L07-线程安全-分布式锁

JAVA并发编程

课程目标

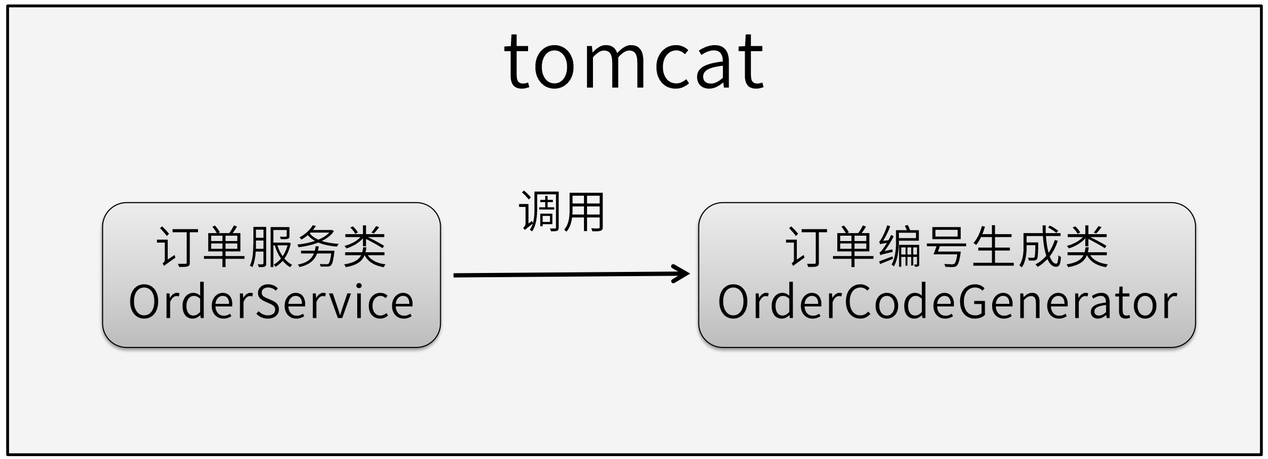
- 掌握分布式锁的应用场景
- 掌握分布式锁的原理、实现方式
- 掌握基于zookeeper的分布式锁的实现

