SACC

2012中国系统架构师大会

SYSTEM ARCHITECT CONFERENCE CHINA 2012

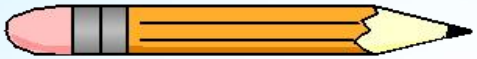
架构设计 · 自动化运维 · 云计算

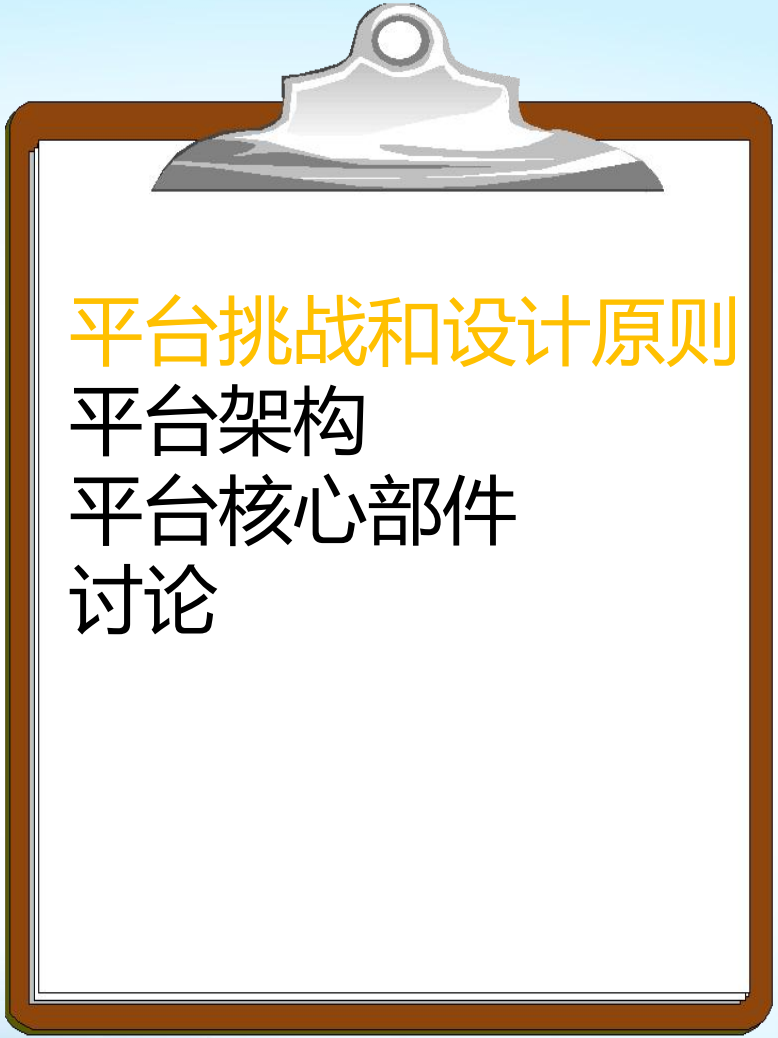
SequeMedia 盛拓传媒 | ITT168.com

低成本和高性能 MySQL云架构探索

关于我

- 淘宝核心系统资深技术专家 余锋
- 超过15年互联网行业的网络、内核以及底层软件开发经验
- 专注于高性能分布式服务器的研究和实现
- 擅长构建大规模集群服务器
- 对数据库系统和分布式文件存储有深入的研究





平台挑战和设计原则

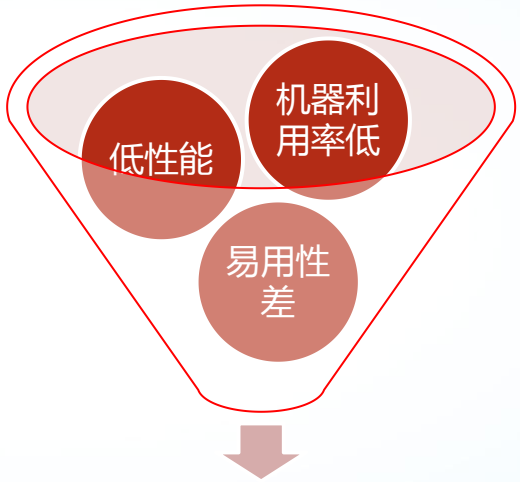
平台架构

平台核心部件

讨论

MySQL运维面临的问题

- 易用性偏差
- 性能
  - 软硬件未经优化，处于原始阶段
  - 不同阶段的软硬件性能相差巨大
- 集群
  - 主备不完全同步和备机利用率低
  - 以业务划分，集群分散管理，运维成本高

