物联网分布式接入服务

百度开放云-----胡云

提纲

- 物联网背景以及挑战
- 接入协议比较简介
- 分布式MQTT架构
- Router介绍
- 分布式Session
- 服务高可用性以及数据高可靠性
- 数据一致性模型
- 开发过程中遇到问题

IoT背景

a 很早就有IoT了

> 40多年了很多公司已经在这个领域耕耘

IoT已近在传统领域被广泛应用比如监控报警

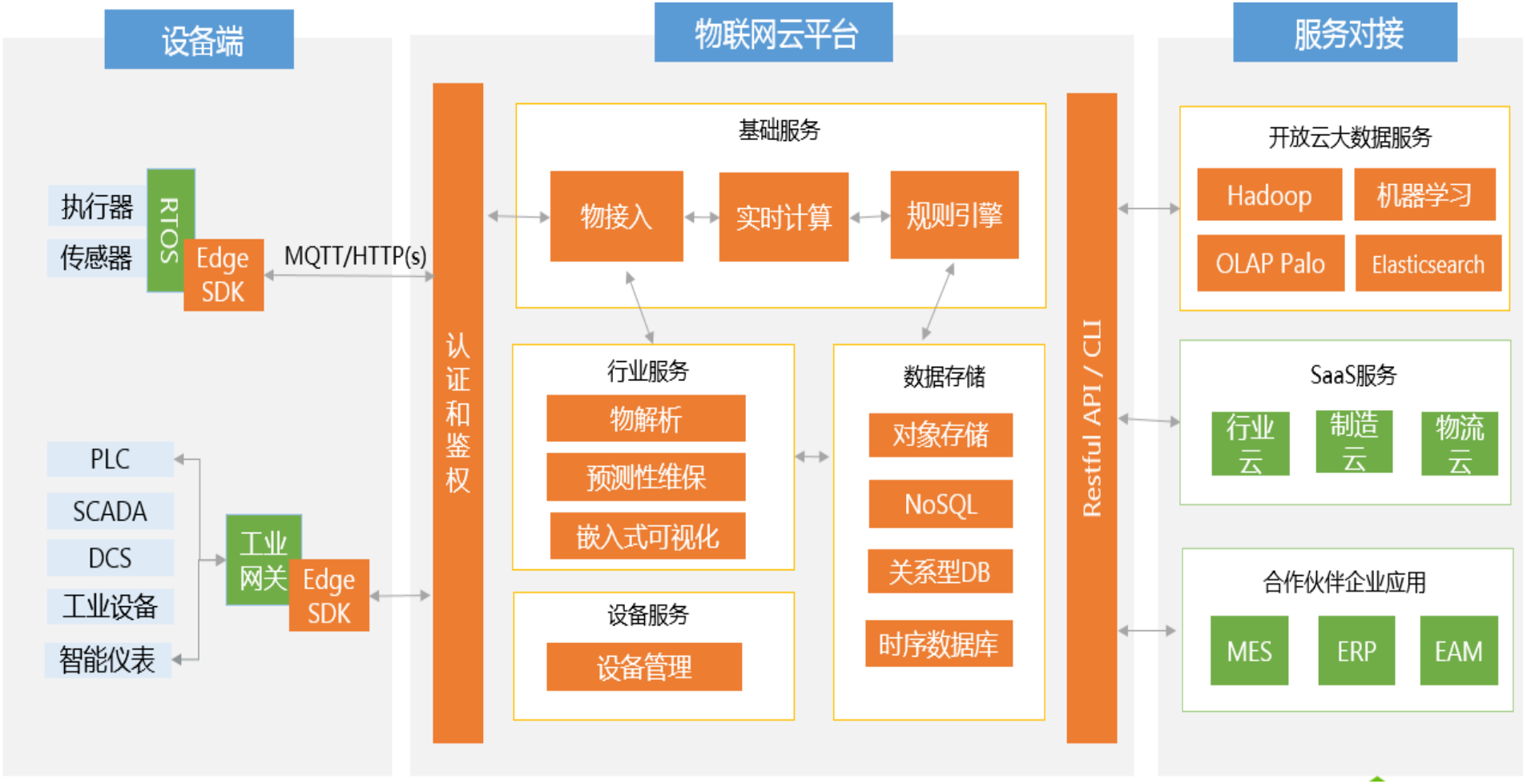
b 一个新型领域

随着硬件设备的成本大幅下降，IoT可以被应用到很多日常生活场景，产生大量设备连进网络，应用更加丰富

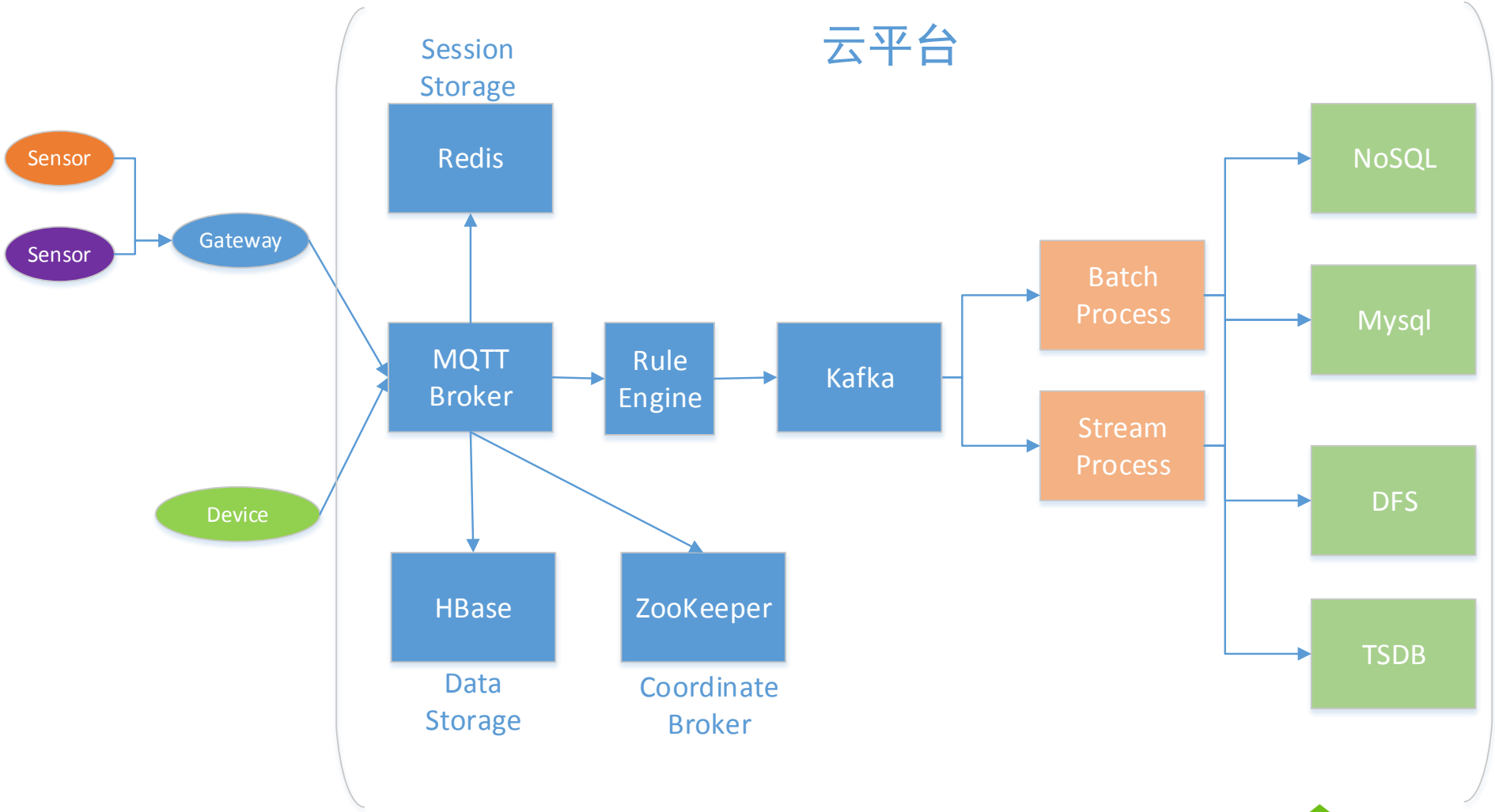
IoT挑战

- **规模**
  - 未来连接物联网的设备将要达到数100亿级别
  - 比如可穿戴式设备，家庭智能设备，工业网关
- **存储**
  - 分布式的时序数据库，不同NoSQL，比如HBase, Redis, Cassandra
  - 分布式文件系统，传统的RDS
- **工具**
  - 大数据分析工具，实时流式处理
  - 分析设备产生数据的机器学习平台，训练模型
- **环境**
  - 安全以及个人隐私:防止攻击，数据安全，数据加密
  - 不同协议的融合 （MQTT， CoAP， HTTP， XMPP）