为什么需要分布式锁

场景描述

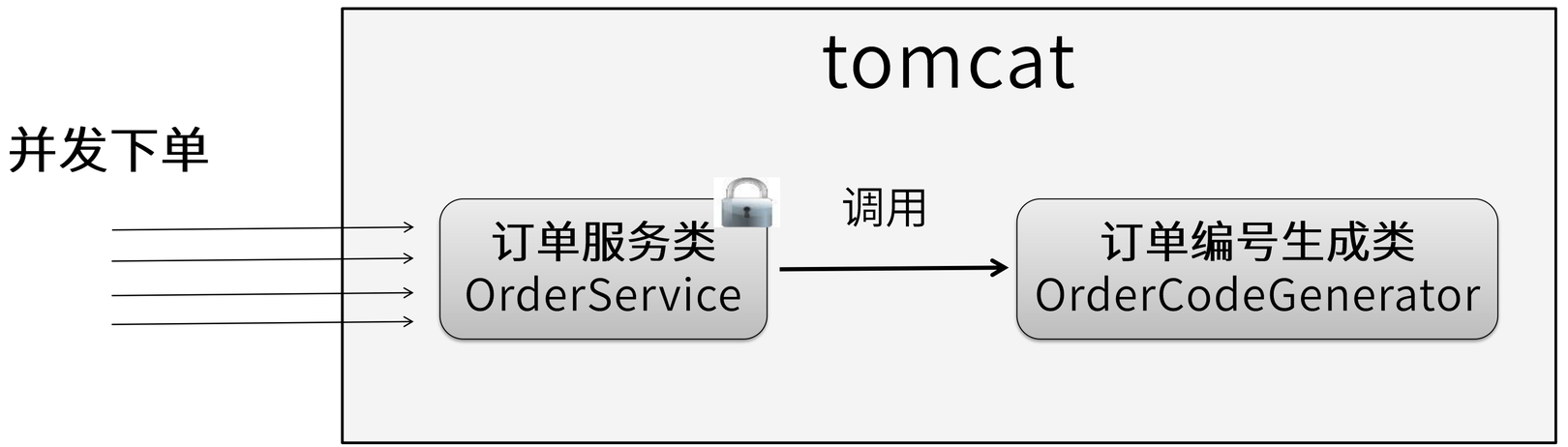
- 小型电商网站，下单，生产有一定业务含义的唯一订单编号

如：2017-05-15-20-52-33-01
年 月 日 时 分 秒 序号



场景描述

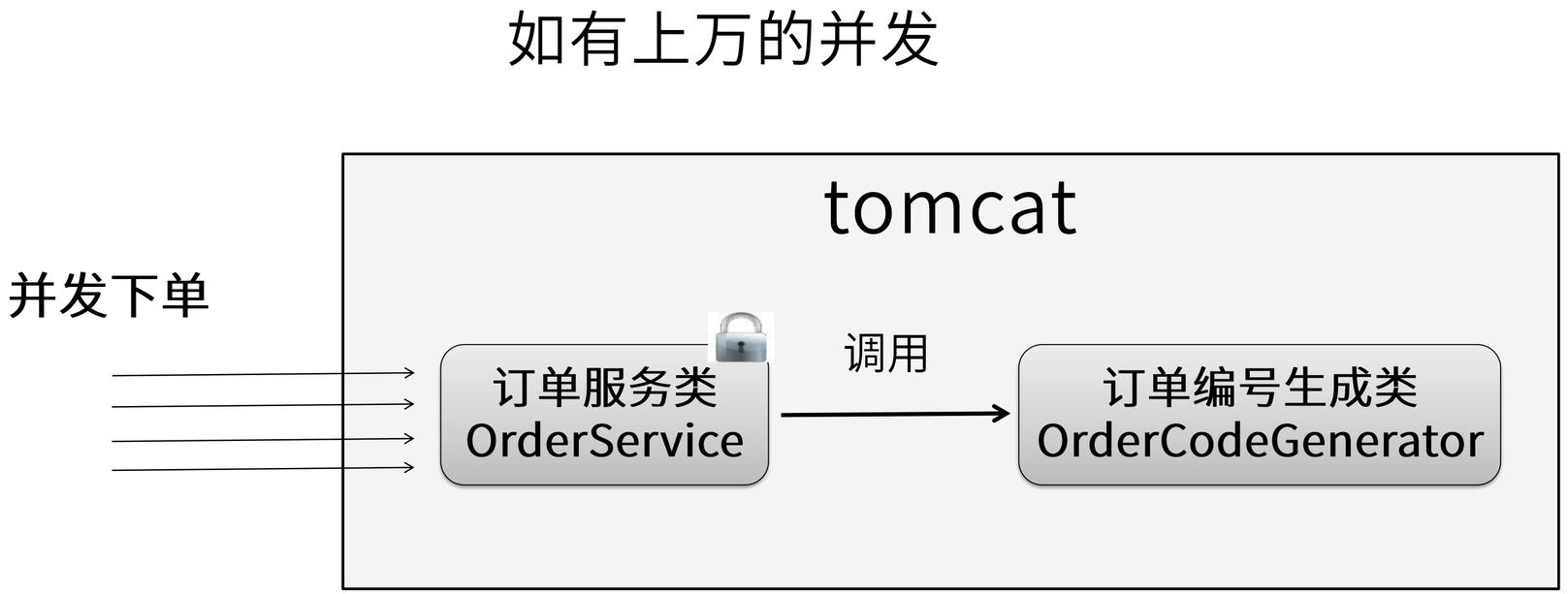
- 如果同时有很多人下单....



此时订单号能做到唯一吗?

