



KubeCon



CloudNativeCon

China 2018

# 在Kubernetes上运行 区块链服务（BaaS）

余珊 阿里云

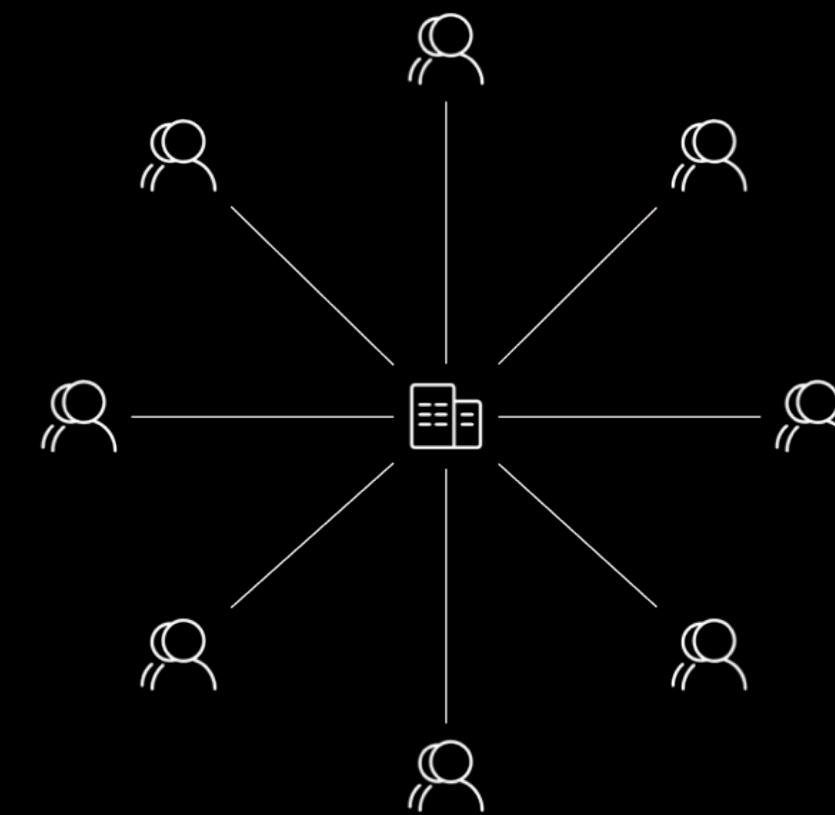


# 议程

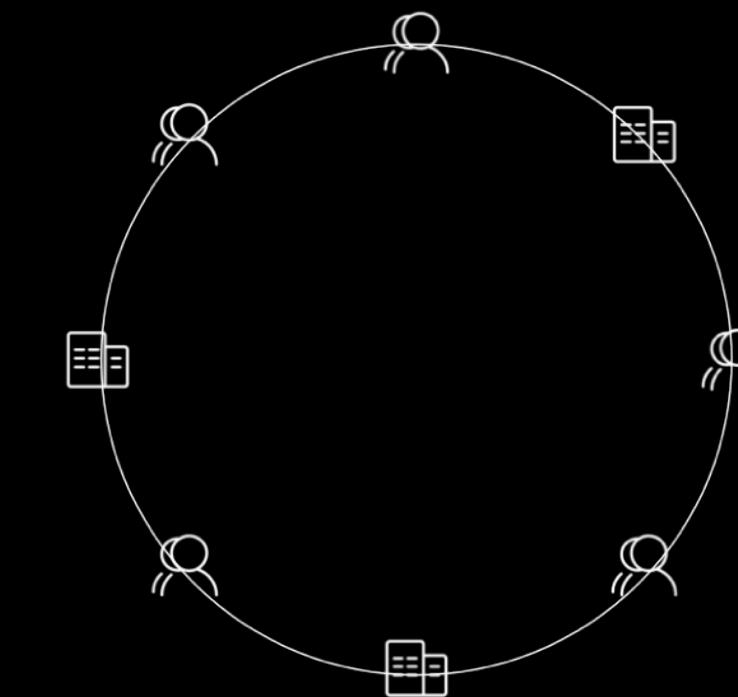
- 1 区块链和BaaS概述
- 2 区块链结合容器与Kubernetes
- 3 关键问题探讨
- 4 区块链业务应用场景与模式
- 5 Demo

