阿里云 × CLOUD NATIVE COMPUTING FOUNDATION

云原生技术公开课

第 16 讲

深入理解 etcd：基本原理解析

曾凡松 阿里巴巴高级技术专家

关注“阿里巴巴云原生”公众号
获取第一手技术资料

本节大纲

- etcd 项目的发展历程
- 架构及内部机制解析
- 典型的使用场景介绍

1 etcd 项目发展的历程

etcd 项目发展的历程

etcd 诞生于 CoreOS 公司，最初用于解决集群管理系统中 OS 升级时的分布式并发控制、配置文件的存储与分发等问题。基于此，etcd 被设计为提供高可用、强一致的小型 KeyValue 数据存储服务。项目当前隶属于 CNCF 基金会，被包括 AWS、Google、Microsoft、Alibaba 等大型互联网公司广泛的使用。

2013-06

Initial Commit

- CoreOS contribution

2014-06

etcd v0.2

- Kubernetes v0.4
- 10x community

2015-02

etcd v2.0 First Please Release

- Raft consistency protocol
- 1000s of writes/s

2017-01

etcd v3.1

- New API
- Fast linearized read
- gRPC proxy

2018-11

CNCF Incubation

- 30+ of projects use etcd
- 400+ contributors
- 9 maintainers of 8 companies

2019

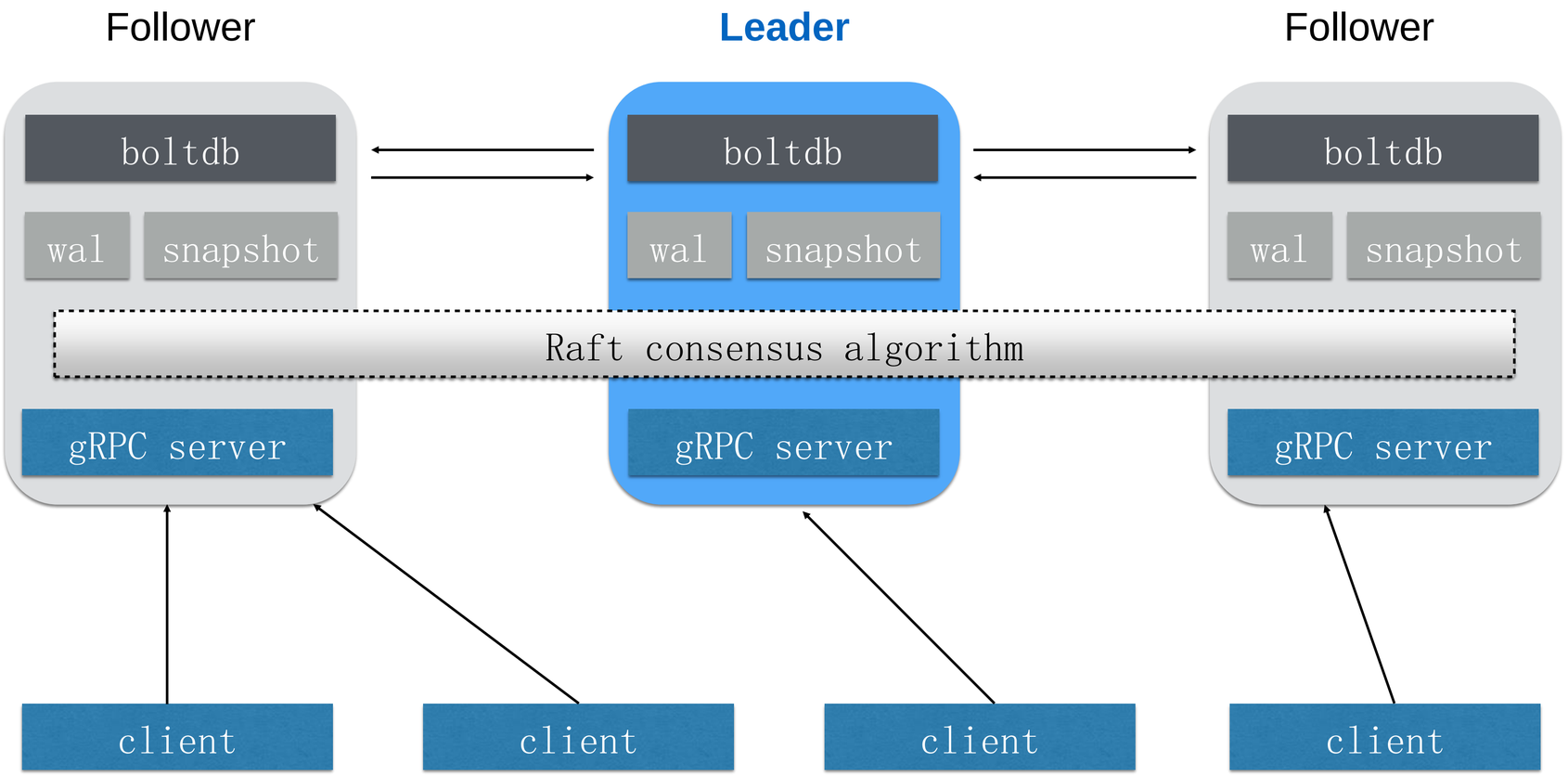
etcd v3.4

- learner member
- fully concurrent read