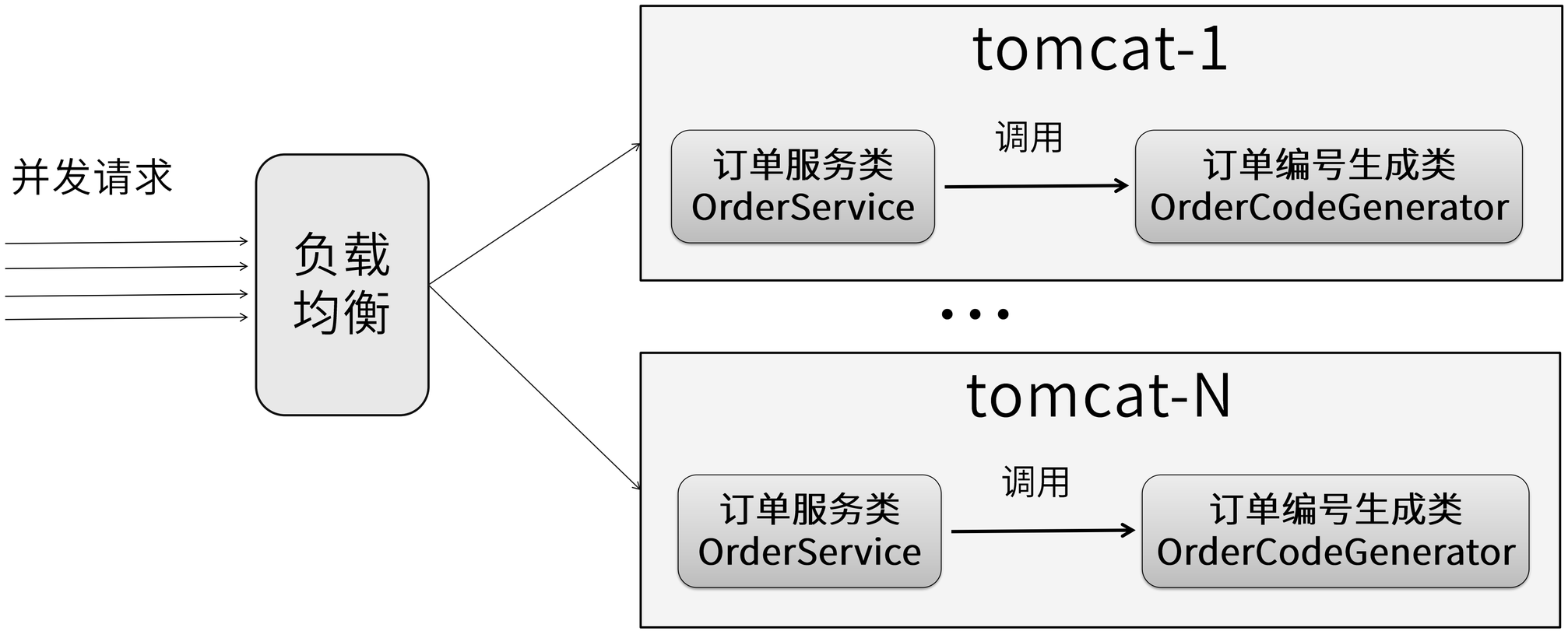
场景描述

- 如果单台服务器已无法撑起并发量…… 怎么办?