# 区块链和BaaS简介

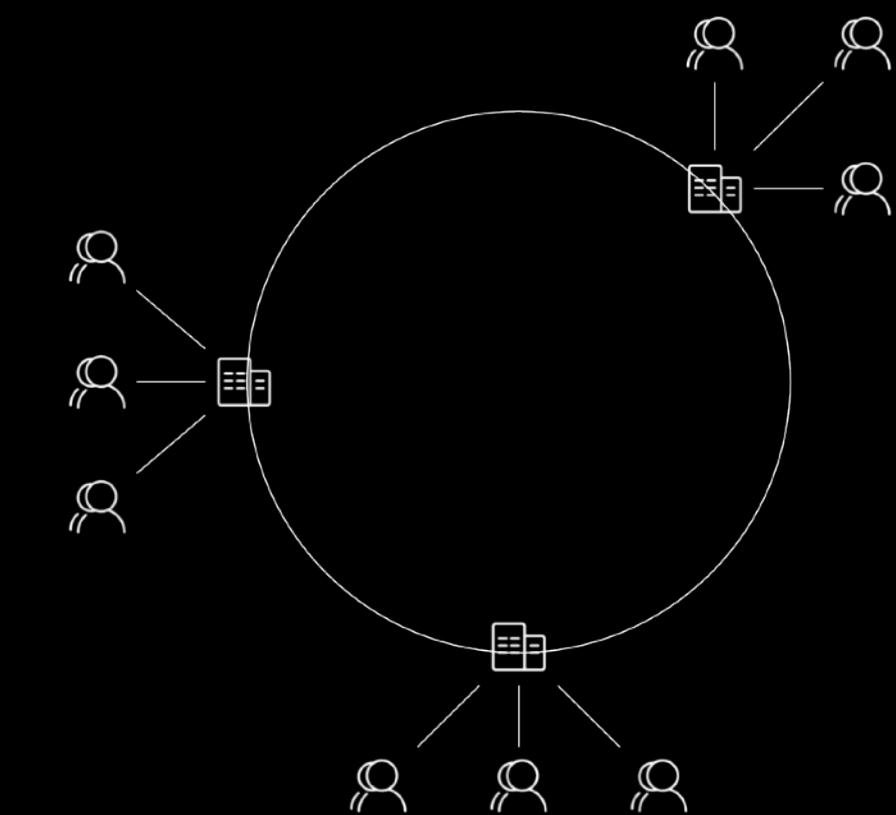
- 区块链是什么
  - 狹义：一种分布式共享账本技术，基于智能合约，在各参与方之间达成对交易的共识，并实现账本交易历史的不可篡改。
  - 广义：在机构、个人、机器之间，构建分布式信任网络、连接可信数据、实现价值流动的一种新的架构和协作模式
- BaaS是什么
  - "Blockchain as a Service"，云平台之上的区块链平台服务，提供区块链系统的部署、运维、治理的能力，提供区块链应用运行和管理的能力



中心化系统  
(如传统中心化系统、私有链)



去中心化系统  
(如公有链)



多中心化系统  
(如联盟链)

# 区块链结合容器和Kubernetes的思考

- 区块链的特点分析
  - 区块链系统：数据为核心，高度分布式，Full-Mesh网络，Long-Running，复杂系统类型
  - 区块链业务应用：没有统一标准，包含各种应用类型
- 区块链使用容器的优势分析
  - 提供了标准化的软件打包、分发的能力
  - 实现了运行环境的一致性以及与底层的解耦
- 区块链使用Kubernetes的优势分析
  - 灵活的资源调度能力
  - 强大的运维管理能力
  - 支持各种应用类型以及微服务架构
  - 云平台集成的优势
  - 丰富的安全和隔离功能
  - 活跃的社区和丰富的生态