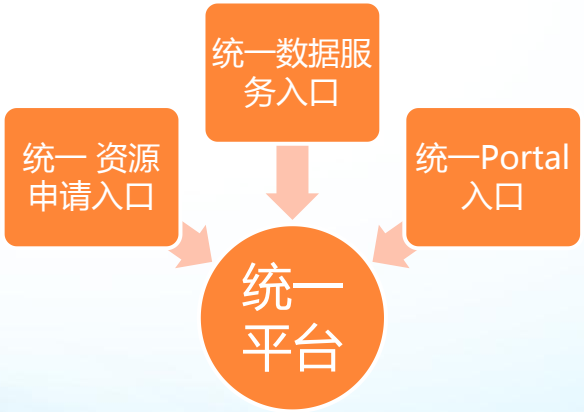


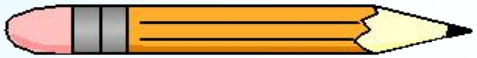
MySQL平台化要解决什么？

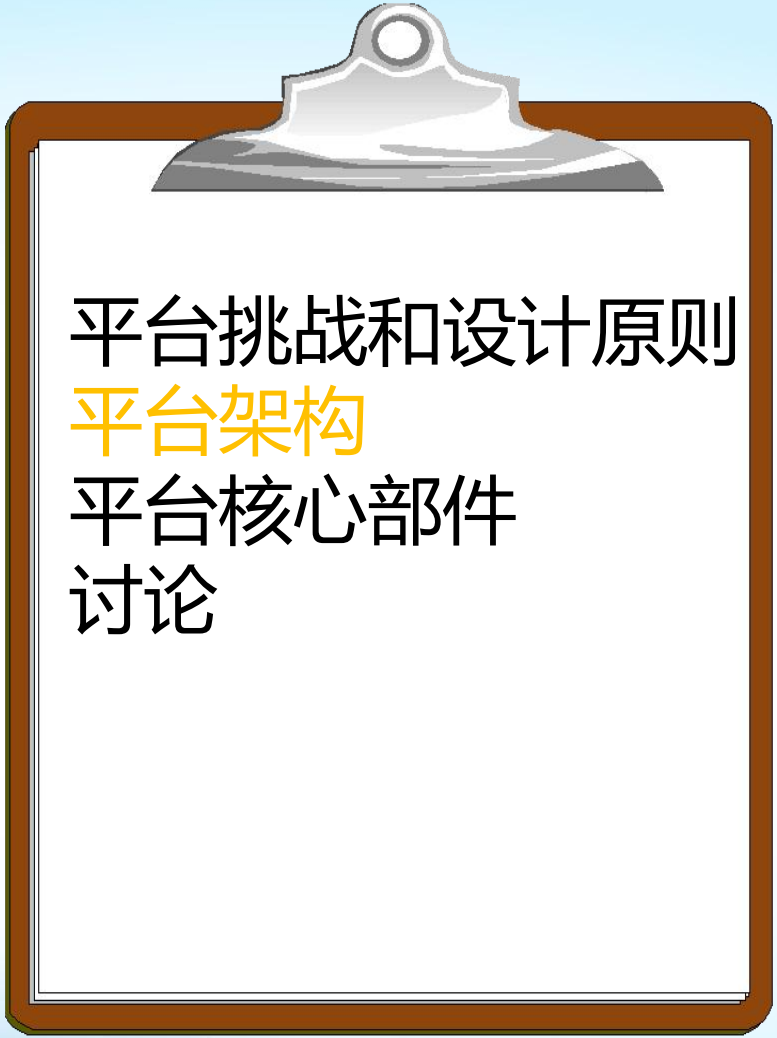
- 高效MySQL数据库服务支持，自动化运维
- 提高资源利用率，降低平台成本
- 7*24长期运行，屏蔽软硬件的变化