场景描述

- 如果单台服务器已无法撑起并发量…… 怎么办？ 集群



此时订单号能做到唯一吗？

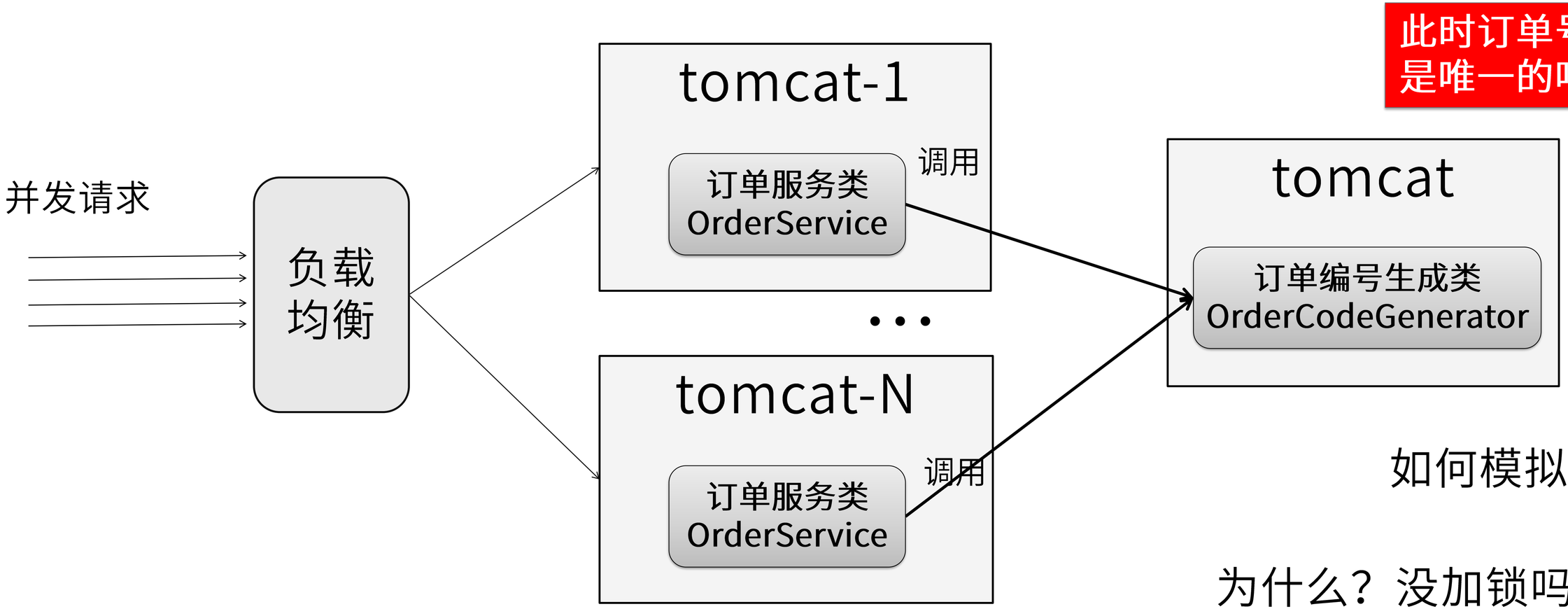
如何模拟验证？

原因是什么？

怎么办？

场景描述

- 将订单编号生成独立为共享的服务

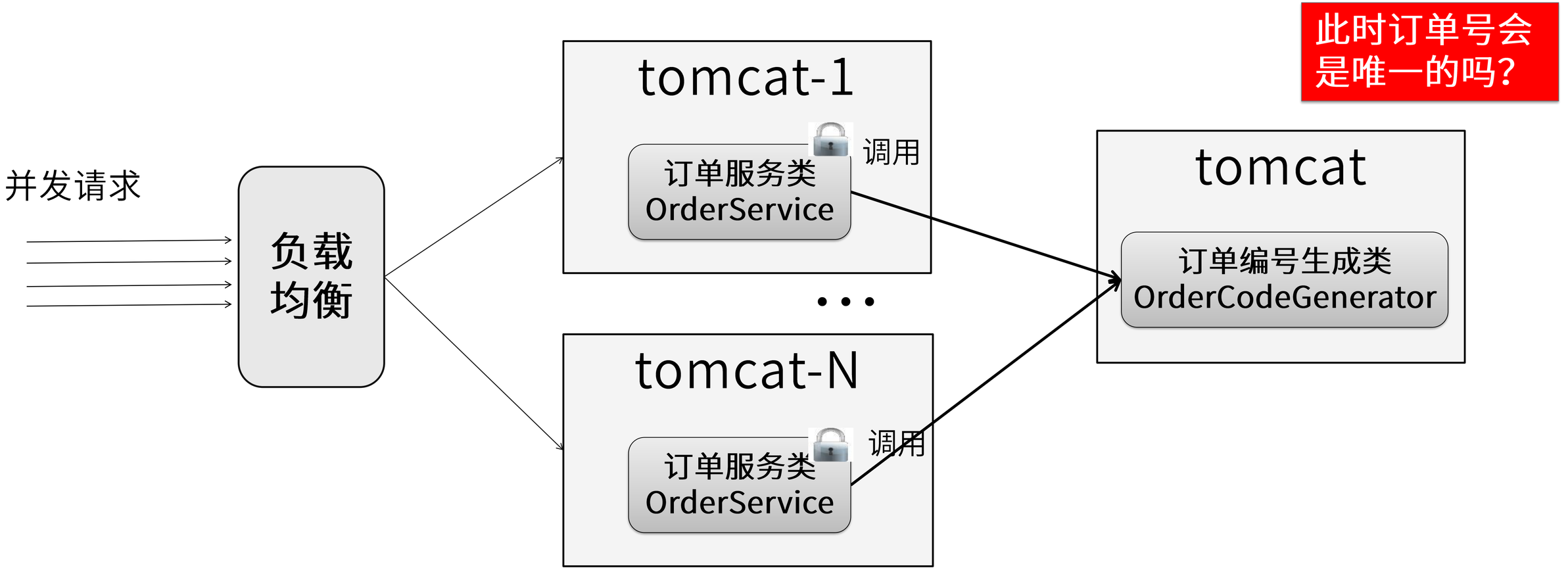


如何模拟验证?

为什么?没加锁吗?