微服务架构？

# 阿里云区块链发展历程



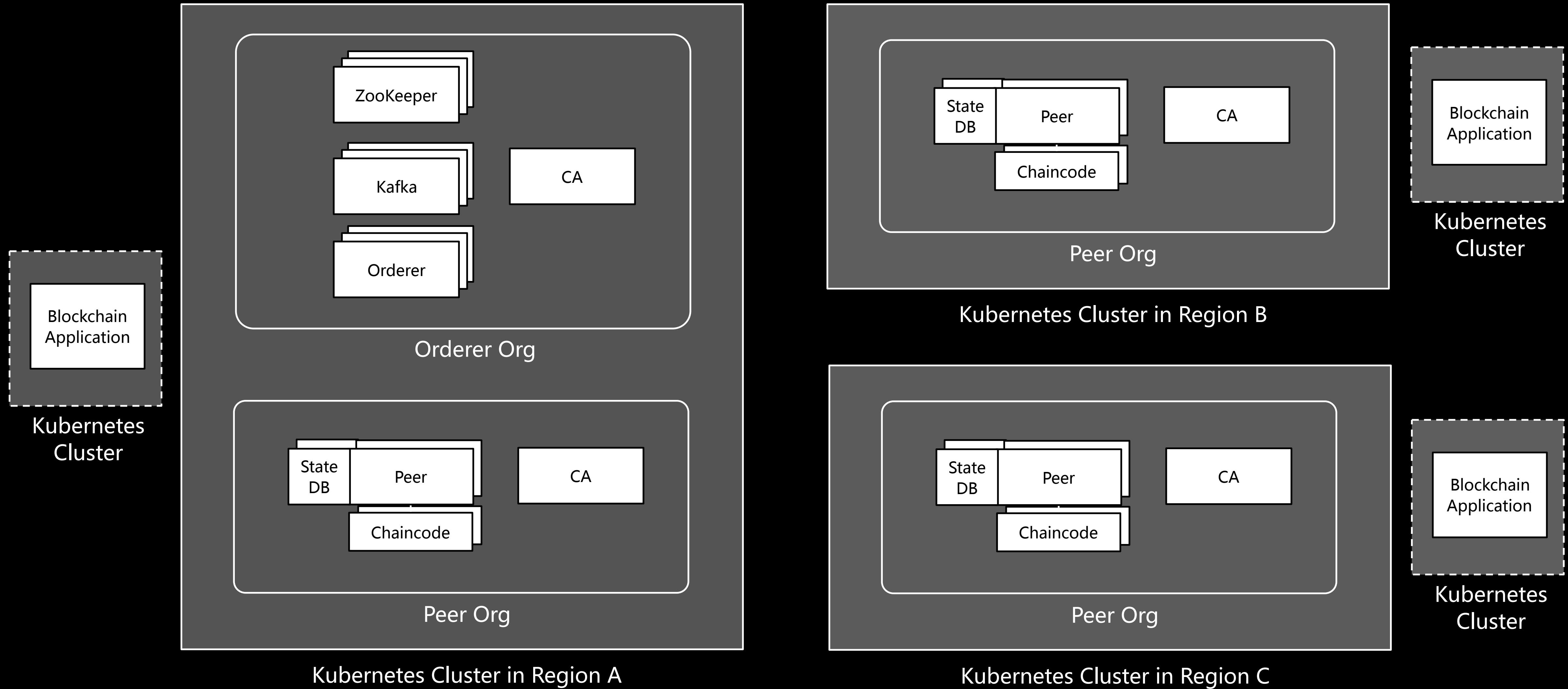
# 构建于Kubernetes之上的阿里云区块链服务BaaS



# 阿里云容器服务Kubernetes版 (ACK)



# 阿里云BaaS - Hyperledger Fabric部署架构图



# BaaS打包、发布、服务编排问题探讨

## 问 题

系统复杂，容器、服务、镜像数量较多  
服务相互依赖较强

## 解决思路

打包  
部署

Helm Chart  
容器镜像

存储  
库管理

阿里云OSS  
阿里云容器镜像服务

配置  
方式

ConfigMap, Secret  
Chart Values

服务  
编排