MySQL平台设计原则

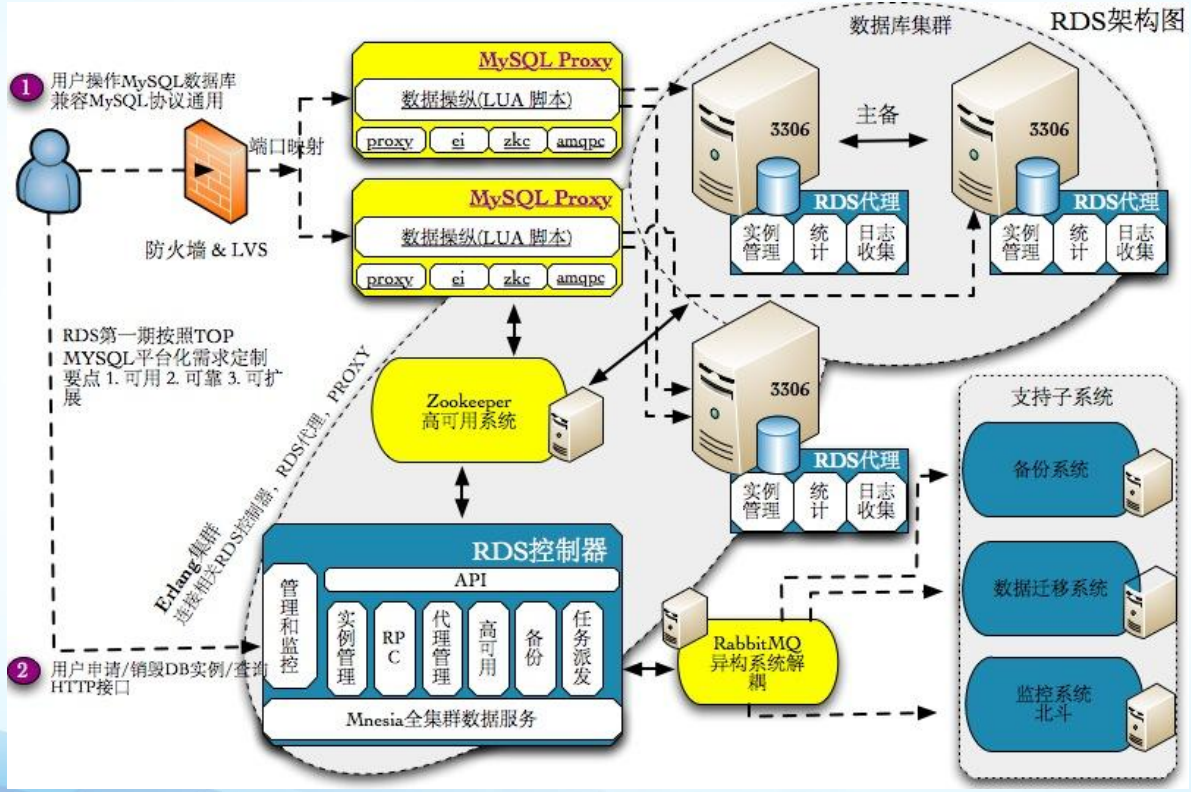
- 平台对外保持单一入口，对内维护单一的资源池。
- 保证服务的高可用性，消除单点故障
- 保证系统是弹性可伸缩的，可以动态的增加、删减计算与存储节点。
- 保证分配给用户的资源也是弹性可伸缩的，资源之间相互隔离。