场景描述

- 将订单编号生成独立为共享的服务

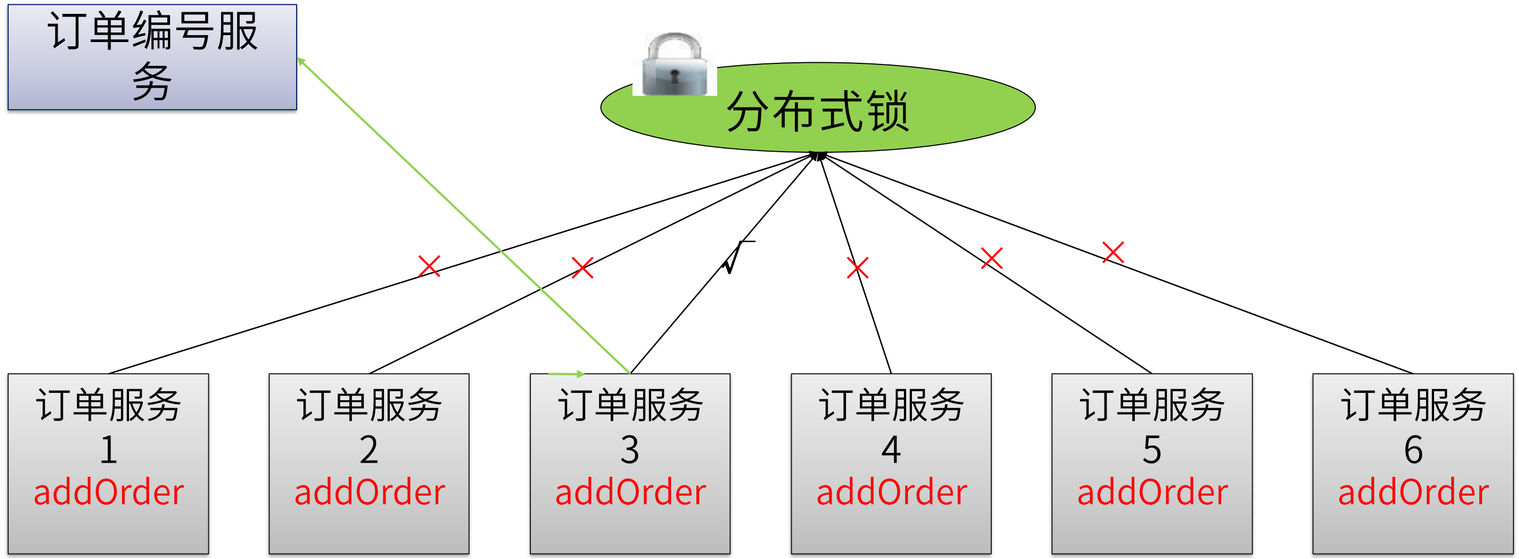


虽然有锁，但各tomcat中使用的是各自的锁，所以订单号做不到唯一。

使用jvm锁能行吗?

场景描述

- 此时就需要用到分布式锁了



分布式锁的用途：在分布式环境下协同共享资源的使用。

分布式锁的实现方式有哪些

分布式锁思路分析

锁具有什么特点?

- 排他性：只有一个线程能获取到
- 阻塞性：其他未抢到的线程阻塞，直到锁释放出来，再抢。
- 可重入性：线程获得锁后，后续是否可重复获取该锁。

分布式锁思路分析

- **我们掌握的计算机技术中，有哪些能提供排他性？**
  - 文件系统
  - 数据库：　主键 唯一约束　for update
  - 缓存 redis：　setnx
  - zookeeper　类似文件系统