- performance enhancement

2 · 架构及内部机制解析

架构及内部机制解析

A distributed, reliable key-value store for the most critical data of a distributed system



两个 quorum
一定存在交集

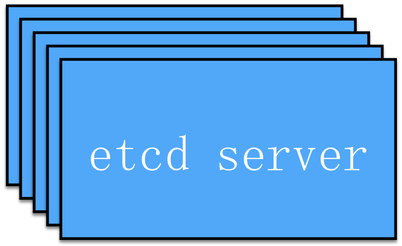
quorum = (n+1) / 2

3个节点容忍1个故障

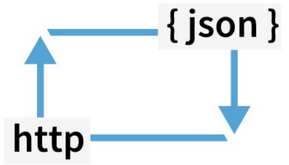
5个节点容忍2个故障

架构及内部机制解析

A distributed, reliable key-value store for the most critical data of a distributed system



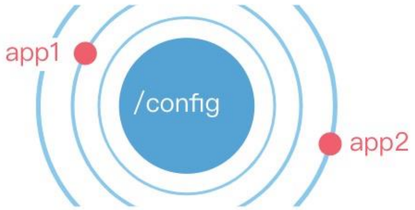
Simple Interface



Key-value DB

```
/config
  └─/database
/feature-flags
  ├─/verbose-logging
  └─/redesign
```

Watch for changes



架构及内部机制解析

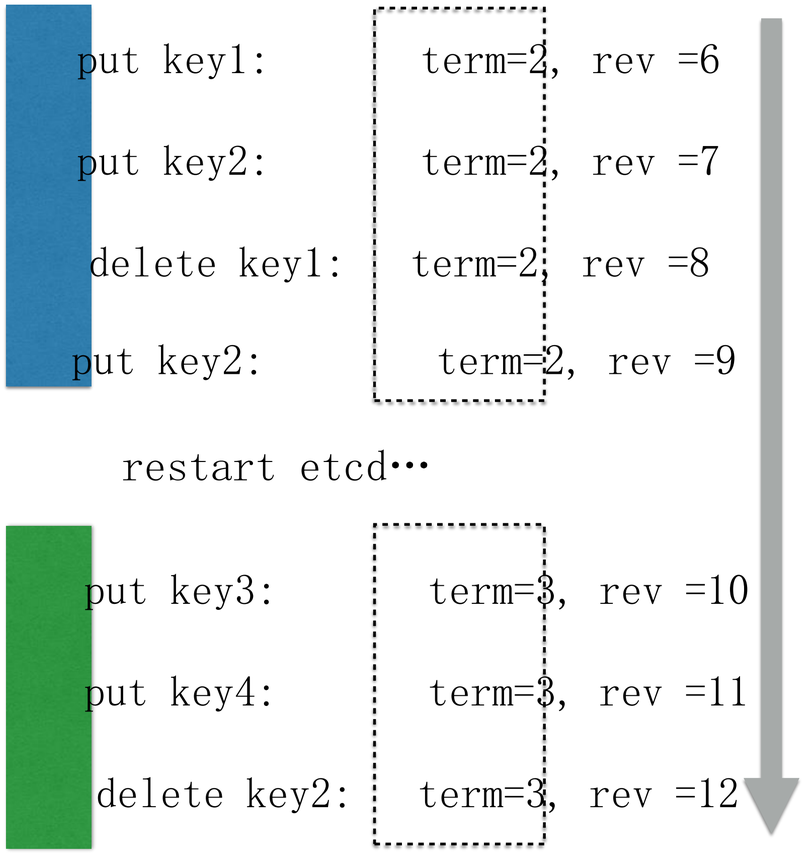
etcd 主要提供了如下一组 APIs

- Put(key, value) / Delete(key)
- Get(key) / Get(keyFrom, keyEnd)
- Watch(key / keyPrefix)
- Transactions(if / then / else ops).Commit()
- Leases: Grant / Revoke / KeepAlive