平台挑战和设计原则
平台架构
平台核心部件
讨论

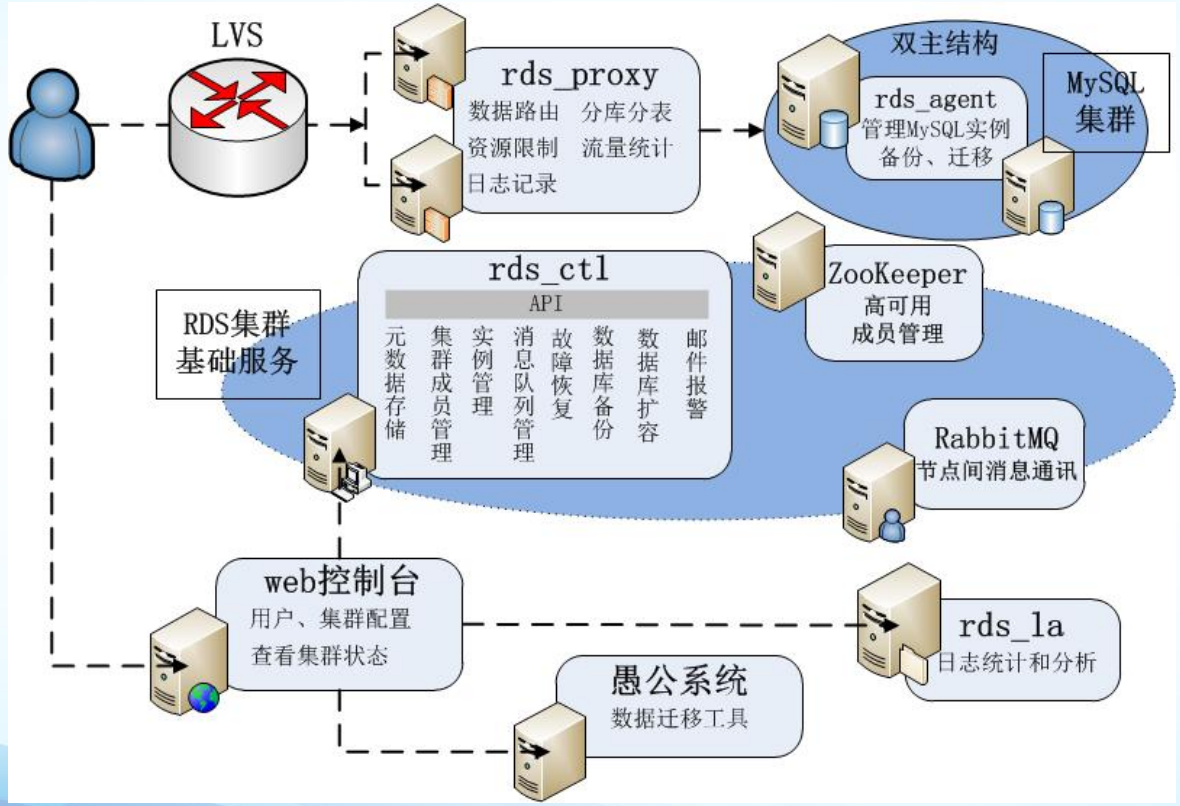
平台架构第一版


RDS架构图
数据库集群
1 用户操作MySQL数据库
兼容MySQL协议通用
端口映射
防火墙 & LVS
MySQL Proxy
数据操纵(LUA 脚本)
proxy
ei
zkc
amqpc
MySQL Proxy
数据操纵(LUA 脚本)
proxy
ei
zkc
amqpc
3306
主备
3306
RDS代理
实例管理
统计
日志收集
RDS代理
实例管理
统计
日志收集
RDS第一期按照TOP
MYSQL平台化需求定制
要点 1. 可用 2. 可靠 3. 可扩展
Zookeeper
高可用系统
3306
RDS代理
实例管理
统计
日志收集
支持子系统
备份系统
数据迁移系统
监控系统
北斗
Erlang集群
连接相关RDS控制器，RDS代理，PROXY
RDS控制器
管理和监控
API
实例管理
RPC
代理管理
高可用
备份
任务派发
Mnesia全集群数据服务
RabbitMQ
异构系统解耦
2 用户申请/销毁DB实例/查询
HTTP接口

第一版的经验和教训

- 经验
  - 采用开放成熟的第三方部件的好处
  - 开放的平台，方便用户扩展
  - 热部署和升级对不停机维护的意义
  - 容错系统设计的重要性
- 教训
  - 保持和MySQL的绝对兼容的重要性
  - 数据访问主路径必须短且稳定
  - proxy 性能、稳定性和成本的关系
  - 日志实时收集和处理的难度
  - 系统各部件部署的粒度，减少系统升级带来不良影响

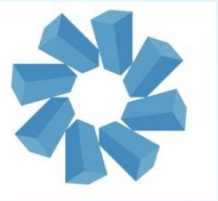
平台架构第二版


LVS
rds_proxy
数据路由 分库分表
资源限制 流量统计
日志记录
双主结构
rds_agent
管理MySQL实例
备份、迁移
MySQL
集群
RDS集群
基础服务
rds_ctl
API
元数据存储
集群成员管理
实例管理
消息队列管理
故障恢复
数据库备份
数据库扩容
邮件报警
ZooKeeper
高可用
成员管理
RabbitMQ
节点间消息通讯
web控制台
用户、集群配置
查看集群状态
rds_la
日志统计和分析
愚公系统
数据迁移工具