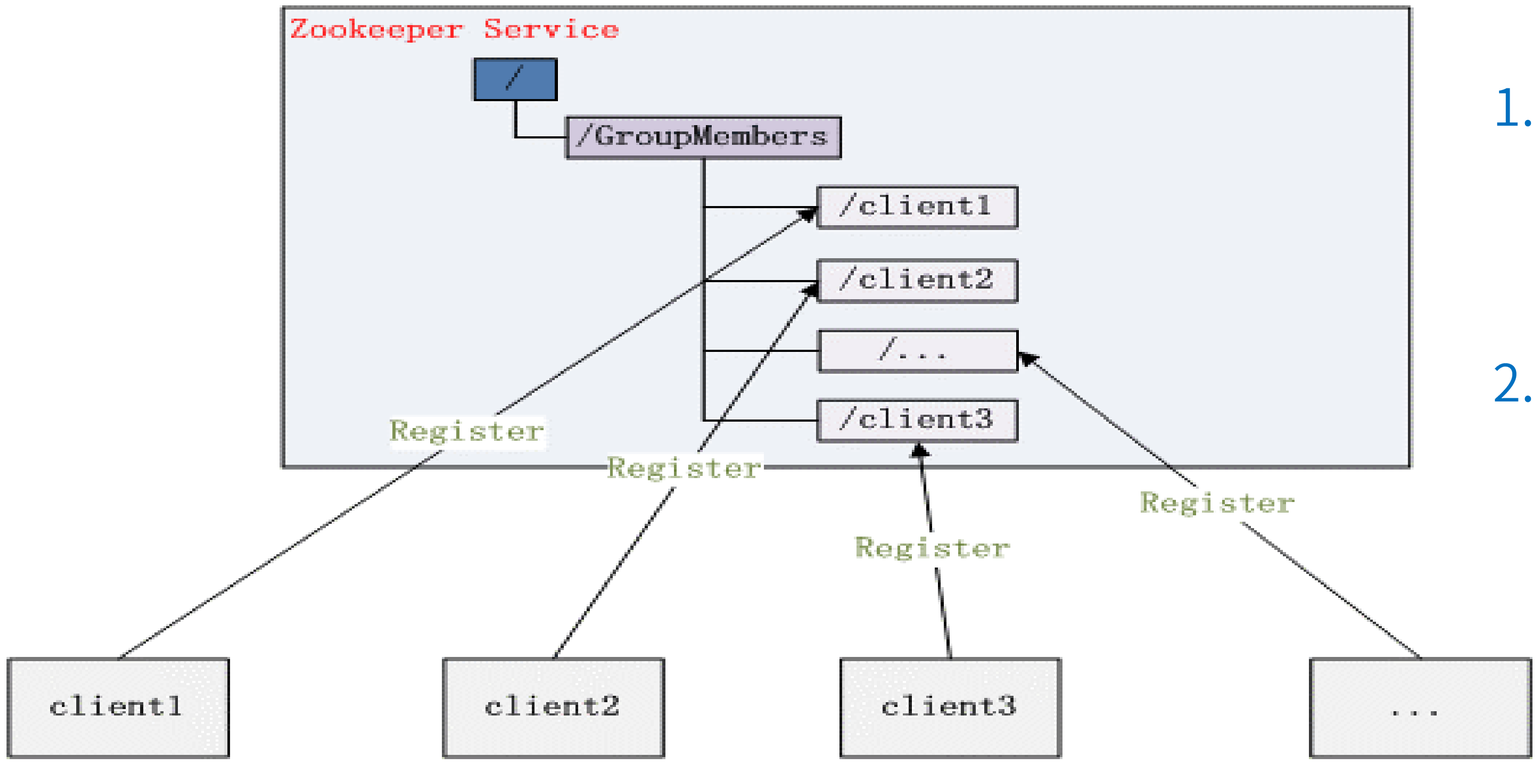
常用分布式锁实现技术

- **基于数据库实现分布式锁**
  - 性能较差，容易出现单点故障
  - 锁没有失效时间，容易死锁
- **基于缓存实现分布式锁**
  - 实现复杂
  - 存在死锁（或短时间死锁）的可能
- **基于zookeeper实现分布式锁**
  - 实现相对简单
  - 可靠性高
  - 性能较好

基于zookeeper实现分布式锁

Zookeeper简介

- ZooKeeper是一个分布式的，开放源码的分布式应用程序协调服务，是Hadoop和Hbase的重要组件。



特性：

1. 节点树数据结构，znode是一个跟Unix文件系统路径相似的节点，可以往这个节点存储或获取数据；
2. 通过客户端可对znode进行增删改查的操作，还可以注册watcher监控znode的变化。

Zookeeper节点类型

- 持久节点（PERSISTENT）
- 持久顺序节点（PERSISTENT_SEQUENTIAL）
- 临时节点（EPHEMERAL）
- 临时顺序节点（EPHEMERAL_SEQUENTIAL）

