Federated Access Point
eBay统一流量管理方案

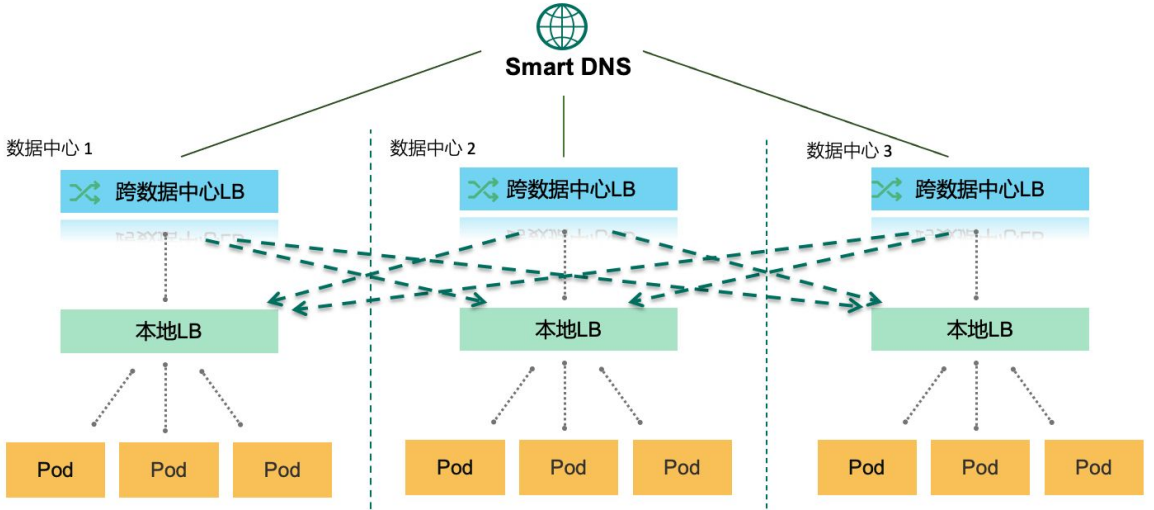
孟凡杰

议程

- 流量管理现状
- 构建基于Istio的流量管理方案
  - Istio部署模式
  - 基于 Locality 信息的高可用接入方案
  - 统一流量模型和自动化流量管理
- 未来展望

eBay流量管理现状

- 采用多活数据中心的网 络拓扑，任何生 产应用都需要完成跨三个数据中心的部署。
- 为满足单集群的高可用，针对每个数据中心，任何 应用都需进行多副本部署，并配置 负载均衡。
- 以实现全站微服 务化，但为保证高可用，服 务之间的调用仍以南北流量 为主。
- 针对核心应用，除集群本地 负载均衡配置以外，还需配置跨数据中心 负载均衡，并通过权重控制将 99% 的请求转入本地数据中心，将 1% 的



规模化带来的挑战

- 异构应用
  - 云业务，大数据，搜索服务
  - 多种应用协议
  - 灰度发布
- 日益增长的安全需求
  - 全链路TLS
- 可见性需求
  - 访问日志
  - Tracing

- 3主数据中心，20边缘数据中心，100+ Kubernetes集群
- 规模化运营Kubernetes集群
  - 总计100,000物理节点
  - 单集群物理机节点规模高达 5,000
- 业务服务全面容器化，单集群
  - Pod实例可达 100,000
  - 发布服务 5,000-10000
- 单集群多环境支持
  - 功能测试、集成测试、压力测试共用单集群
  - 不同环境需要彼此隔离