百度物联网平台


设备端
物联网云平台
服务对接
执行器
传感器
RTOS
Edge SDK
MQTT/HTTP(s)
PLC
SCADA
DCS
工业设备
智能仪表
工业网关
Edge SDK
认证和鉴权
基础服务
物接入
实时计算
规则引擎
行业服务
物解析
预测性维保
嵌入式可视化
设备服务
设备管理
数据存储
对象存储
NoSQL
关系型DB
时序数据库
Restful API / CLI
开放云大数据服务
Hadoop
机器学习
OLAP Palo
Elasticsearch
SaaS服务
行业云
制造云
物流云
合作伙伴企业应用
MES
ERP
EAM

物联网平台架构



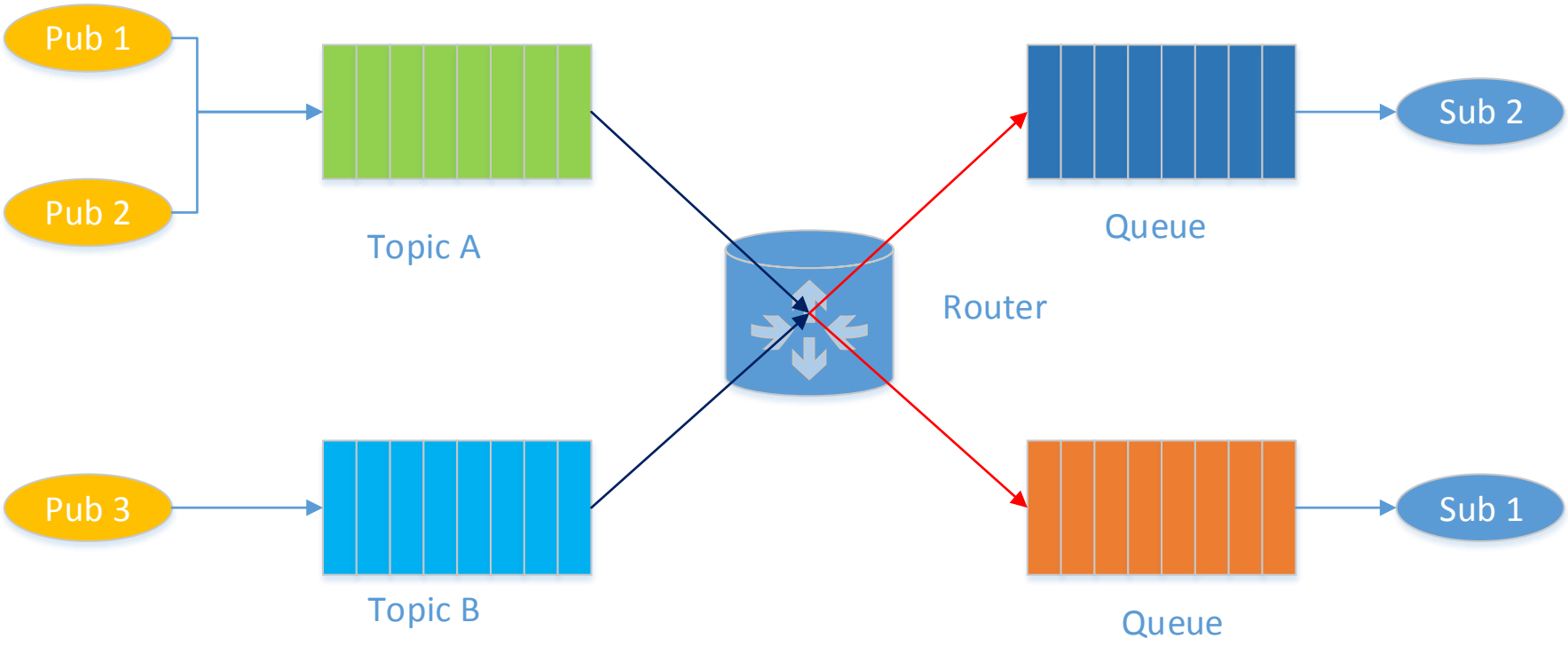
为什么选择MQTT协议

	MQTT	CoAP	XMPP	HTTP
传输方式	TCP	UDP	TCP	TCP
消息模型	发布/订阅	请求/响应	发布/订阅	请求/响应
时效性	好	差	好	差
反控	好	差	好	差
传输效率	高	高	一般	一般
功耗	一般	小	高	高
QoS支持	是	否	是	否

MQTT协议特点

- **维持活跃消息（发送heartbeat来维持连接）**
  - 可以用来检测client是和server之间连接状态，同时还可以维护NAT地址映射表
- **遗愿消息**
  - 指定当client和server之间失去连接之后，server需要往指定的topic发送特定QoS的级别的消息
- **保留消息**
  - 一个特定主题的消息会保留到服务器，当任何一个client订阅了这个主题，都会首先收到这个消息
- **持久化订阅**
  - 当client和server之间断开连接之后，所有发往这个主题的消息都会保留下来，等client重新连接之后，会重新发给该client