架构及内部机制解析

etcd 的数据版本号机制

- term: 全局单调递增, 64bits
- revision: 全局单调递增, 64bits
- KeyValue:
  - create_revision
  - mod_revision
  - version

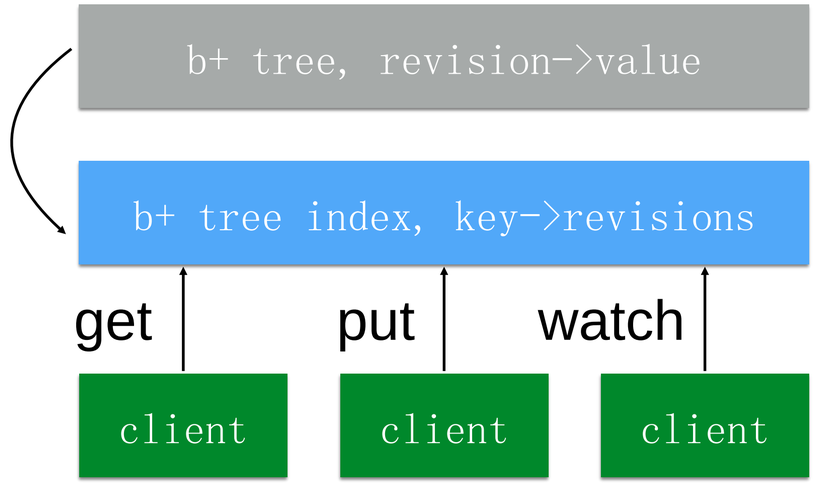


架构及内部机制解析

etcd mvcc & streaming watch

```
Put(key, value1) rev=5

Put(key, value2) rev=6

Get(key) -> value2

Get(key, rev=5) -> value1

…
```

```
watcher = Watch(key, rev)

for {

    event = watcher.Recv()

    handle(event)

    …

}
```