Chart Template , Chart Hook  
Init Container, Shell Scripts

# BaaS高可用设计

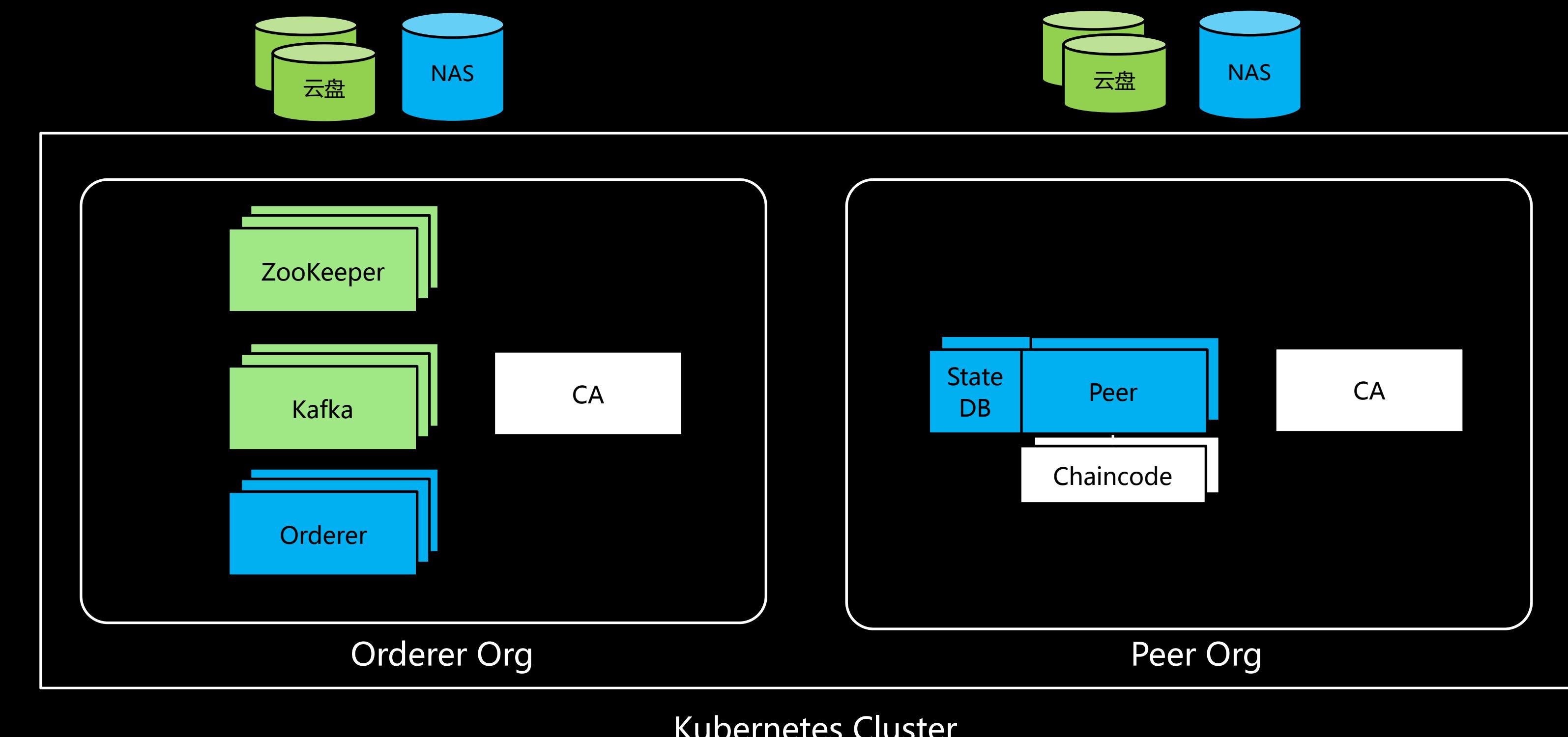


# BaaS数据持久化存储

- 存储类型的选择
  - 文件系统：[阿里云NAS](#)
  - 块存储：[云盘](#)
  - 均提供极高的数据可靠性
    - NAS : 99.99999999% , 云盘 : 99.999999%
  - 均选用SSD类型保证I/O性能
- Kubernetes存储使用方式
  - PV + PVC
- 阿里云NAS适合区块链账本存储的特点
  - 存储动态无缝扩容，应用需无中断/重启
  - 容量越大、性能越高