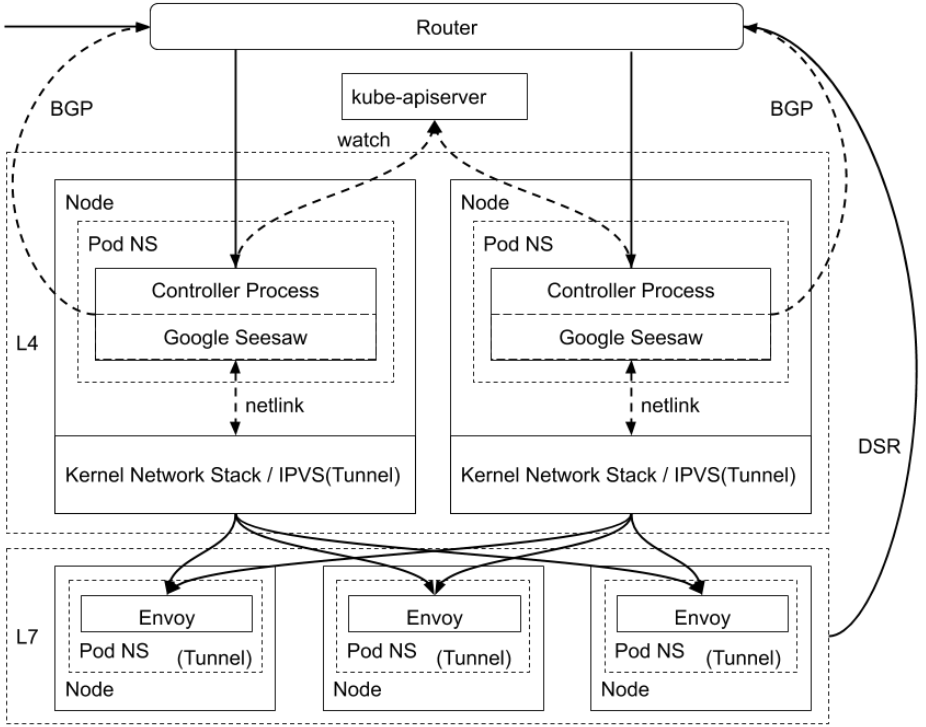
部署模式

多集群部署

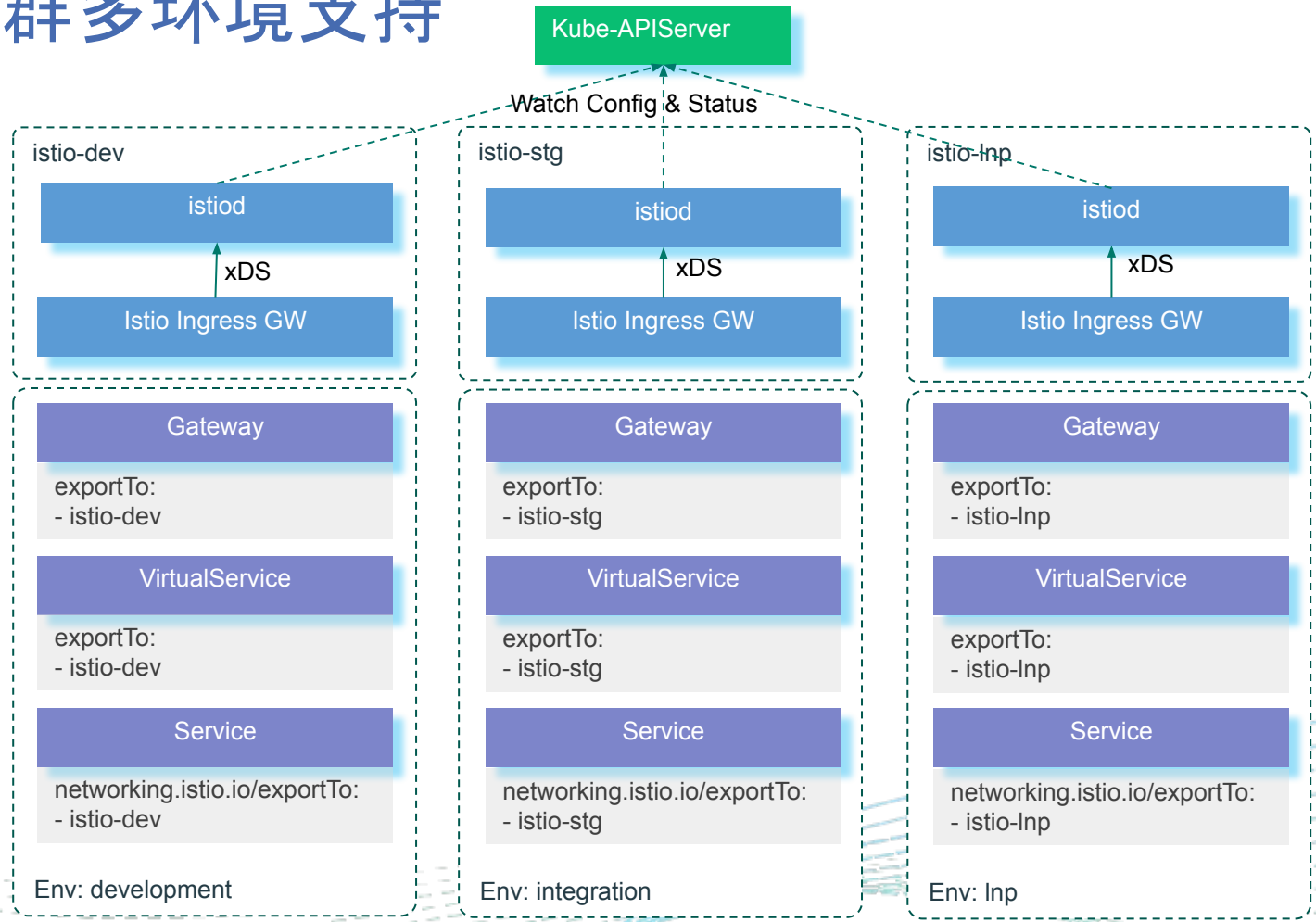
- Kubernetes集群联邦
  - 集群联邦APIServer作为用户访问kubernetes集群入口
  - 所有Kubernetes集群注册至集群联邦
- 可用区
  - 数据中心中具有独立供电制冷设备的故障域
  - 同一可用区有较小网络延迟
  - 同一可用区部署了多个Kubernetes集群
- 多集群部署
  - 同一可用区设定一个网关集群
  - 网关集群中部署Istio Primary
  - 同一可用区的其他集群中部署Istio Remote
  - 所有集群采用相同RootCA
  - 相同环境TrustDomain相同
- 东西南北流量统一管控
  - 同一可用区的服务调用基于Sidecar
  - 跨可用区的服务调用基于Istio Gateway

入站流量架构 L4 + L7

- 为不同应用配置独立的网关服务以方便网络隔离
- 基于IPVS/xDP的Service Controller
  - 四层网关调度
  - 虚拟IP地址分配
  - 基于IPIP协议的转发规则配置
  - 基于BGP的IP路由宣告
  - 在Ingress Pod中配置Tunnel设备，并绑定虚拟IP地址以卸载IPIP包