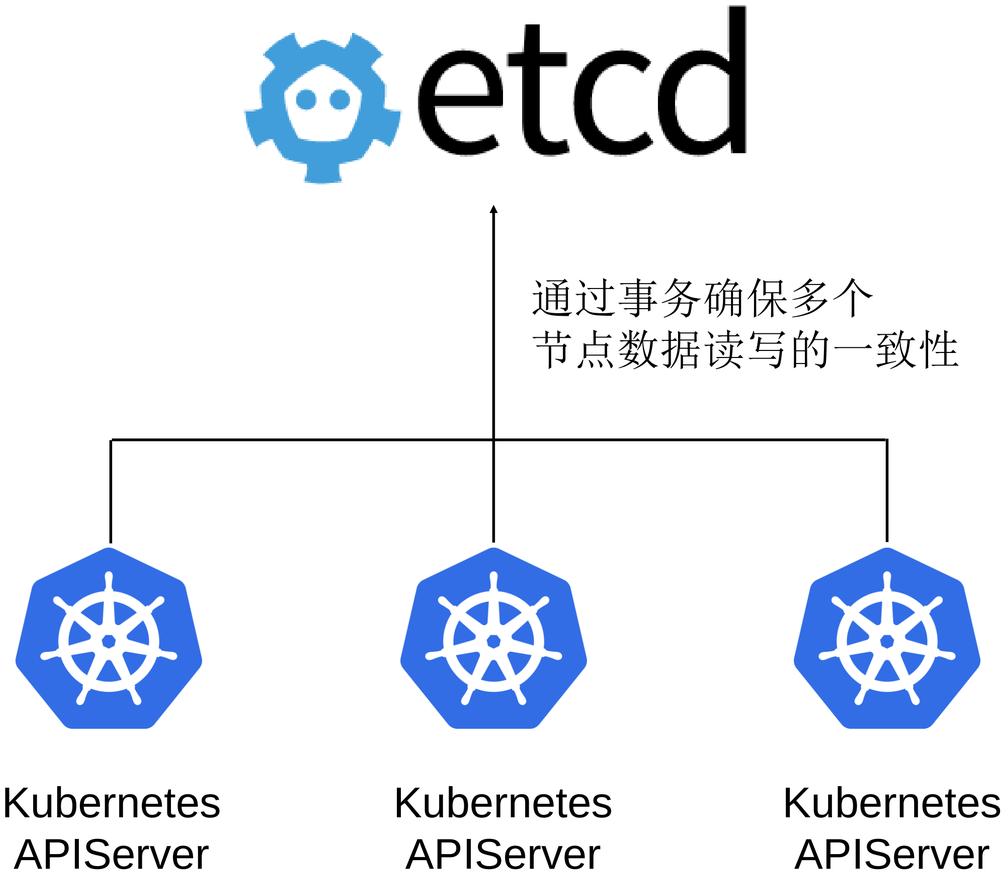


- 一个数据有多个版本
- 通过定期的 Compaction 来清理历史数据

架构及内部机制解析

etcd mini-transactions

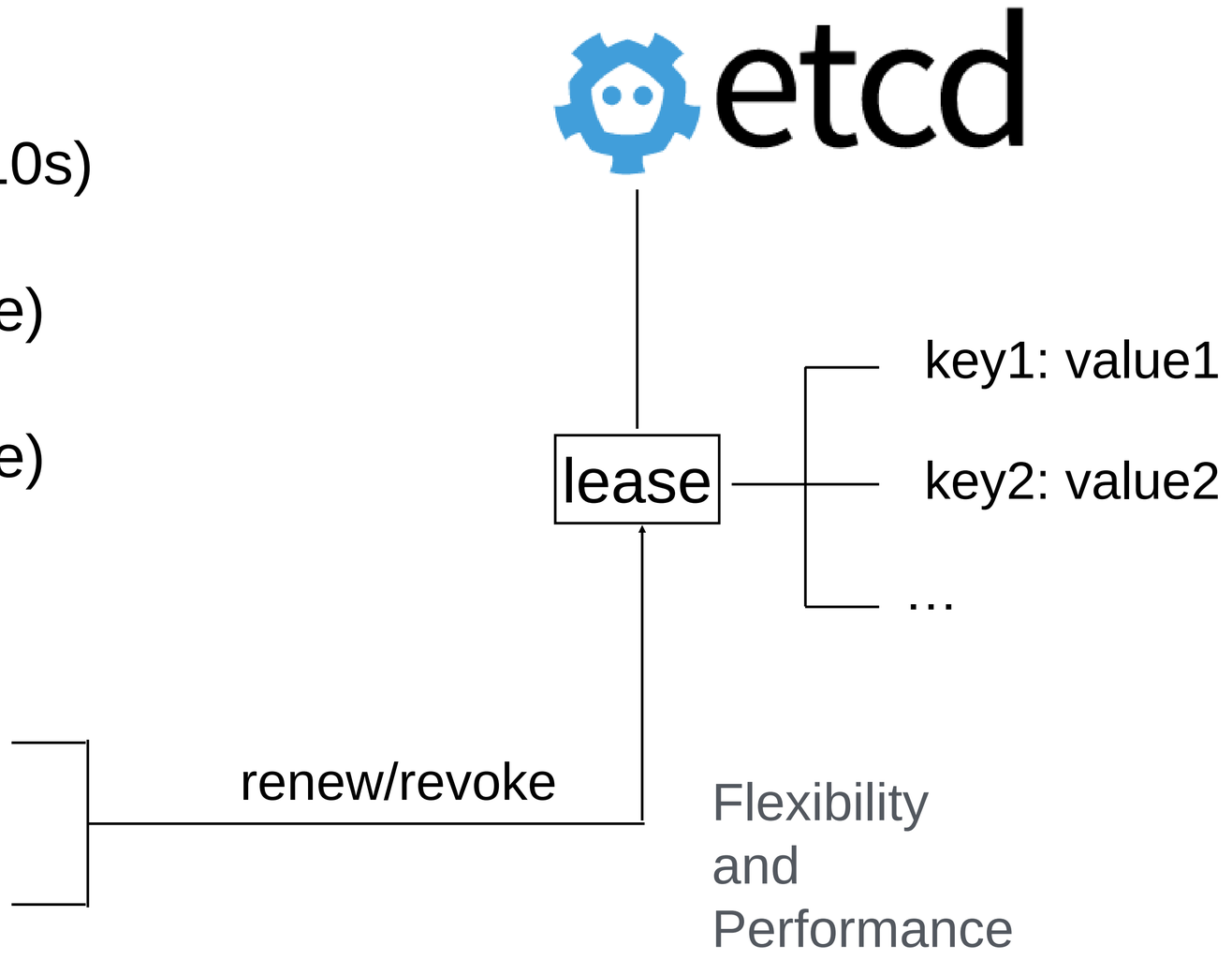
```
Txn.If(
  Compare(Value(key1), ">", "bar"),
  Compare(Version(key1), "=", 2),
  …
).Then(
  Put(key2, valueX),
  Delete(key3)…
). Else(
  Put(key2, valueY)…
).Commit()
```