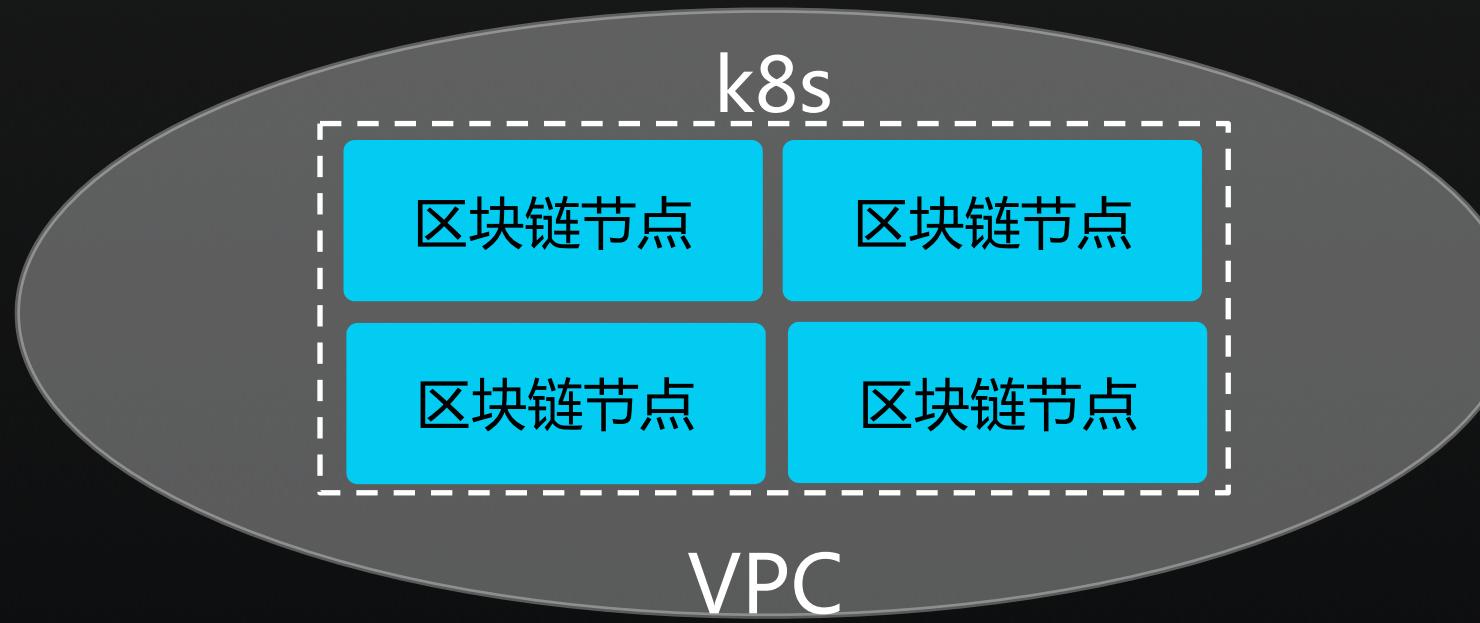
本地盘的问题：

- Kubernetes调度存在的“漂移”
- 如使用NodeSelector，可能存在的容灾恢复问题



# BaaS联盟链网络方案方面的挑战

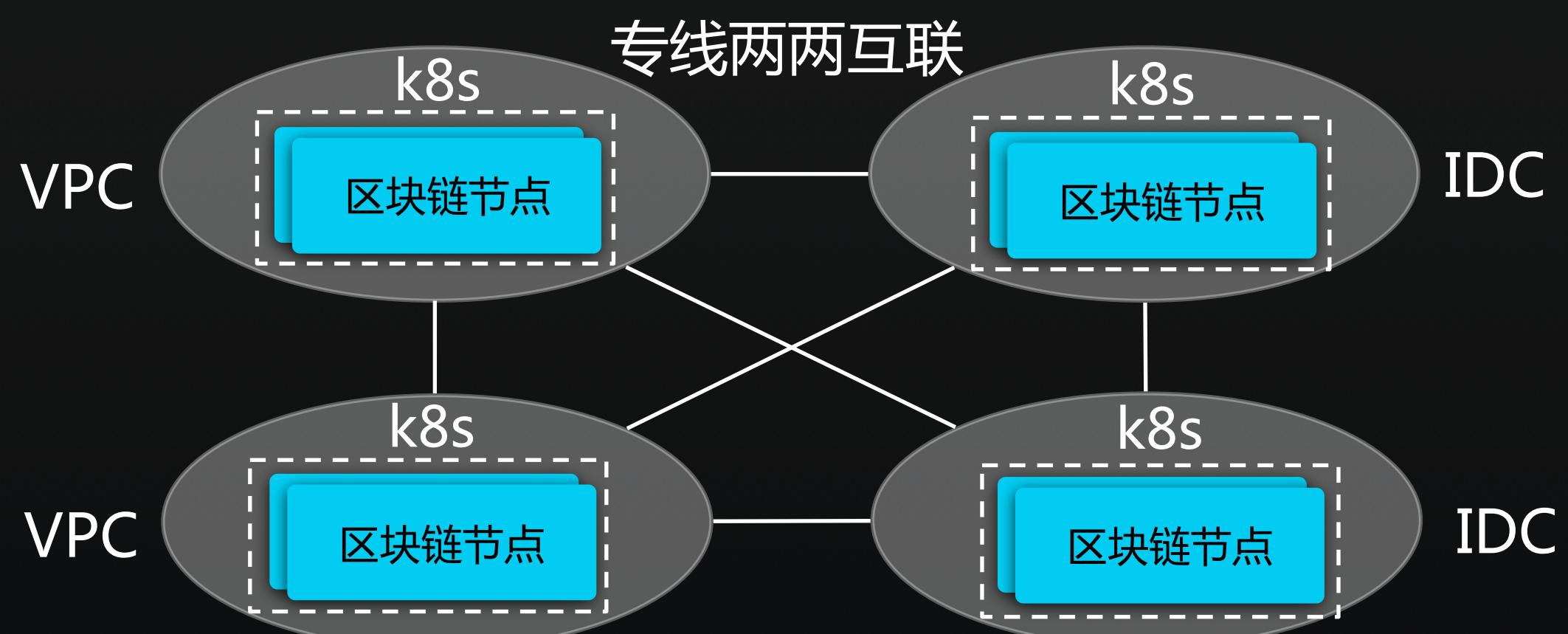
单一VPC的联盟链网络方案  
近似私有链



基于公网的联盟链网络方案  
满足不了高安全需求

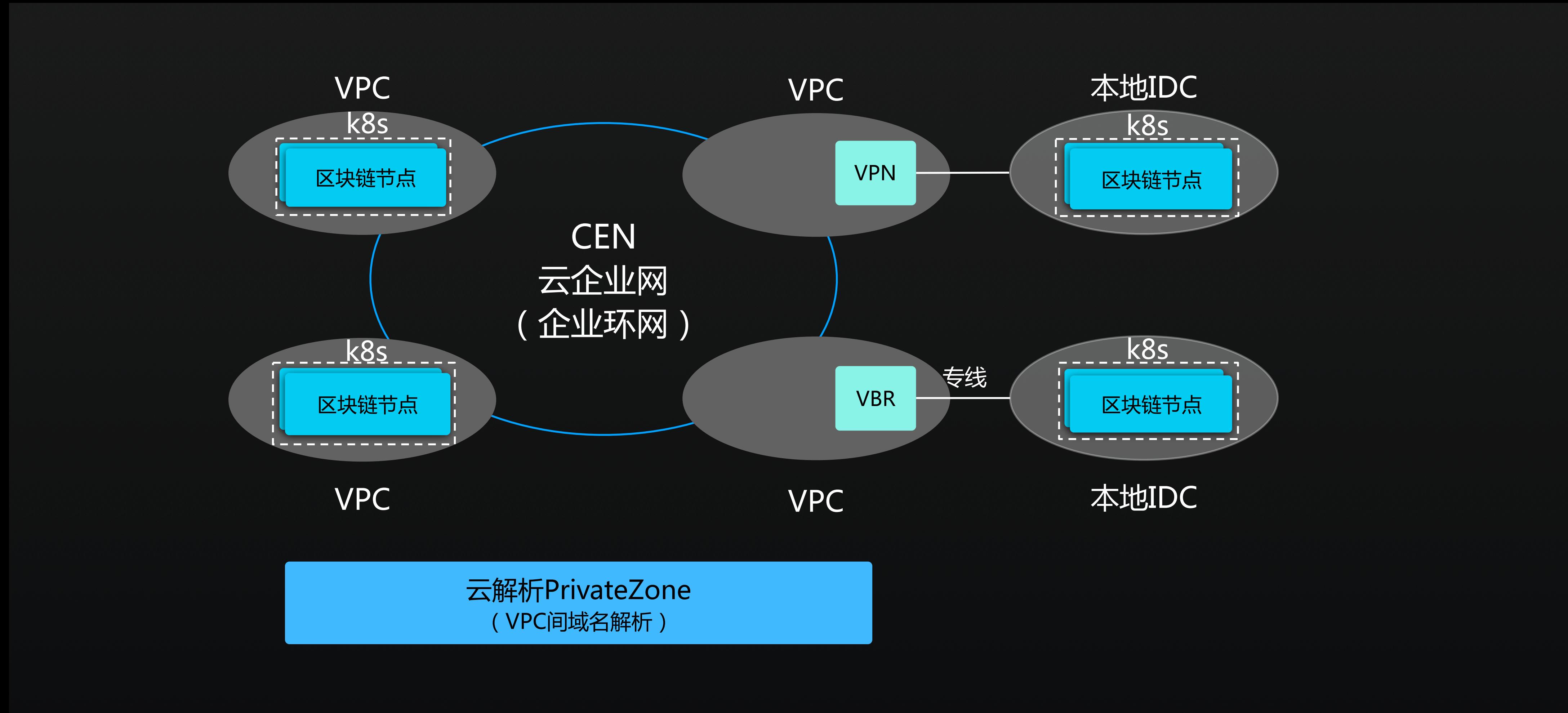


基于专线互联的联盟链网络方案  
复杂度和成本高

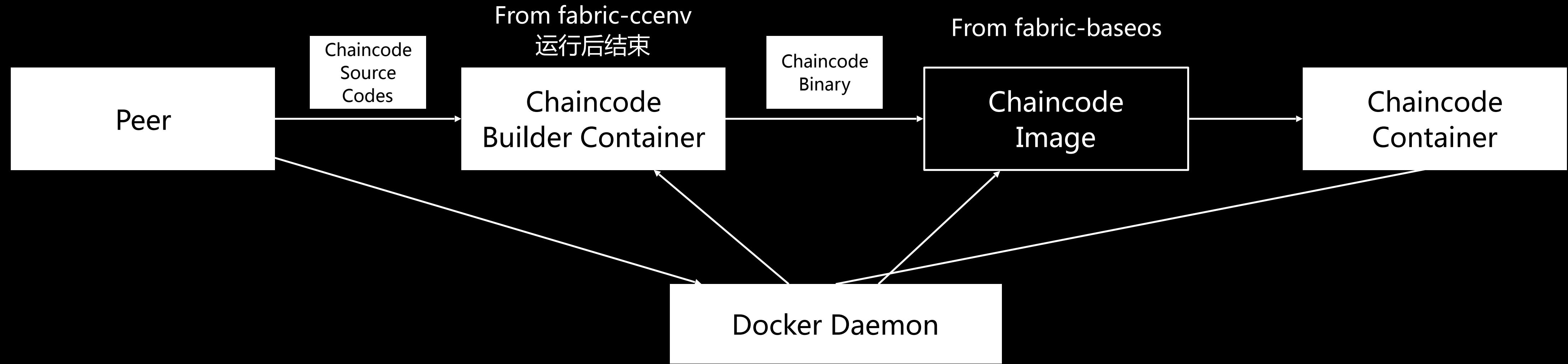


# 基于CEN云企业网的安全联盟链网络方案

跨企业、跨账户



# 智能合约Chaincode容器探讨



- Chaincode容器存在的问题分析
  - 独立于Kubernetes体系之外运行，难以对chaincode容器进行生命周期管理
  - 无法基于Kubernetes的namespace隔离、network policy等机制实现对chaincode容器的安全管理

# 智能合约Chaincode容器探讨

- 问题解决思路
  - 思路一：将Chaincode容器纳入到Kubernetes体系（如Pod）进行管理
    - 是最理想的方案，不仅可实现全生命周期与Fabric其他类型节点一致的管理方式，并且可结合Kubernetes的NetworkPolicy控制Chaincode容器的访问策略
    - Hyperledger Fabric社区相关需求(JIRA)，但目前仍未实现：<https://jira.hyperledger.org/browse/FAB-7406>