平台特性

- 平台足够稳定，支持热升级
- 支持几千台物理机规模，
- 以对用户透明的形式提供主从热备、数据备份、迁移、容灾、读写分离、分库分表功能
- 资源隔离，按需分配和限制CPU、内存和IO资源
- 不影响提供数据服务的前提下根据用户业务的发展动态的扩容和缩容
- 屏蔽数据节点不同的软硬件差异

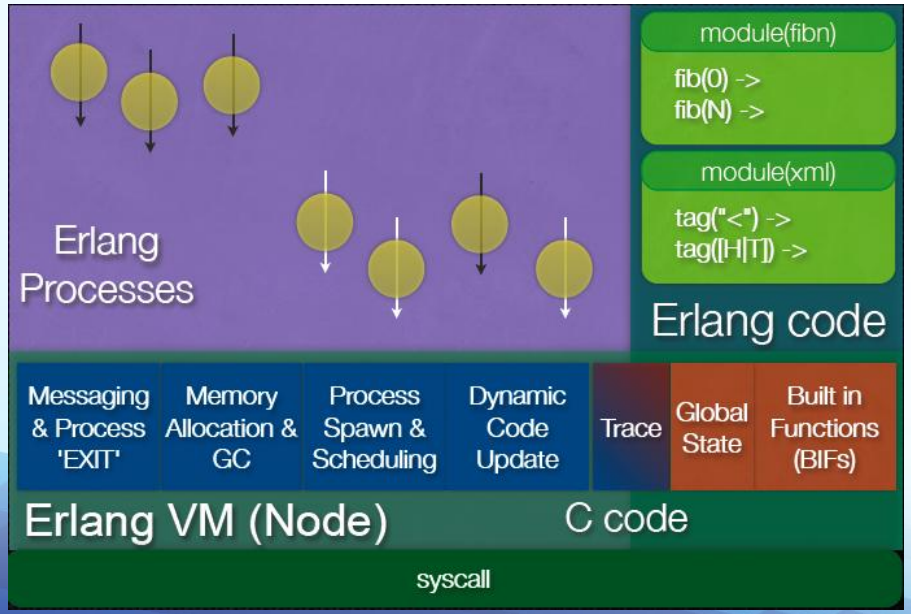
平台概况

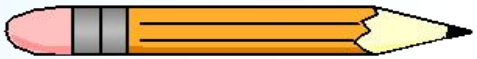


- 稳定性生产系统验证过
- 依赖的开源组件：Mnesia、Lvs、RabbitMQ、ZooKeeper
- 代码规模
  - 核心使用以高性能、健壮以及可伸缩性出名的Erlang语言开发
  - 5万行Erlang代码，3万c代码，2万其他代码
  - 六人团队，历时1年

为什么要用Erlang实现

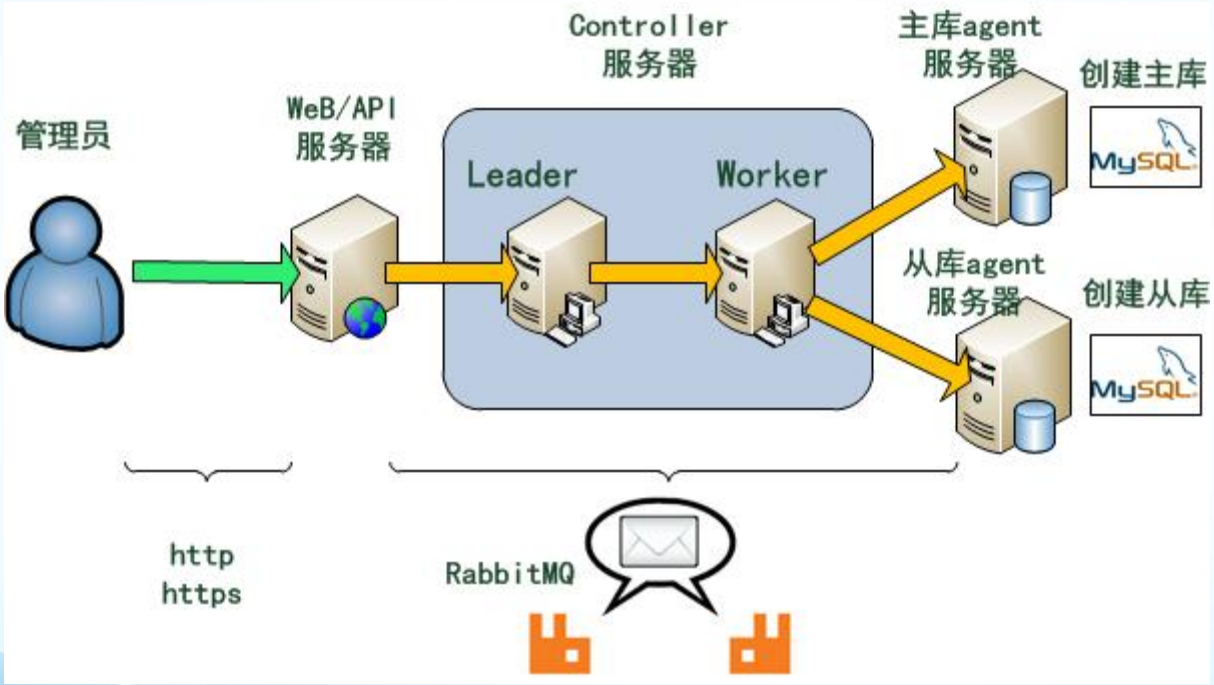
- 高并发,高性能,集群易扩展
- 时间检验的高可靠
- 强大的管理功能,方便的问题定位支持
- 强大的交互性, 与其他系统整合能力