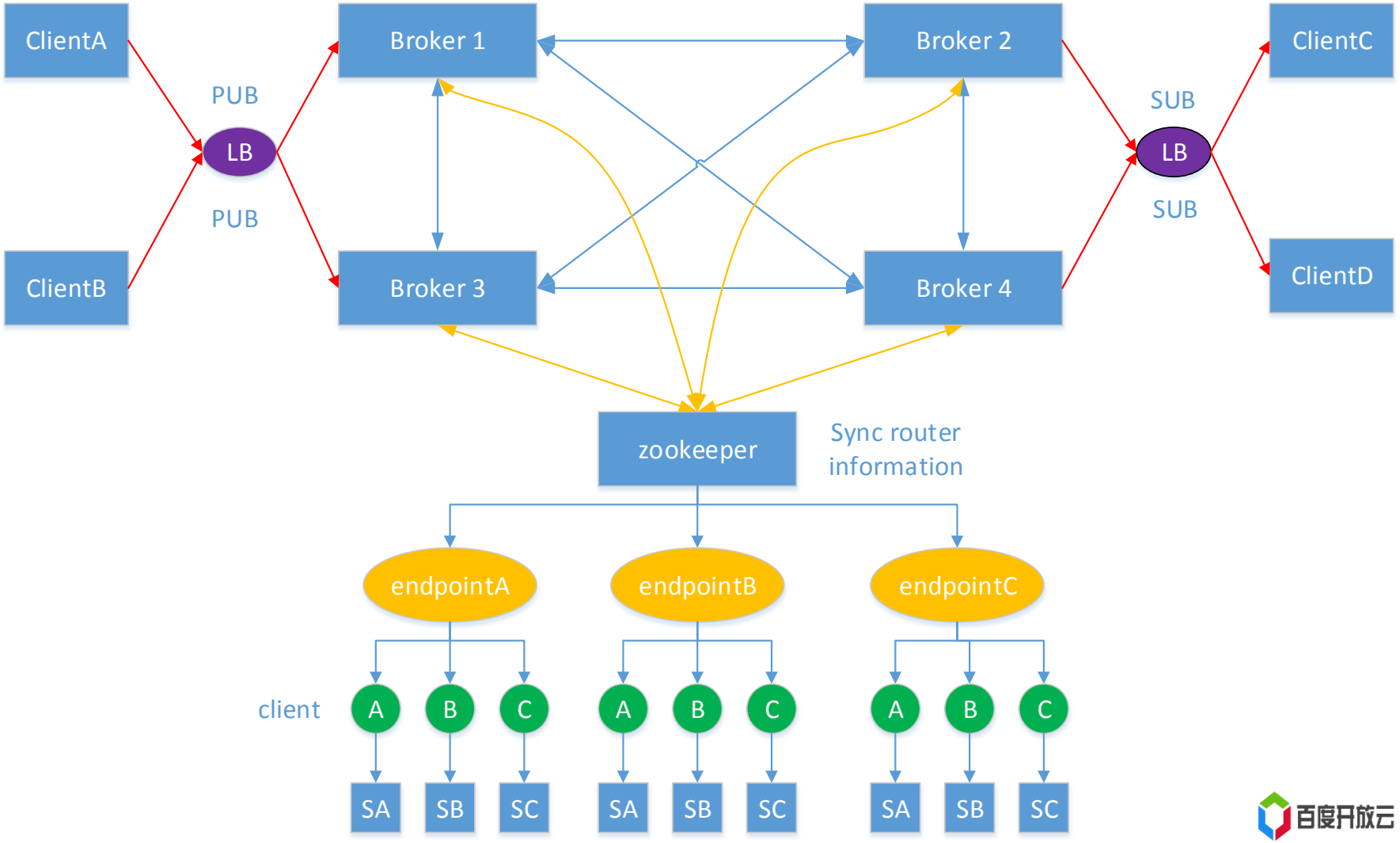
MQTT PUB/SUB系统



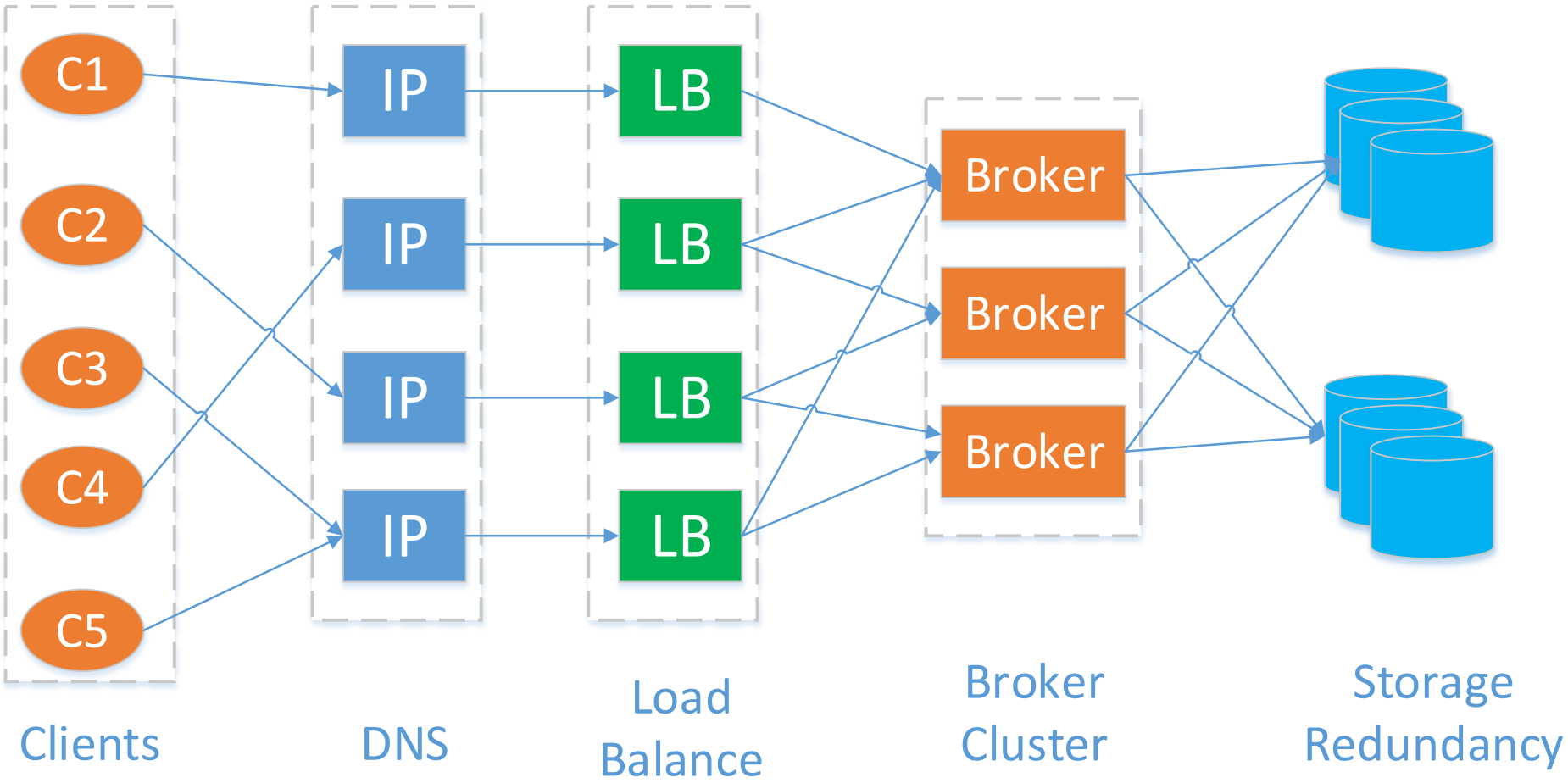
开源的MQTT服务

- 单机版本
  - Mosquitto
  - Moquette
  - Apollo
  - RabbitMQ
- 分布式
  - EMQTT