未来展望：将智能合约容器调度运行于Serverless Kubernetes之上，提供kernel级别的隔离、保证应用容器之间的安全隔离

# 智能合约Chaincode容器探讨

- 问题解决思路
  - 思路二：将Chaincode容器放入Docker-in-Docker (DIND)环境
    - 试验代码示例
    - 观察和分析

## 优点

- 无需依赖宿主节点的/var/run/docker.sock
- 无需专门清理每个worker节点的Chaincode image

## 缺点

- 每次创建部署或恢复peer节点会很慢（因为dind需要去拉取fabric-ccenv镜像 1.4GB）；而如果传统方式只需在worker节点拉取一次即可。
- Chaincode instantiate速度稍微变慢
- 当peer节点或者整个org的网络删掉重建之后（数据目录复用的模式），启动速度比起原来的方式会慢很多（原因同1）
- 在业界实践中DIND方法主要用于CI/CD，但对于生产环境使用则稳定性有较多挑战
- 仍解决不了chaincode容器的安全访问控制和隔离的问题

```
spec:  
  containers:  
    # Docker-in-docker container for peer to build chaincode image and run chaincode container  
    # without dependency on docker.sock of host machine  
    - name: dind  
      image: docker:dind  
      securityContext:  
        privileged: true  
      volumeMounts:  
        - mountPath: /var/lib/docker  
          mountPropagation: ""  
          name: dind-storage  
      lifecycle:  
        postStart:  
          exec:  
            command: ["/bin/sh", "-c", "docker pull {{ $.Values.dockerImageRegistry }}/fabric-ccenv:x86_64-{{ $.Values.hyperledgerFabricVersion }}  
              && docker tag {{ $.Values.dockerImageRegistry }}/fabric-ccenv:x86_64-{{ $.Values.hyperledgerFabricVersion }}  
                hyperledger/fabric-ccenv:x86_64-{{ $.Values.hyperledgerFabricVersion }}"]  
  
    - name: peer  
      image: {{ $.Values.dockerImageRegistry }}/fabric-peer:x86_64-{{ $.Values.hyperledgerFabricVersion }}  
      imagePullPolicy: {{ $.Values.imagePullPolicy | quote }}  
      ports:  
        - containerPort: 7051  
          name: grpc-port  
        - containerPort: 7053  
          name: event-port  
      command: ["peer"]  
      args: ["node", "start"]  
      env:  
        - name: CORE_VM_ENDPOINT  
          value: http://localhost:2375
```

# 智能合约Chaincode容器探讨

- 问题解决思路

- 思路三：综合配置方式先解决最主要问题
  - Fabric Peer的配置保证与Chaincode的通信
  - 使用docker rm和docker rmi命令清理

Chaincode 容器和镜像 ("dev-"前缀 )

- 可选位置：DaemonSet + lifecycle.preStop.exec.command ( 事后清理 )
  - 可选位置：initContainer ( 事前清理 )
  - 采用iptables规则，对Chaincode container 进行网络隔离
    - 在Helm Chart安装阶段配置K8S Worker节点的iptables规则
    - 实现限制Chaincode对K8S网络和对公网的访问

```
containers:
  - name: peer
    env:
      - name: CORE_VM_ENDPOINT
        value: "unix:///host/var/run/docker.sock"
      - name: CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE
        value: "bridge" # Use docker0 of host
      - name: CORE_PEER_ADDRESSAUTODETECT
        value: "true" # Ensure to get IP address of peer pod
      - name: CORE_PEER_ADDRESS
        value: "{{ include \"orgName\" $ }}-{{ include \"peerName\" $ }}{{ $peerIndex }}:7051"
    ...
    volumeMounts:
      - name: docker-sock
        mountPath: /host/var/run/docker.sock
```

```
[root@izbp11nv9k9gni0rax7y1fz ~]# docker ps --no-trunc | grep dev
02fa83e9a96ba2220d102aee0c36f55ed9bae3f0dace551fde15da3a3e93da6f
47a8afc4ef58e
9 days ago          Up 9 days           dev-network01-peer1-mvcc-1.0-7c9a71f75ec87c695dca1da2a81bf3e966e18118471983
"chaincode -peer.address={{ .IP }}:7051"
dev-network01-peer1-mycc-1.0
```