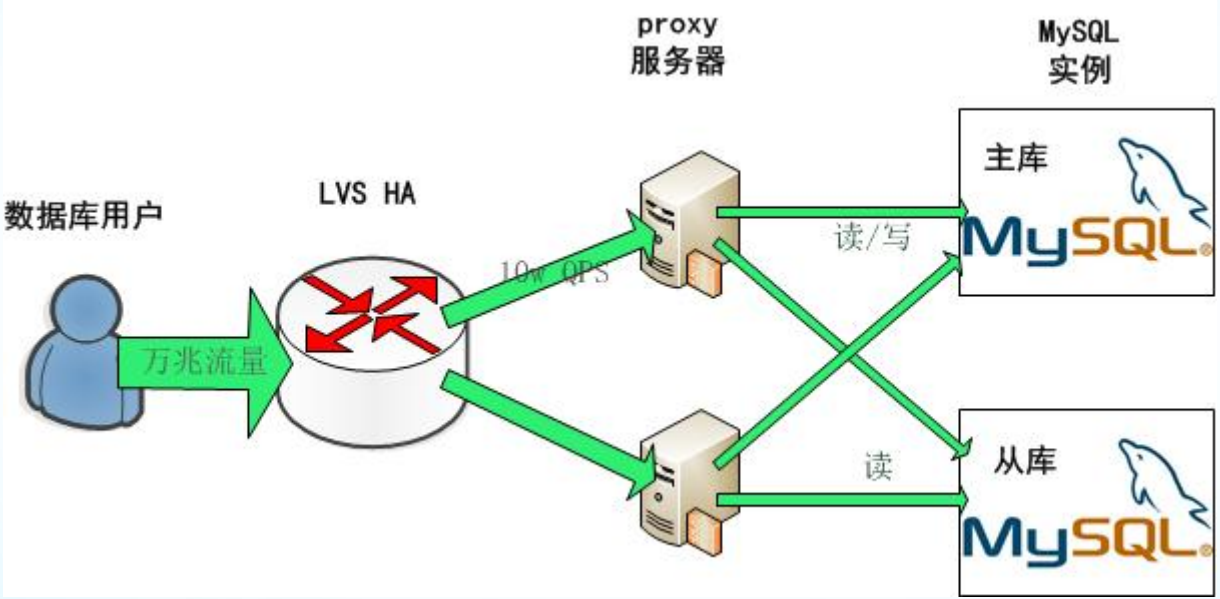


平台挑战和设计原则
平台架构
平台核心部件
讨论

创建数据库实例流程

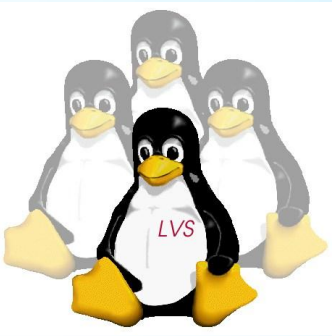

管理员
WeB/API
服务器
Controller
服务器
Leader
Worker
主库agent
服务器
创建主库
MySQL
从库agent
服务器
创建从库
MySQL
http
https
RabbitMQ