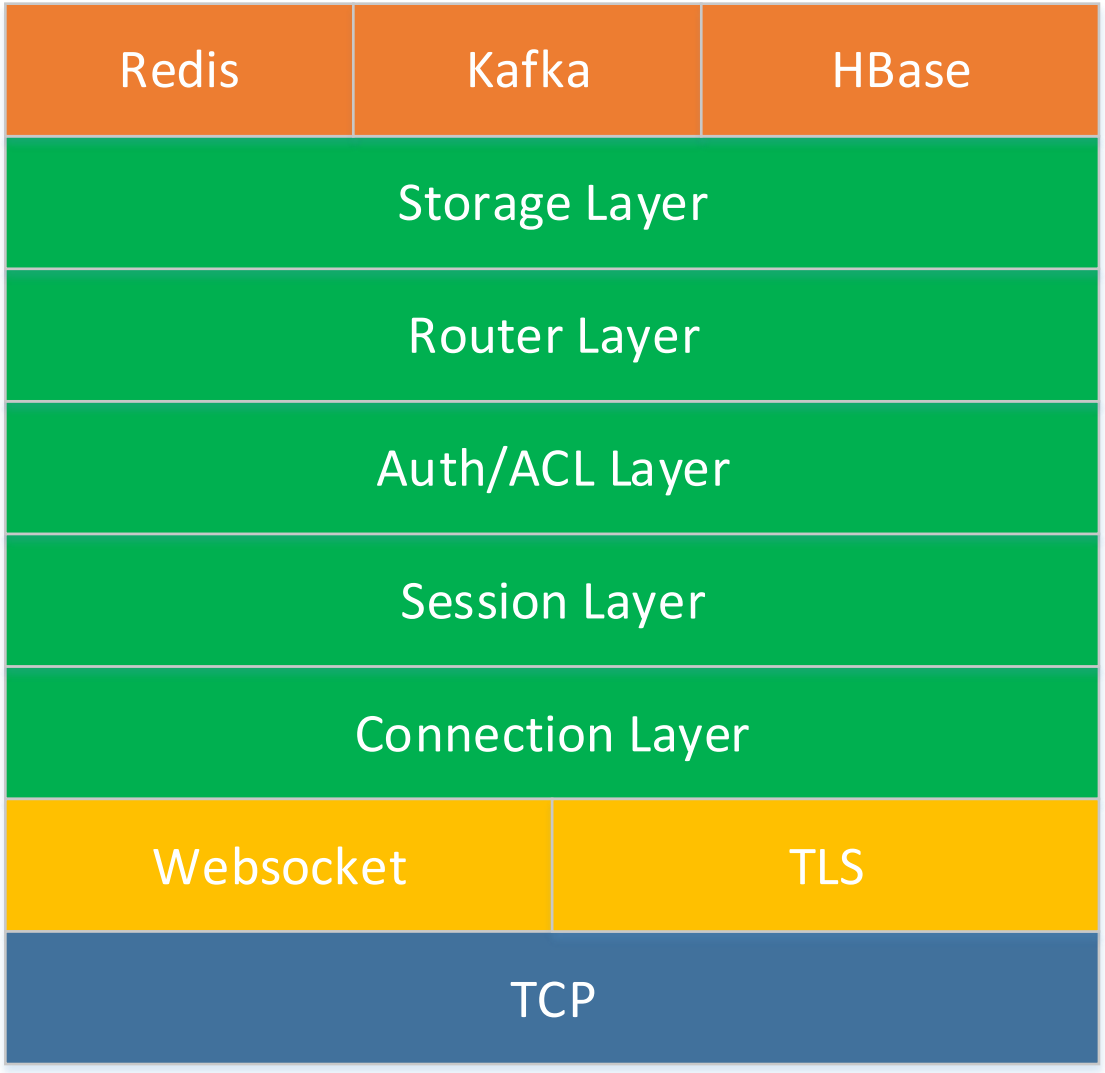
分布式MQTT架构



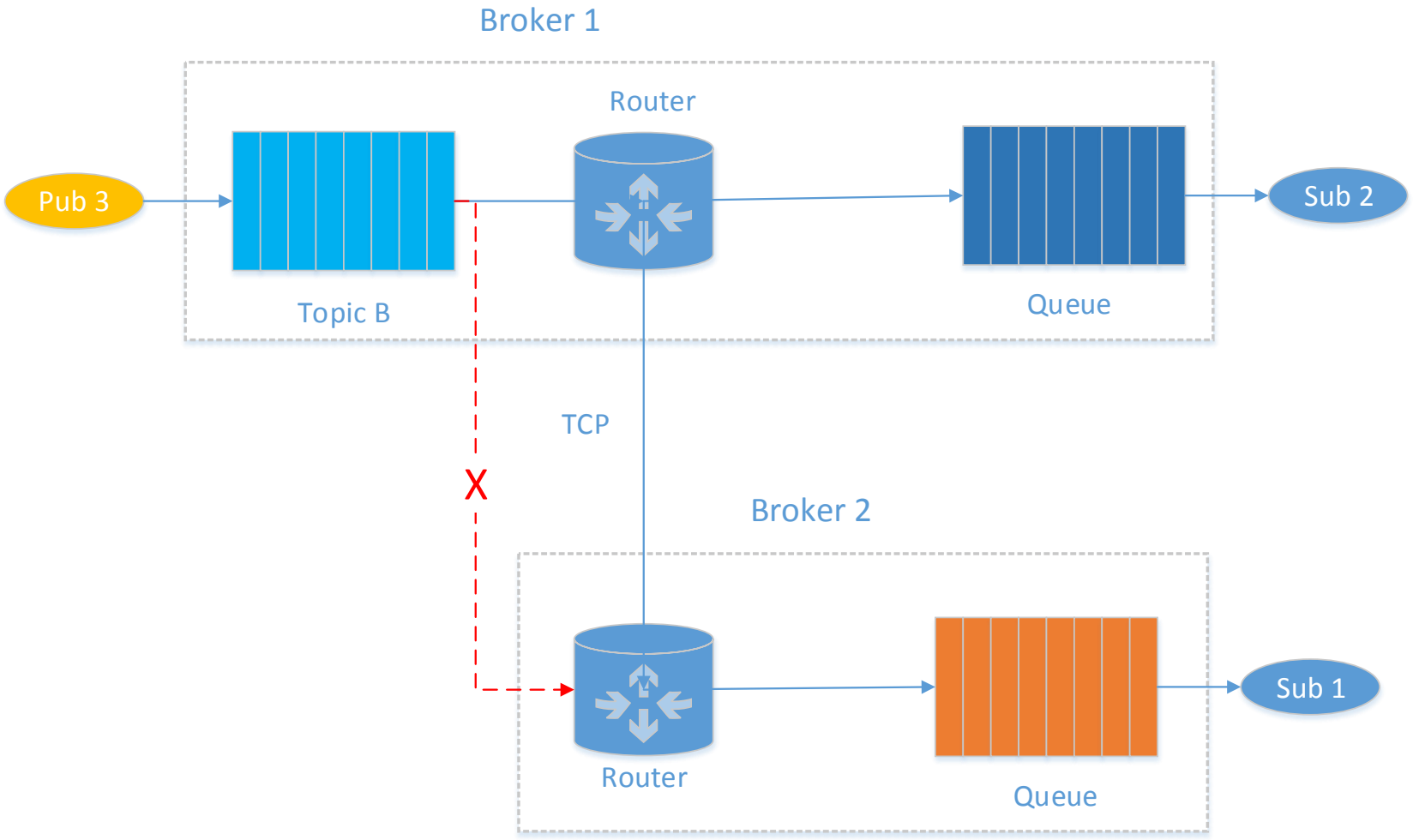
HA & HR


C1
C2
C3
C4
C5
IP
IP
IP
IP
LB
LB
LB
LB
Broker
Broker
Broker
Clients
DNS
Load Balance
Broker Cluster
Storage Redundancy