自动解析出Peer的Pod IP并在启动Chaincode容器时传入

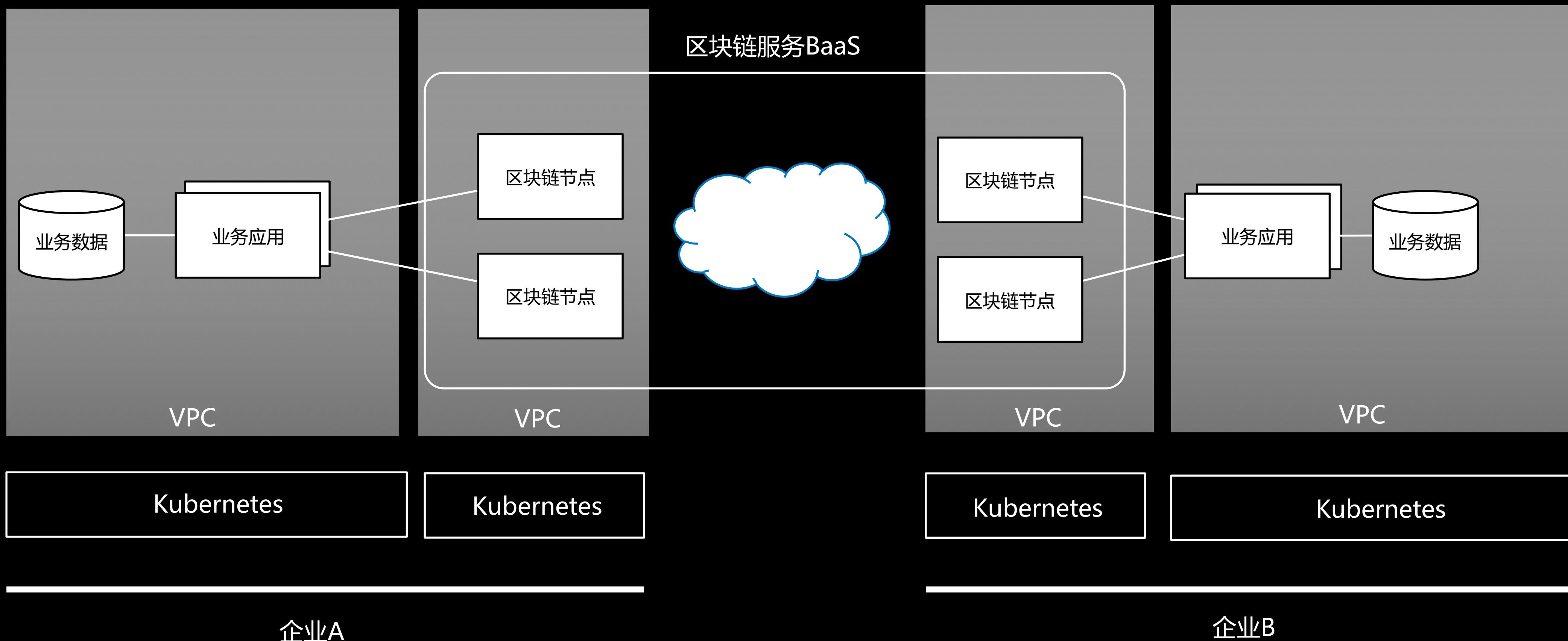
# 区块链业务应用场景示例

## 商品溯源



零售

# 构建于Kubernetes之上区块链系统和应用基本模式



# Demo

- 快速创建跨企业（账号）、跨region的联盟链
- 动态添加新组织、新通道，完成企业间协同（邀请、审批）
- 风控保障（关键操作短信验证）
- 部署chaincode和client SDK应用（Marbles）

# 开始使用 Get Started

## 阿里云区块链服务BaaS



产品主页

The screenshot shows the Alibaba Cloud homepage with a dark theme. The top navigation bar includes links for Data Visualization, China Station, Shopping Cart, Control Panel, Documents, Record Filing, Email, and Log In. A prominent search bar is also present. The main content area features a large image of a blockchain network and several promotional sections. One section highlights the service's support for Hyperledger Fabric and self-developed blockchain technology, emphasizing security, stability, ease of use, and openness. Buttons for 'Apply for Free Experience' and 'Product Documentation' are visible. A vertical sidebar on the right offers 'Consultation & Suggestions'.



产品文档



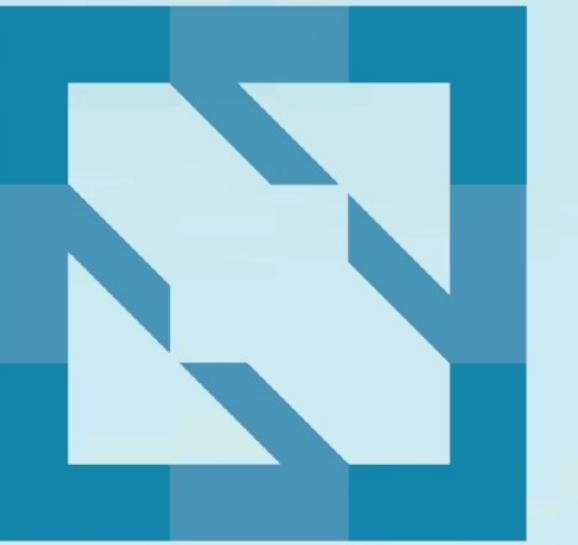
Question ?

# Thank You





**KubeCon**



**CloudNativeCon**

China 2018