数据主路径流程


proxy
服务器
MySQL
实例
数据库用户
LVS HA
10w QPS
万兆流量
读/写
读
主库
从库
MySQL
MySQL

LVS

- 安全隔离
- L4流量切割，实现负载均衡
- Proxy故障自动转移



ZooKeeper

- 作为配置服务器
- 提供分布式锁
- MySQL插件监控所有实例的可用性

RabbitMQ

- Erlang消息机制与AMQP极度吻合
- 系统中各节点间的可靠通信（不包括SQL查询、日志等大数据流的传输）
- 系统各部件广播服务
- 标准的AMQP协议，方便第三方对接和扩展
- 工作流方式消息流动，方便监控

Mnesia分布式数据库

- MySQL NDB出自同门，可靠性长时间验证过
- Mnesia支持分布式事务，也支持脏读写
- 无中心点，带持久内存数据库，数据存取软实时
- 核心元数据全集群可见

分库分表

- 对用户半透明，需要用户协助写Hint
- 目前支持Range和取模二种规则，后端兼容愚公系统
- 目前支持四种DML语句的基本形式，order by和group by
- 从性能考虑，SQL解释、重写、结果集重构用纯C实现。

容灾

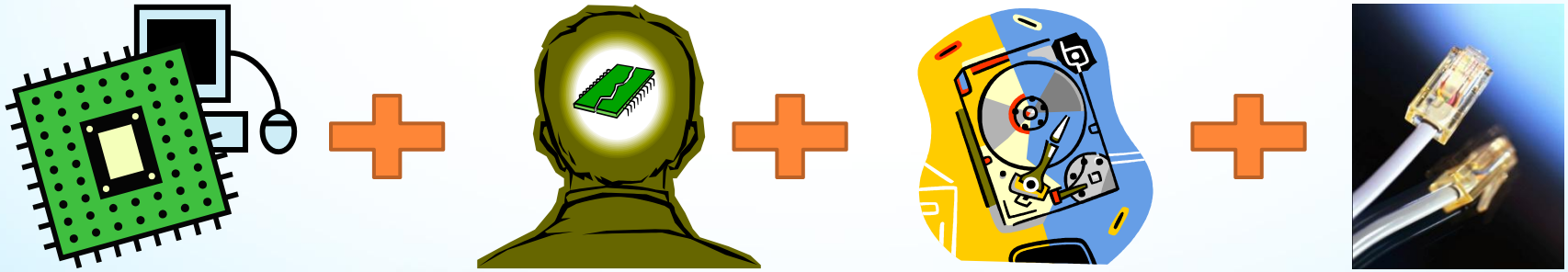
- 对用户透明的故障恢复过程
- 数据服务容灾
  - 数据库主从库的复制置成Dual Master结构
  - proxy通过捕捉错误，延迟重试的方法屏蔽故障
- 平台服务容灾
  - 无单点设计，服务冗余
  - ZooKeeper提供的分布式锁算法选举出一个leader，负责调度和监控各种系统任务
  - 任务中间状态持久化，保证任务可断点续做

读写分离

- 写操作发送到主库，读操作负载均衡到从库，提高从库的利用率
- 何时分离
  - 透过主从是否同步状态判断
  - 时间维度

资源隔离

- 过CGroup的cpuset、memcg以及blkio子模块分别限制用户的MySQL进程最大可以使用的CPU使用率、内存和IOPS
- 对用户SQL执行过程中索引使用情况、IO操作数量等进行分析，指导Proxy增加延迟的方法去限制用户的QPS，达到了减小该用户消耗的系统资源的目的



资源调度

- 用户级别的QOS保证
- 系统自动迁移，倒腾闲置资源

小实例	中等规模	大规模
N : 1	1:1	1:N

数据安全

- 支持SSL连接
- 通过白名单来设置允许访问数据库的IP地址列表
- Proxy会把用户所有的数据库操作记录到日志分析服务器，扫描检查安全漏洞
- Proxy可以根据安全部门的要求拦截各种类型的SQL语句
- 误操作删除数据又没有备份的情况可以通过Flashback工具恢复数据
- …

下一阶段重点

- 提高各部件特别是proxy的效率，降低平台自身的成本消耗
- 资源隔离和调度完善, 进一步降低成本
- 和已有系统的融合，提高易用性

Q&A

http://yufeng.info

新浪微博：@淘宝褚霸