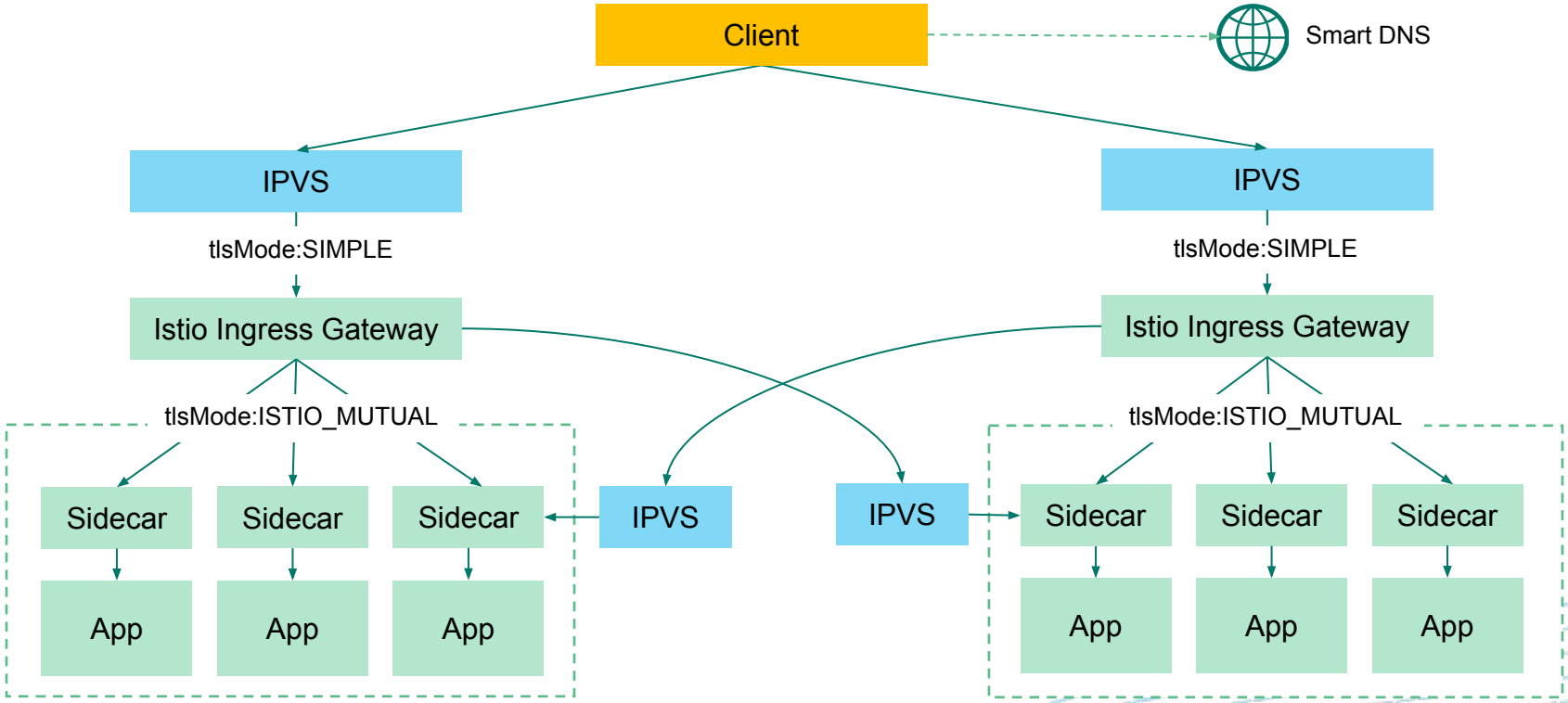


单网关集群多环境支持


Kube-APIServer
Watch Config & Status
istio-dev
istiod
xDS
Istio Ingress GW
istio-stg
istiod
xDS
Istio Ingress GW
istio-lnp
istiod
xDS
Istio Ingress GW
Gateway
exportTo:
- istio-dev
VirtualService
exportTo:
- istio-dev
Service
networking.istio.io/exportTo:
- istio-dev
Env: development
Gateway
exportTo:
- istio-stg
VirtualService
exportTo:
- istio-stg
Service
networking.istio.io/exportTo:
- istio-stg
Env: integration
Gateway
exportTo:
- istio-lnp
VirtualService
exportTo:
- istio-lnp
Service
networking.istio.io/exportTo:
- istio-lnp
Env: lnp

构建生产化应用接入方案

应用高可用接入方案


Client
Smart DNS
IPVS
IPVS
tlsMode:SIMPLE
tlsMode:SIMPLE
Istio Ingress Gateway
Istio Ingress Gateway
tlsMode:ISTIO_MUTUAL
tlsMode:ISTIO_MUTUAL
Sidecar
Sidecar
Sidecar
IPVS
IPVS
Sidecar
Sidecar
Sidecar
App
App
App
App
App
App

为应用发布服务

- 定义 LoadBalancer Type Service，提供集群外可访问的 LoadBalancerIP
- 其他集群可通过定义WorkloadEntry指向该LoadBalancerIP，以实现故障转移目的