架构及内部机制解析

etcd lease 的概念及用法

- lease = CreateLease(10s)
- Put(key1, value1, lease)
- Put(key2, value2, lease)
- …
- lease.KeepAlive()
- lease.Revoke()



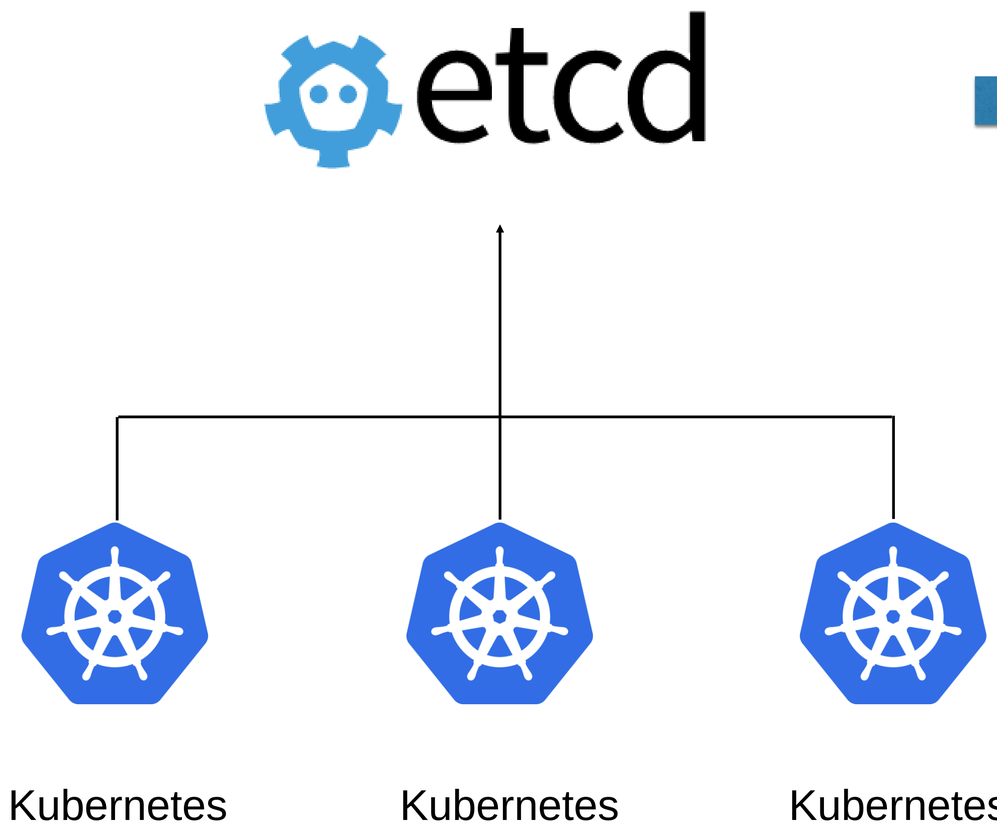
架构及内部机制解析

示例演示

3 典型的使用场景介绍

典型的使用场景介绍

元数据存储——Kubernetes

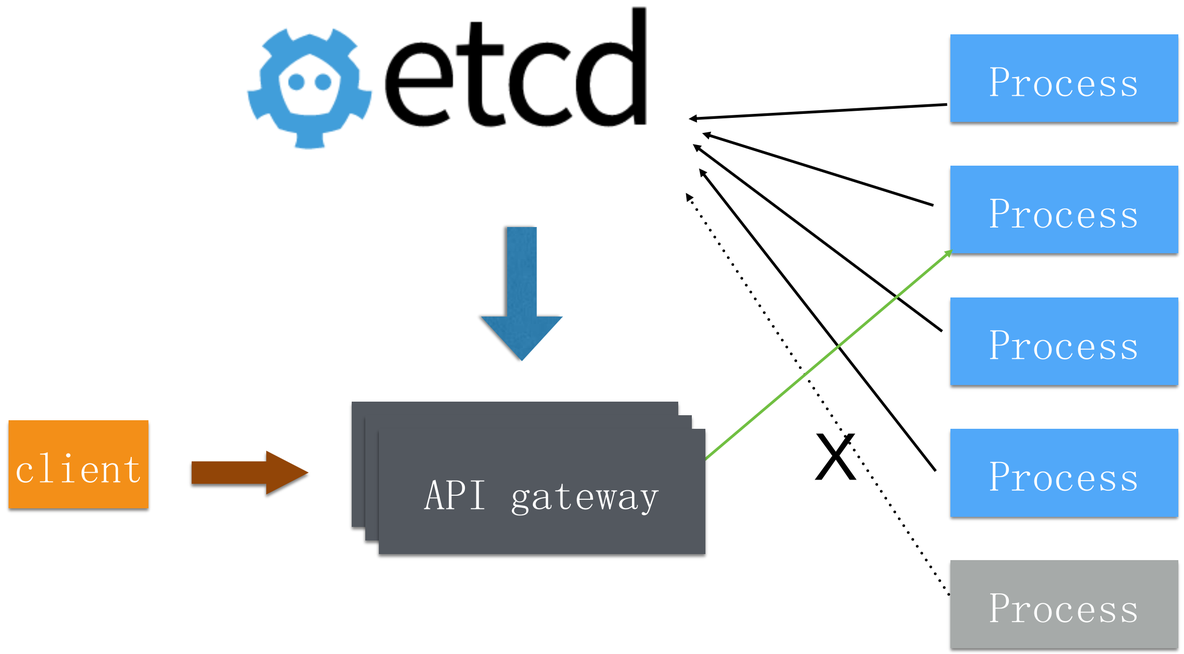


/ns/pods
/minions
/configmaps
/secrets
…

- 元数据高可用，无单点故障
- 系统无状态，故障修复方案简单
- 系统可水平扩展，提高性能及容量
- 简化架构实现，降低系统工程的复杂性

典型的使用场景介绍

Service Discovery (Naming Service)