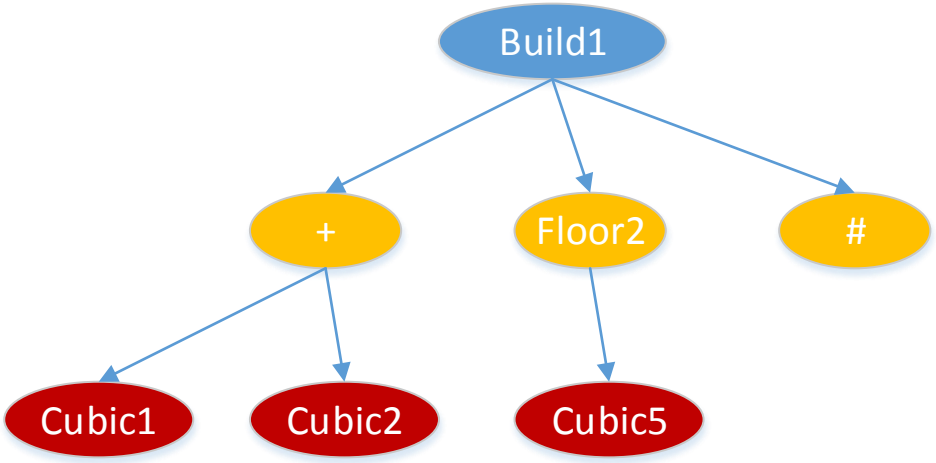
MQTT层堆栈



跨机器Router模型

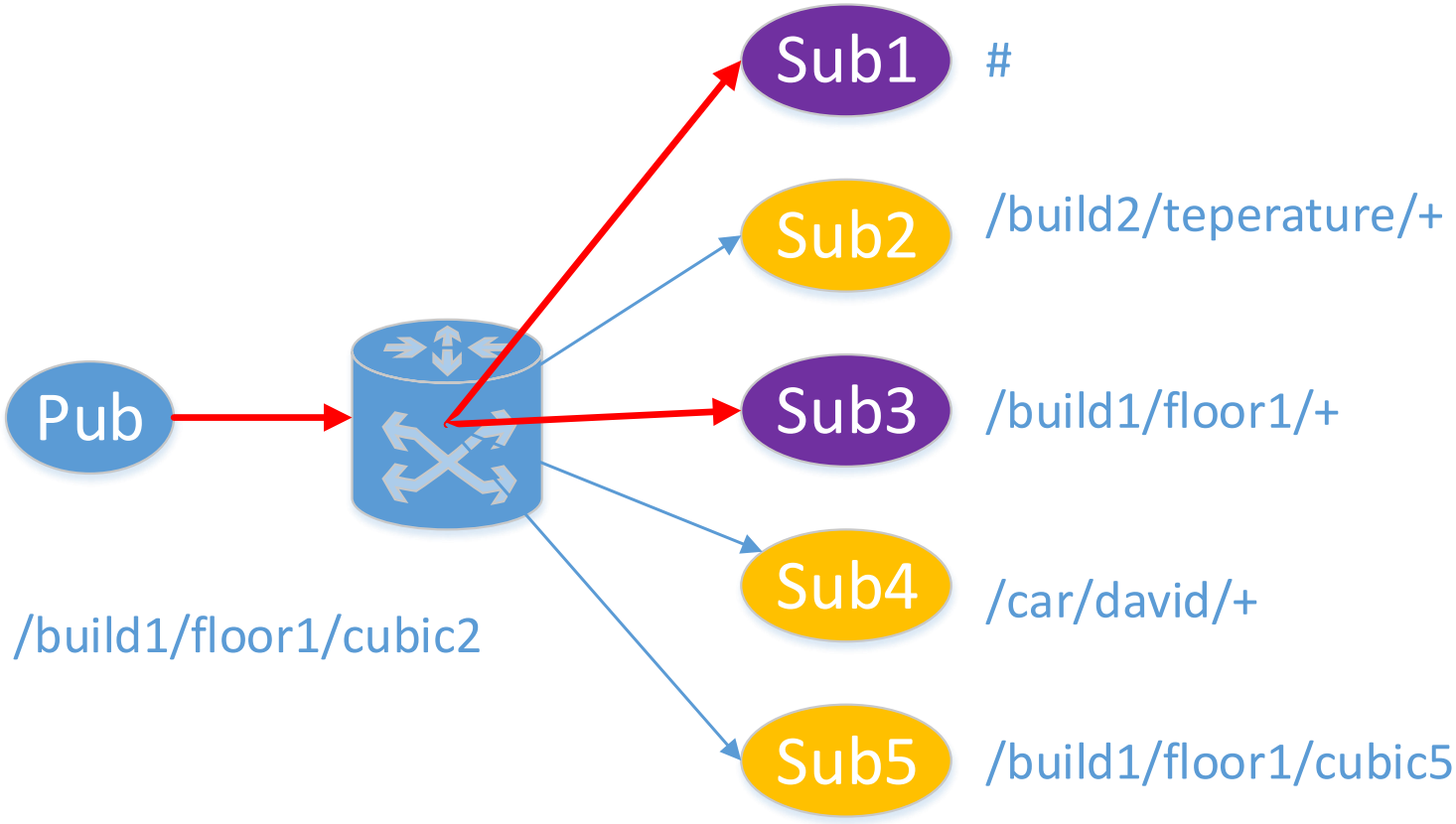


Router性能优化(字典树)



- **Build1/#**
  - 匹配任何以Build1/开头的主题
  - 例如：Build1/floor1/Cubic4, Build1/Floor3
- **Build1/+/Cubic1**
  - 匹配以build1开始，Cubic1结束的3级任何主题
  - 例如：Build1/floor5/Cubic1
- **Build1/Floor2/Cubic5**
  - 就是完全匹配，中间不包含任何通配符

Router性能优化(Cache)



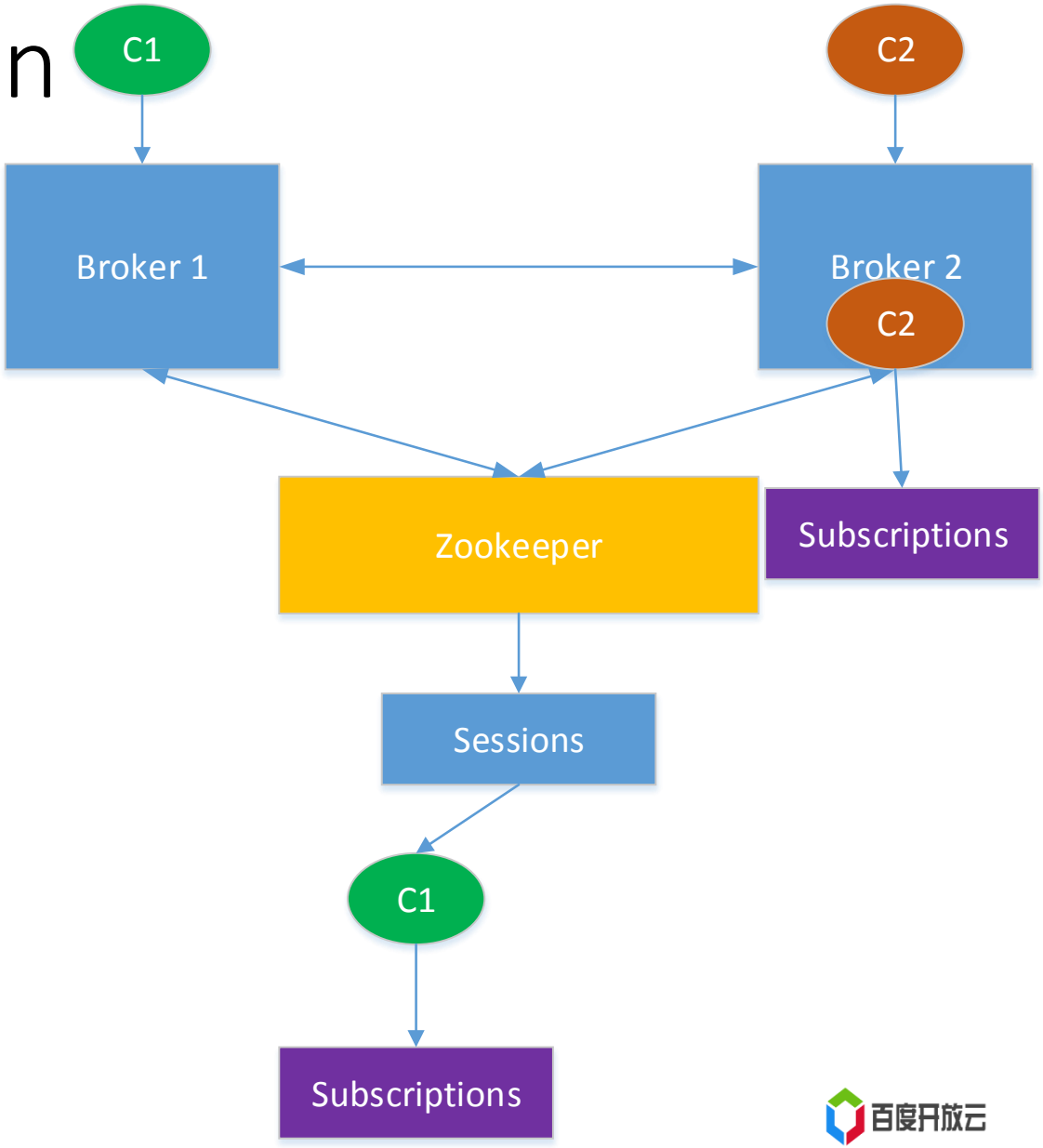
缓存经常发布消息主题的路由表

当前架构的性能指标（单机）

- 并发连接
  - 100w - 200w
- 消息延迟
  - 小于1s
- 最大扇出数
  - 2w - 3w
- 每秒消息处理数目
  - 4w（qps）