创建WorkloadEntry

- 创建WorkloadEntry指向其他数据中心LoadBalancerIP

```
apiVersion:
networking.istio.io/v1beta1
kind: WorkloadEntry
metadata:
  name: foo
spec:
  address: foo.bar.svc.cluster2
  labels:
    run: foo
  locality: region1/zone1
```


故障检测

- 创建WorkloadGroup，定义健康检查规则
- 健康检查规则基于TCP协议而不是httpGet

```
apiVersion: networking.istio.io/v1alpha3
kind: WorkloadGroup
metadata:
  name: foo
  namespace: default
spec:
  metadata:
    labels:
      location: remote
      run: foo
  template:
    ports:
      http-default: 80
  probe:
    initialDelaySeconds: 5
    timeoutSeconds: 3
    periodSeconds: 4
    successThreshold: 1
    failureThreshold: 3
    tcpSocket:
      port: 80
```


定义ServiceEntry同时选择WorkloadEntry和本地Pod

- ServiceEntry对象可以将本地Pod和具有相同Label的WorkloadEntry定义成相同的Envoy Cluster

```
apiVersion:
networking.istio.io/v1beta1
kind: ServiceEntry
metadata:
  name: foo
spec:
  hosts:
  - foo.com
  ports:
  - name: http-default
    number: 80
    protocol: HTTP
    targetPort: 80
  resolution: STATIC
  workloadSelector:
    labels:
      run: foo
```


在VirtualService中引用ServiceEntry

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: foo
spec:
  gateways:
  - foo
  hosts:
  - foo.com
  http:
  - match:
    - port: 80
    route:
    - destination:
        host: foo.com
        port:
          number: 80
```

```
apiVersion:
networking.istio.io/v1beta1
kind: Gateway
metadata:
  name: foo
spec:
  selector:
    istio: ingressgateway
  servers:
  - hosts:
    - foo.com
    port:
      name: http-default
      number: 80
      protocol: HTTP
```

为workload添加 Locality 信息

- Istio 从如下配置中读取，基于这些配置，我们可以为Istio中运行的所有workload添加地域属性
  - Kubernetes Node对象中的地域信息，所有 Pod自动继承该Locality信息
    - region: topology.kubernetes.io/region
    - zone: topology.kubernetes.io/zone
    - subzone: topology.istio.io/subzone
  - Kubernetes Pod的istio-locality标签，可覆盖节点Locality信息
    - istio-locality: "region/zone/subzone"
  - WorkloadEntry的Locality属性
    - locality: region/zone/subzone

定义基于Locality的流量转发规则

Distribute

```
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: foo
spec:
  host: foo.com
  trafficPolicy:
    loadBalancer:
      localityLbSetting:
        distribute:
        - from: "*/*"
          to:
            region1/zone1/*: 99
            region2/zone2/*: 1
        enabled: true
    outlierDetection:
      baseEjectionTime: 10s
      consecutive5xxErrors: 100
      interval: 10s
 tls:
      mode: ISTIO_MUTUAL
```

Failover

```
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: foo
spec:
  host: foo.com
  trafficPolicy:
    loadBalancer:
      localityLbSetting:
        enabled: true
        failover:
        - from: region1/zone1
          to: region2/zone2
    outlierDetection:
      baseEjectionTime: 10m
      consecutive5xxErrors: 1
      interval: 2s
    tls:
      mode: ISTIO_MUTUAL
```


应对规模化集群挑战

应对规模化集群挑战

- Istio xDS默认发现集群中所有的配置和服务状态，在超大规模集群中，Istiod或者Envoy都承受比较大的压力
  - 集群中的有10000Service，每个Service开放80和443两个端口，istio的CDS会discover出20000个Envoy Cluster
  - 如果开启多集群，Istio还会为每个cluster创建符合域名规范的集群
  - Istio 还需要发现remote cluster中的Service，Endpoint和Pod信息，而这些信息的频繁变更，会导致网络带宽占用和控制面板的压力都很大
- meshConfig 中控制可见性

