec，P95 duration 15ms

简洁巡检

- 监控查看组件运行状态、
  - 从 overview 入手
  - 先观察组件是否在线
  - 再看目前主机资源使用量
  - 随后看 TiDB、TiKV 资源是占用
  - *从毛刺分析目前集群是否 稳定运行*
- 收集分析日志
  - TiDB slow query.log or 使用 TiDB 系统表 SLOWQUERY
- *演示 demo*

PingCAP

Part 6️⃣ - 故障分析&恢复

日常故障处理

- 日常风险
  - *--- TiDB slow query*
  - *txn too large*
  - *GC life timeout*
  - *TiKV Region is Unavailable*
  - *TiKV server is busy*
  - *PD 调度问题*
- TiDB PD TiKV 交互地图
  - https://github.com/pingcap/tidb-map/blob/master/maps/diagnose-map.md
- 新鲜出炉的 TiDB Book
  - https://pingcap-incubator.github.io/tidb-in-action/

集群中多数节点不可用

- 多数节点不可用
  - PD / TiKV 基于 Raft 协议
  - TiDB-Server 单机(≈)无状态
- 场景
  - 跨机房部署时：单机房停电、断网
  - 单机房内机柜掉电、人为操作失误
- 影响
  - 一旦多数派不可用，集群或将不可用
  - 三副本坏两台(总 100 台)，当前事务有几率成功
- 修复
  - PD 全部损坏了用 PD recover
  - TiKV 多数派损坏也可恢复(有风险) TiKV ctl

PingCAP

Thank You !

答题 1️⃣

- 开放题：业务程序不稳定返回 error 9002: TiKV server timeout 信息，如何通过监控排查？
- 【单选】TiDB 监控平台中最后通过（）组件发送告警
  - 1. prometheus
  - 2. pushgateway
  - 3. alertmanager
  - 4. nodeexport
- 【单选】TiDB 监控平台中通过组件（）探活 TiDB、PD、TiKV 服务端口
  - 1. prometheus
  - 2. grafana
  - 3. blockbox export
  - 4. node export

答题 2️⃣

- 【单选】TiDB 处理更新、删除、写入动作的事物（事物成功提交情况下）需要向 PD 获取几次 TSO 信息
  - 1. 一次
  - 2. 两次
  - 3. 三次
  - 4. 多次
- 【单选】通过 TiDB 执行 select sum(*) from table，会使用（）接口 组件
  - 1. raw get
  - 2. kv get
  - 3. coprocessor
  - 4. schedules
  - 5. storage readpool