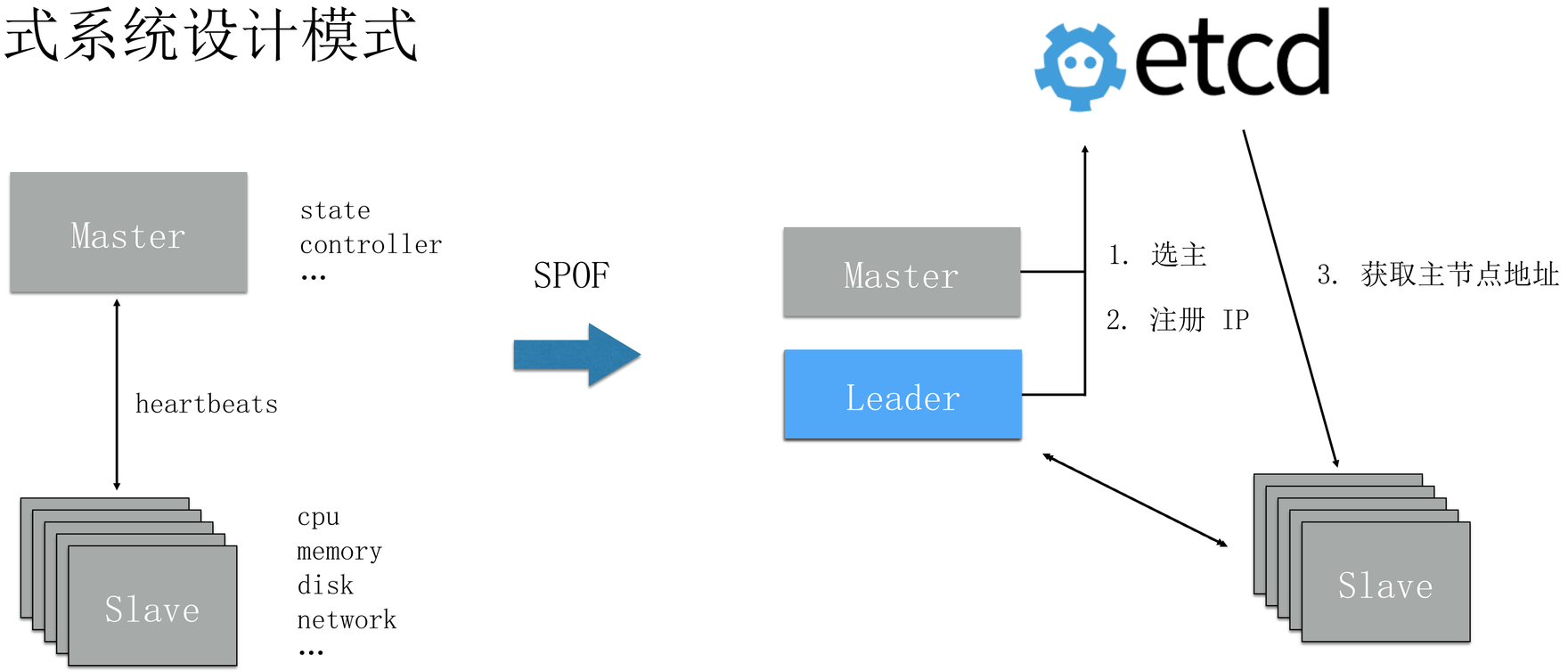


- 资源注册
- 存活性检测
- API gateway 无状态，可水平扩展
- 支持上万个进程的规模

典型的使用场景介绍

Distributed Coordination: leader election

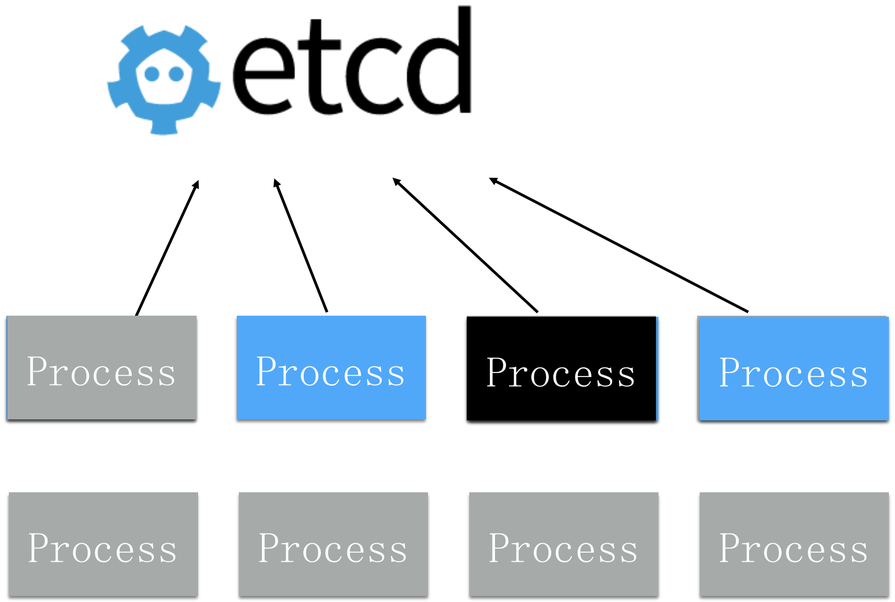
分布式系统设计模式



典型的使用场景介绍

Distributed Coordination

分布式系统并发控制



- 分布式信号量
- 自动踢出故障节点
- 存储进程的执行状态

本节总结

- etcd 项目是如何诞生的，在发展过程中经历的几个阶段
- 介绍了 etcd 架构设计及基本的操作接口，理解了 etcd 如何实现高可用，并结合实践学习了 etcd 数据操作及数据版本管理机制
- 介绍了 3 类典型的 etcd 使用场景，以及对应场景下的系统设计思路

关注“阿里巴巴云原生”公众号
获取第一手技术资料