同一个znode下，节点的名称是唯一的！

Zookeeper典型应用场景

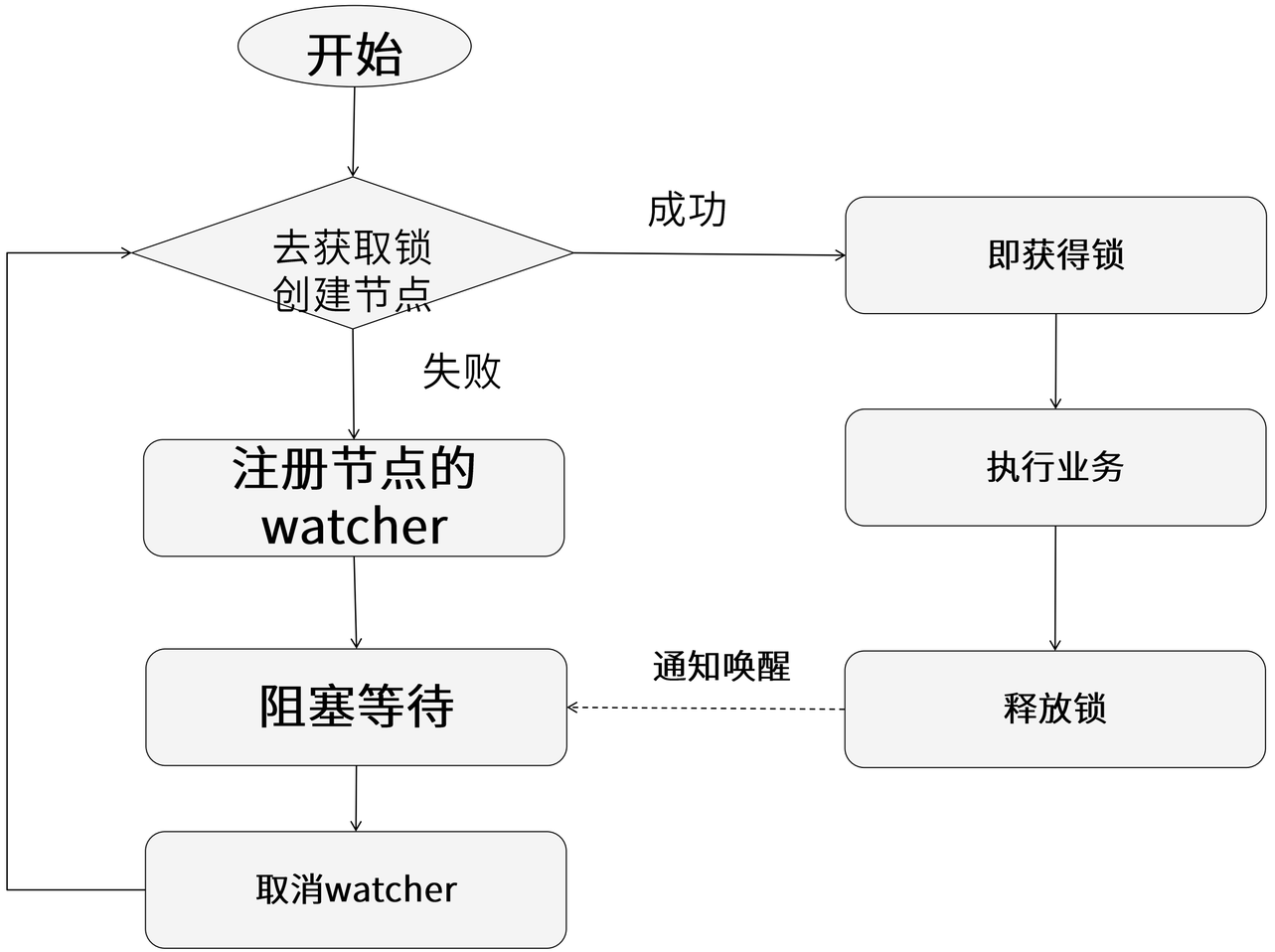
1. 数据发布订阅（配置中心）
2. 命名服务
3. Master选举
4. 集群管理
5. 分布式队列
6. 分布式锁