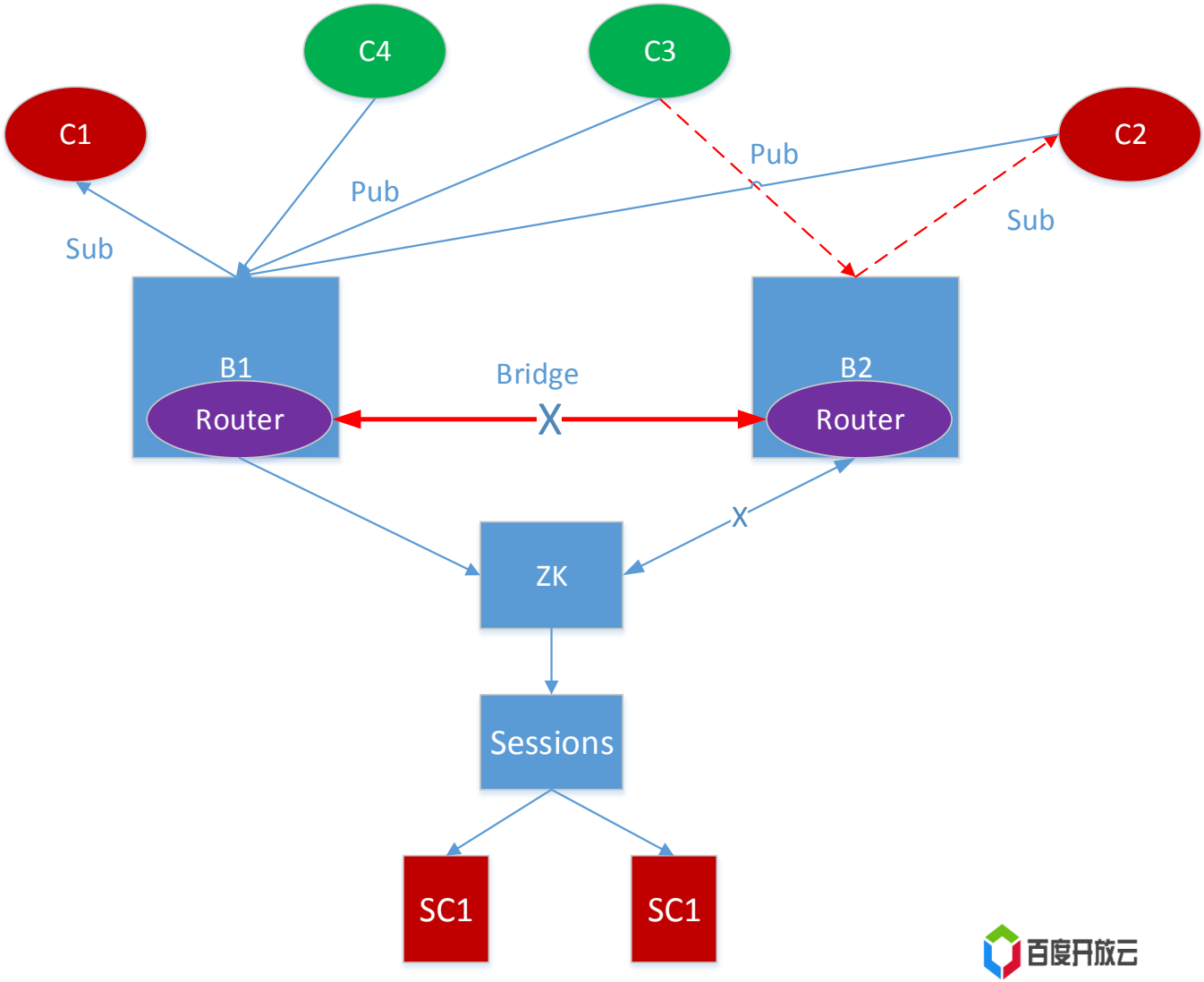
分布式session

- 易逝session
  - 断开连接之后session就会被清除
  - 订阅主题信息会删除
- 持久化Session
  - 即使client断开连接，session还是会保留
  - 订阅主题仍然保留，数据仍然会写到该client的数据队列



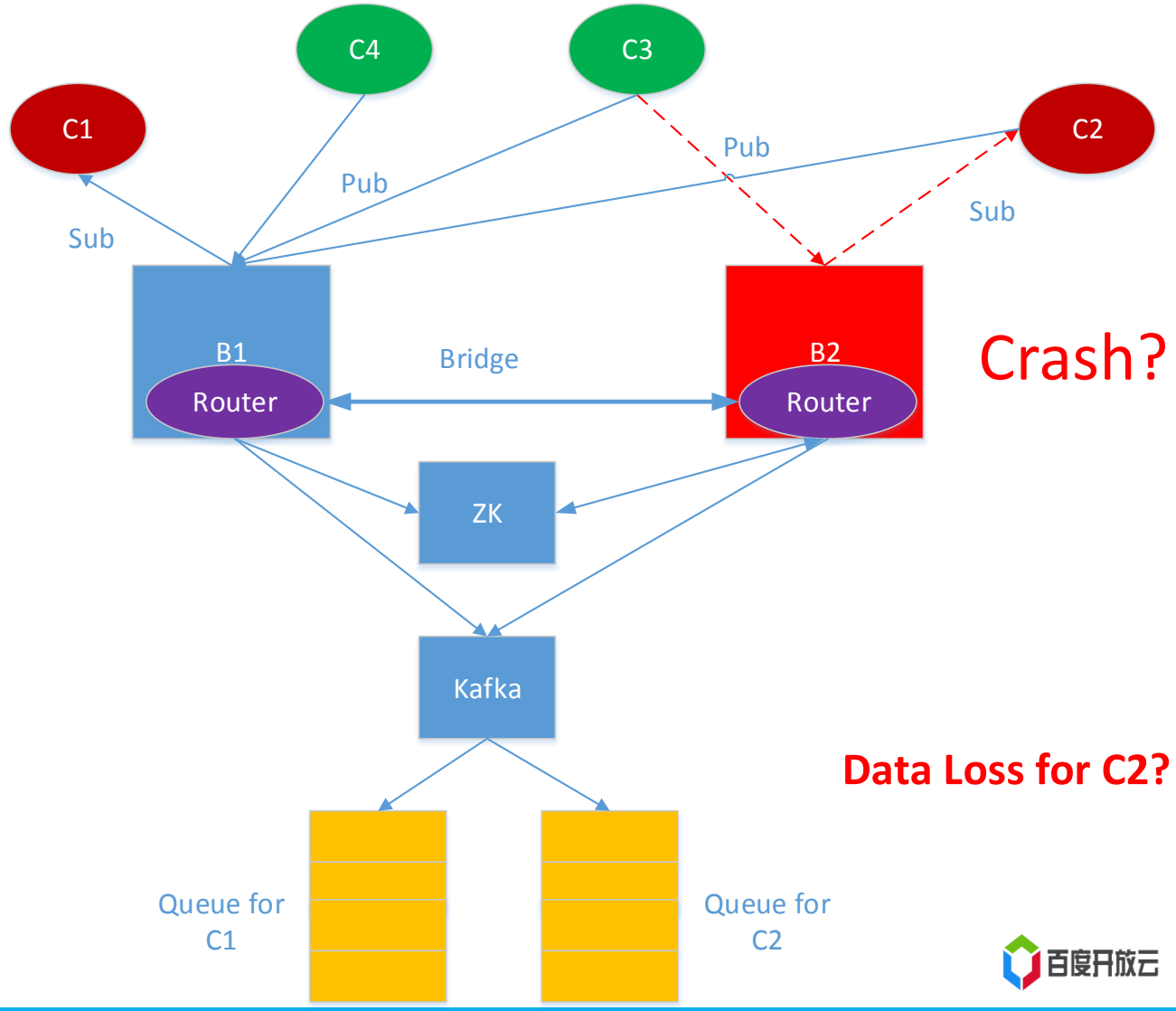
服务高可用性

Broker和ZK以及其他Broker都失去连接



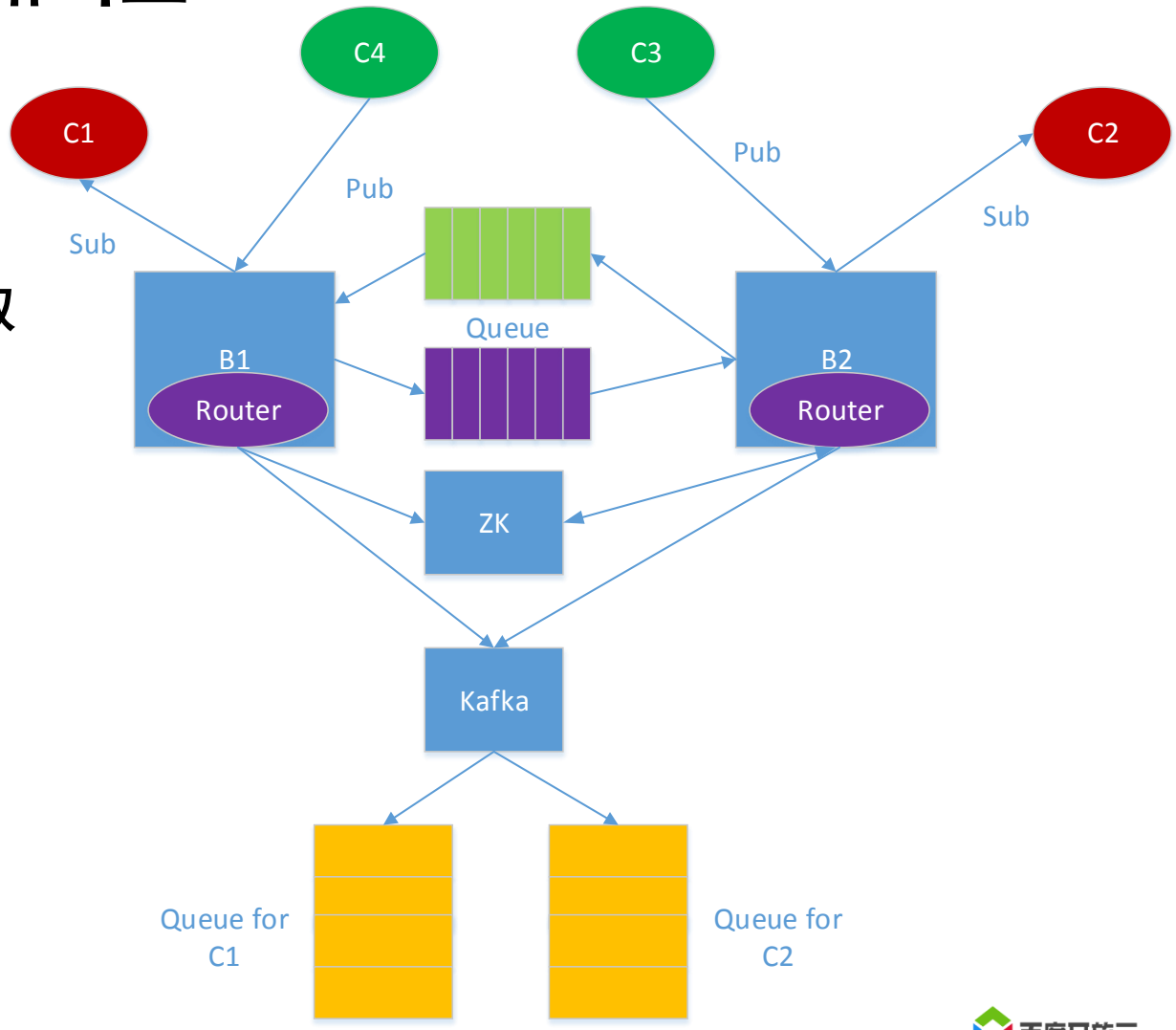
Failover数据丢失?

持久化C2丢失数据的问题?