用Zookeeper实现分布式锁一

- 特性：同父的字节点不可重名

用Zookeeper实现分布式锁逻辑一

- 特性：同父的字节点不可重名



用持久节点可以吗？

万一持有锁的进程挂了……
可就死锁了

用临时节点呢？

惊群效应

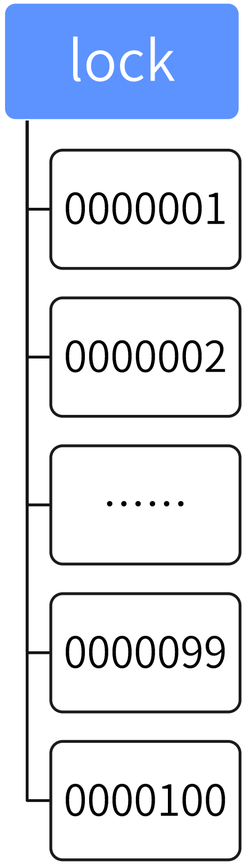
在集群规模较大的环境中带来的危害：

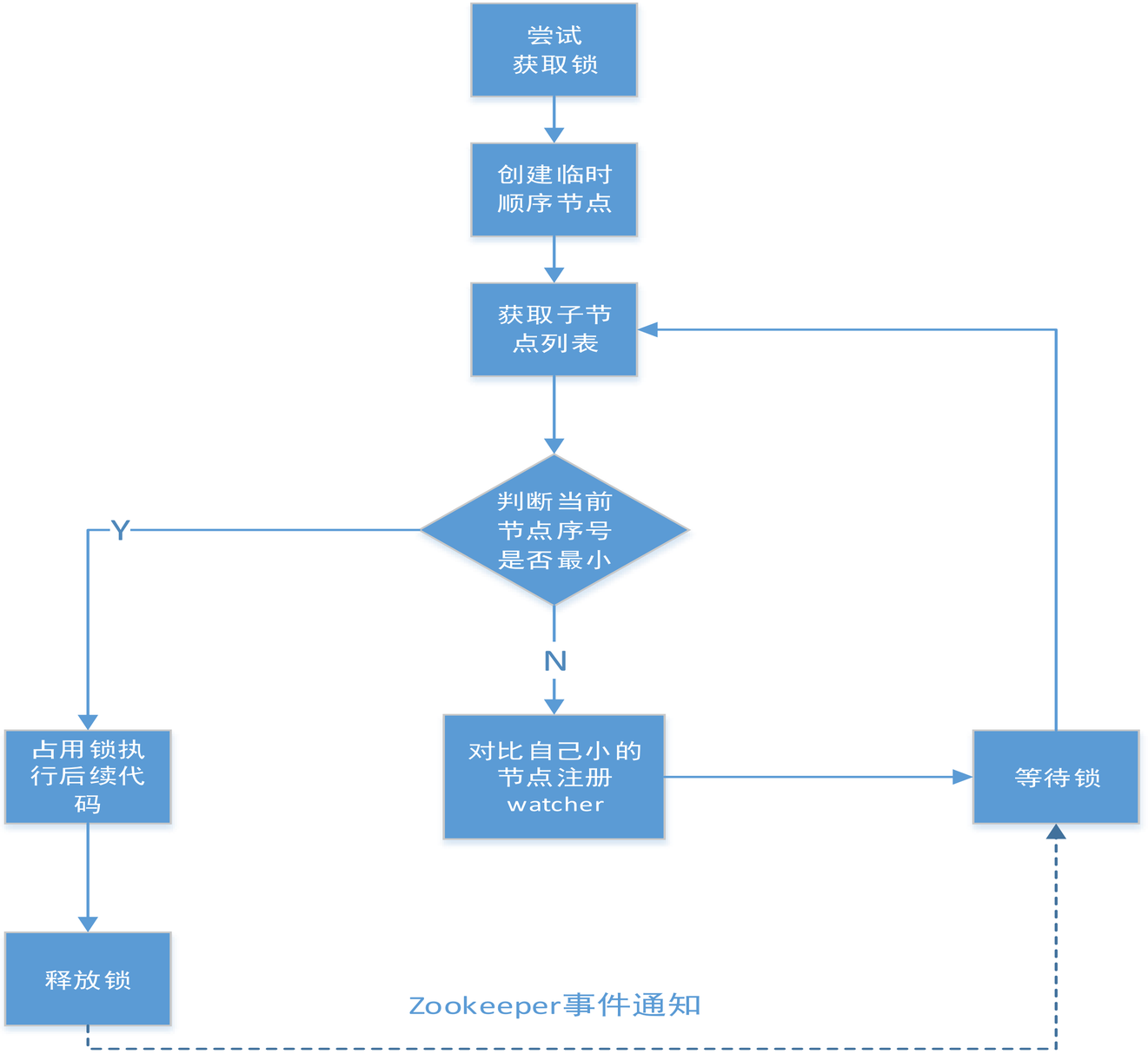
- 巨大的服务器性能损耗
- 网络冲击
- 可能造成宕机

用Zookeeper实现分布式锁二

- 特性：临时顺序节点

改进后的实现数据结构和逻辑





动手实践，下节课见