数据高可靠性

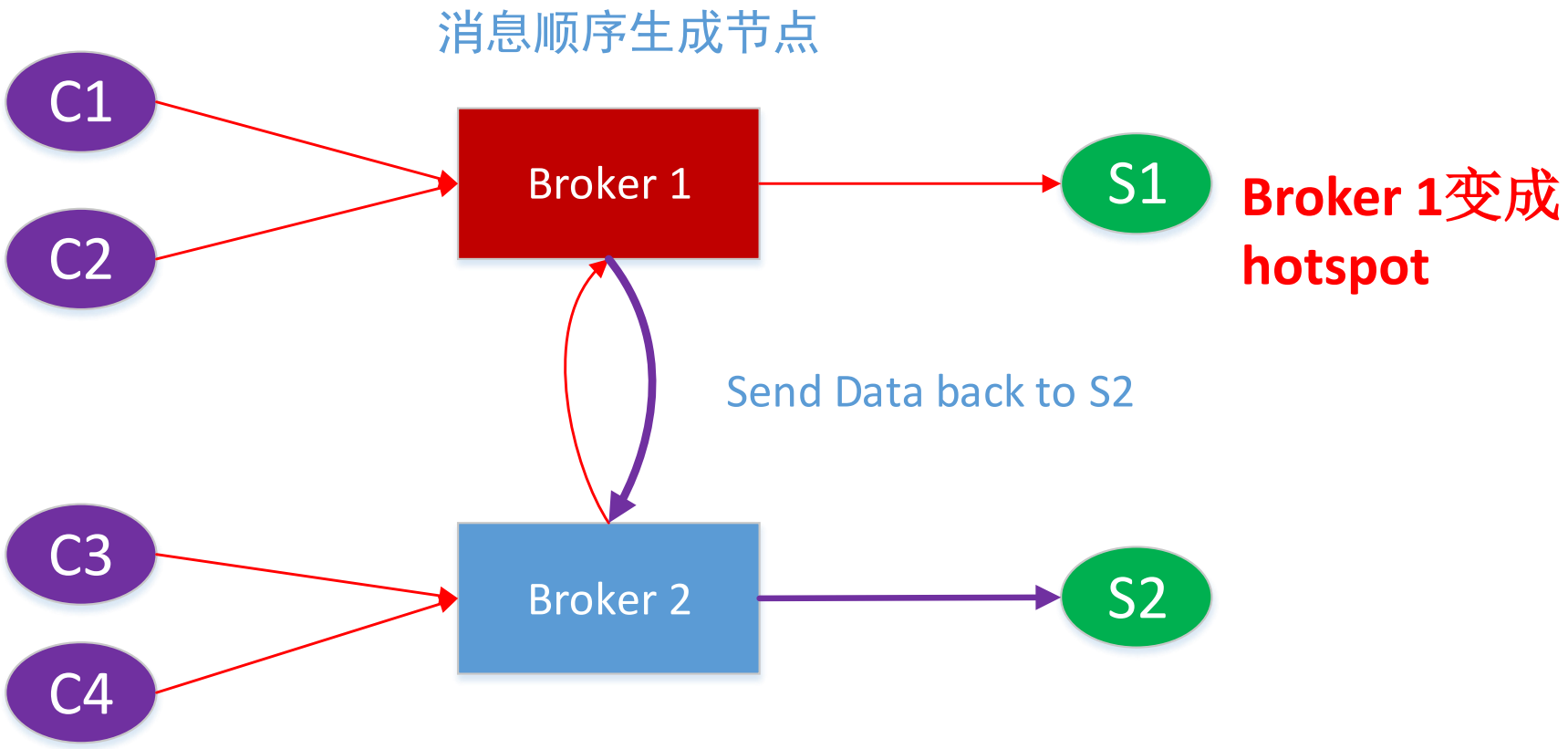
数据交换采用先存储在读取



后端数据存储可配置

1. Kafka
2. Redis
3. HBase

全局数据一致性？



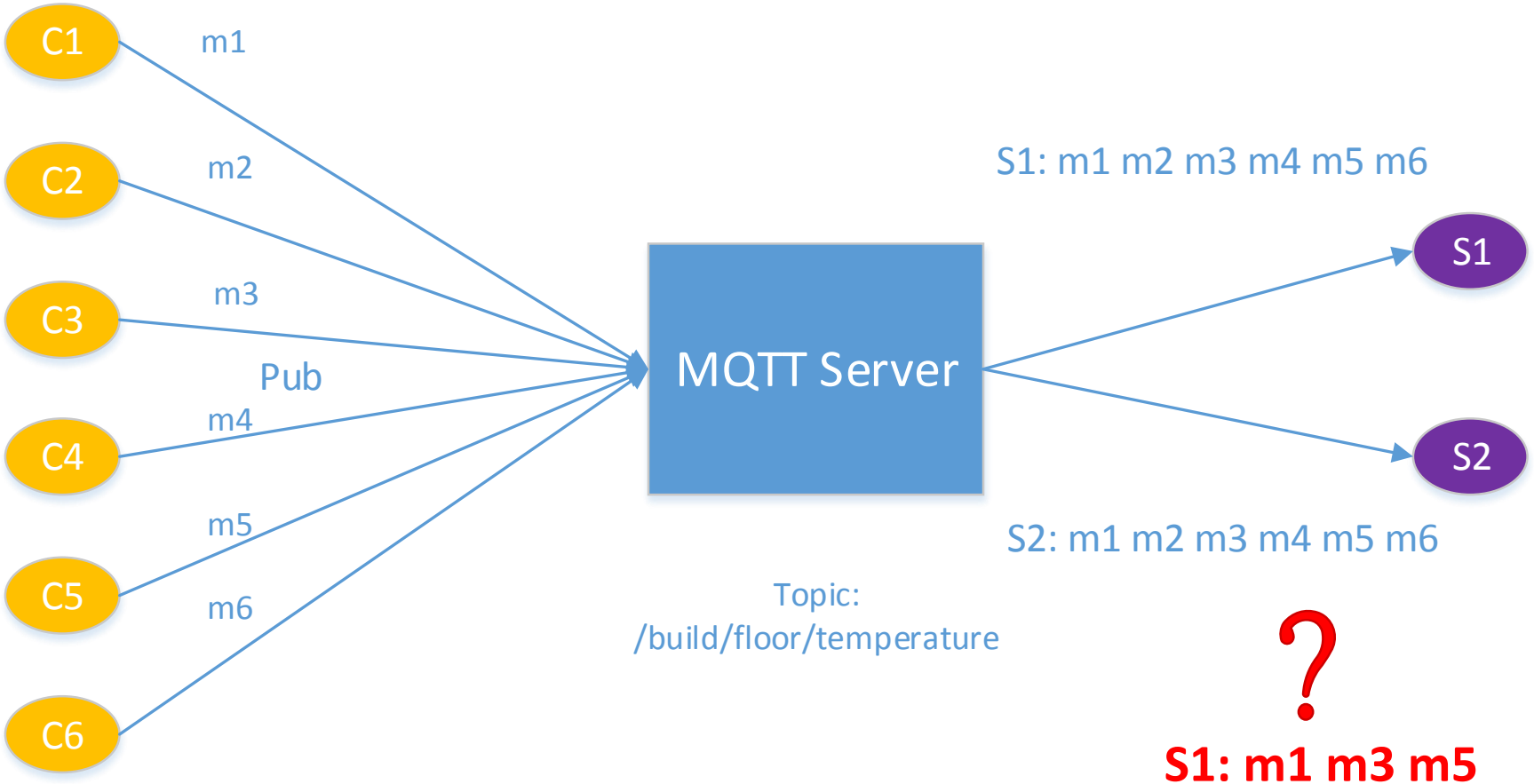
数据一致性

- **支持同一client发布消息顺序性**
  - 保证每个订阅者收到来自同一个broker上发送到同一个topic消息的有序性
  - 不同Broker发送消息顺序取决于他们发送到这个broker的时间决定的
- **不支持各个client之间消息全局有序**
  - 这样就需要选择一个master来分配每个消息的ID
  - 对于MQTT大部分场景来说，没必要全局有序

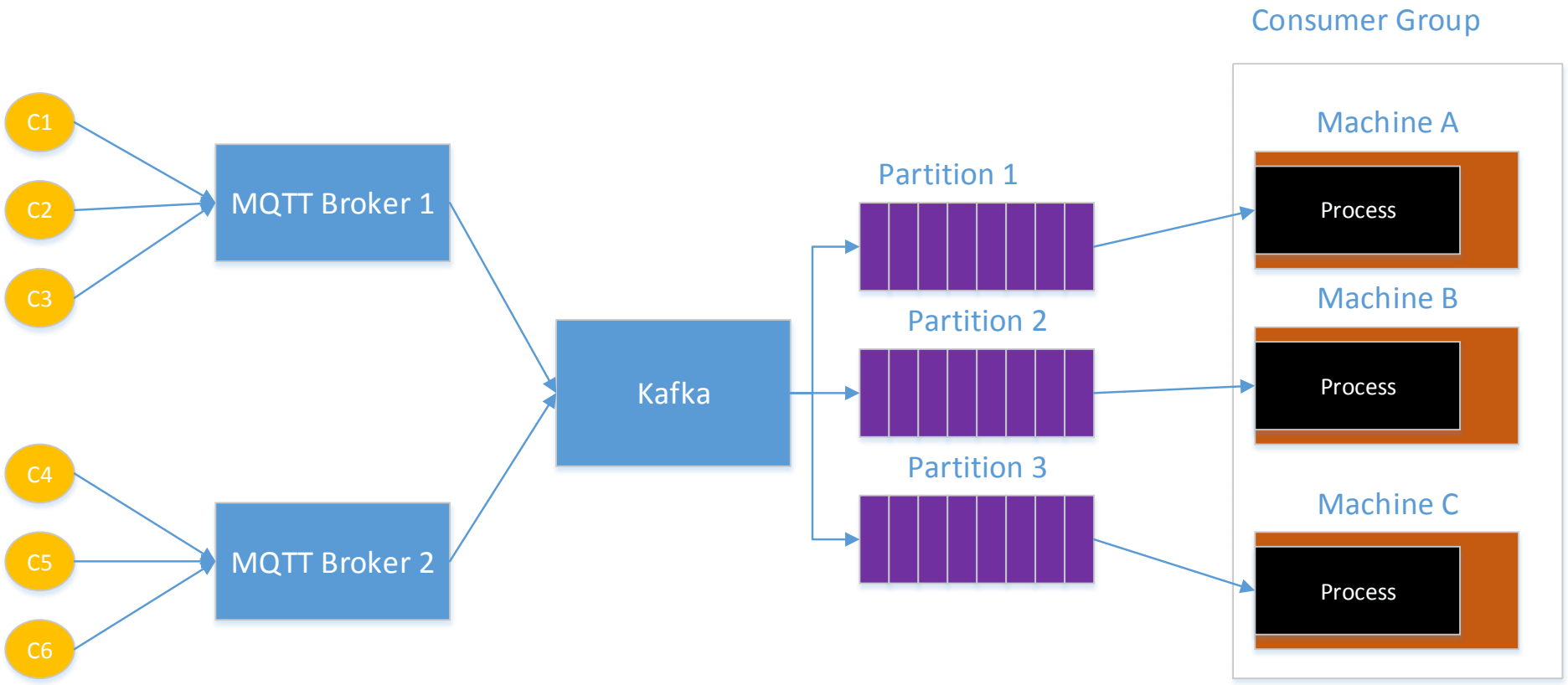
数据一致范例

- **C1 发送消息到 M1，M2，M3 主题 T2**
- **C2 发送消息到 M4，M5，M6 主题 T2**
  - M1必须在M2，M3之前到达T2
  - M2必须在M3之前到达T2
  - M4必须在M5，M6之前到达T2
  - M5必须在M6之前到达T2
  - T2收到的消息可能是C1和C2发送消息交织的结果

MQTT订阅协议的局限



大数据并发处理架构



开发过程中遇到问题

- **采用REST API方式做authentication & Authorization**
  - 由于是短链接导致大量TIME_WAIT状态的TCP连接，消耗太多端口资源
  - 采用RPC连接池极大降低短链接导致端口资源消耗，采用cache机制
- **跨机器消息传递瓶颈问题**
- **建议大规模数据订阅采用从kafka消费数据**
    - Broker直接将消息写入到Kafka，从kafka订阅数据
- **避免使用Zookeeper管理大量metadata以及watch**
  - 会导致zookeeper系统达到资源极限，比如最大watch的节点数太多会导致session重建出现失败

MQTT服务接入

- **接入语言支持**
  - C/C++
  - Java
  - PHP
  - Python
  - C#
  - JavaScript
- **接入方式**
  - TCP / TLS
  - HTTP / TLS
  - WebSocket TLS

- **支持设备**
  - Arduino
  - Raspberry Pi
  - MTK
  - Beagle Bone
  - Edison

应用场景

- 工业4.0
- 零售020
- 智慧物流
- 节能减排
- 智能硬件
- 车联网

Q & A

Thanks