```
defaultServiceExportTo:
- "."
defaultVirtualServiceExportTo:
- "."
defaultDestinationRuleExportTo:
- "."
```

- 通过Istio对象中的exportTo属性覆盖默认配置

Istiod自身的规模控制

- 社区新增加了discoverySelector的支持，允许istiod只发现添加了特定label的namespaces下的Istio以及Kubernetes对象。
- 但因为Kubernetes 框架的限制，改功能依然要让istiod接收所有配置和状态变更新细，并且在istiod中进行对象过滤。在超大集群规模中，并未降低网络带宽占用和istiod的处理压力。
- 需要继续寻求从Kubernetes Server端过滤的解决方案。

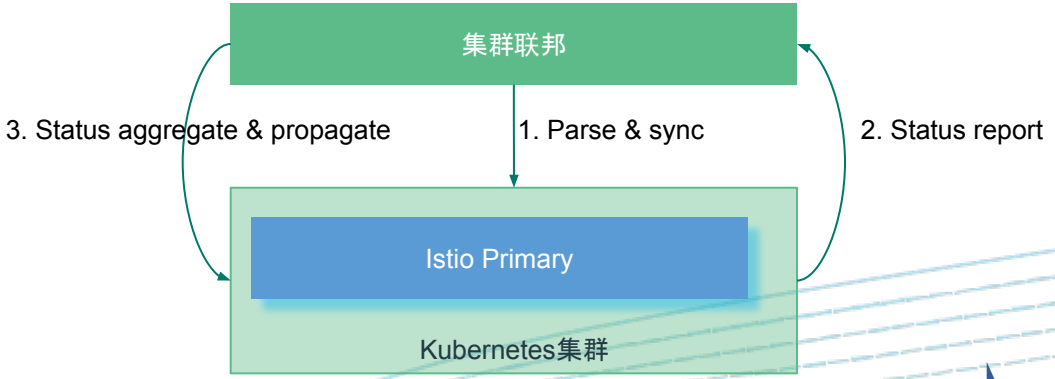
自动化流量管理

- 统一流量模型
- 统一控制器

统一流量模型 - FederatedAccessPoint

- Spec
  - Scope
  - TrafficTemplate
    - Istio models
    - Kubernetes Services
  - Override
    - 可为不同目标集群修改模板属性值
    - 支持多种override方法
      - jsonPatch
      - mergePatch
  - Policy
    - PlacementPolicy
    - RolloutPolicy
    - RuntimePolicy

- Status
  - Conditions
    - 四层网关状态
    - 七层网关配置完成度
    - 证书版本
    - 网关服务IP和FQDN



统一流量模型 - NameService

- Spec
  - Global Name FQDN
  - TTL
  - DNSPolicy
    - RoundRobin
    - Locality
    - Ratio
  - HeathCheck Port
  - Target
    - Target Service FQDN
    - Ratio
- Status
  - Conditions
    - 域名配置结果
  - 配置错误信息

AccessPoint控制器

- PlacementPolicy 控制，用户可以选择目标集群来完成流量配置，甚至可以 选择关联的 FederatedDeployment 对象，使得 AccessPoint 自动发现目标集群并完成配置。
- 完成了状态上报，包括网关虚拟 IP 地址，网关 FQDN，证书安装状态以及版本信息，路由策略是否配置完成等。这补齐了 Istio 自身的短板，使得任何部署在 Istio 的应用的网络配置状态一目了然。
- 发布策略控制，针对多集群的配置，可 实现单集群的灰度发布，并且能够自动暂停发布，管理员验证单个集群的变更正确以后，再 继续发布。通过此机制，避免因 为全局流量变更产生的故障。
- 不同域名的 AccessPoint 可拥有不同的四层网关虚拟 IP 地址，以实现基于 IP 地址的四层网络隔离。
- 控制器可以基于AccessPoint自动创建WorkloadEntry，并设置Locality信息

未来展望

未来展望

- 全面构建基于Mesh的流量管理
- 在用户无感知的前提下将南北流量转成东西流量
- 数据平面加速 Cilium

Thank you!

¥69.00

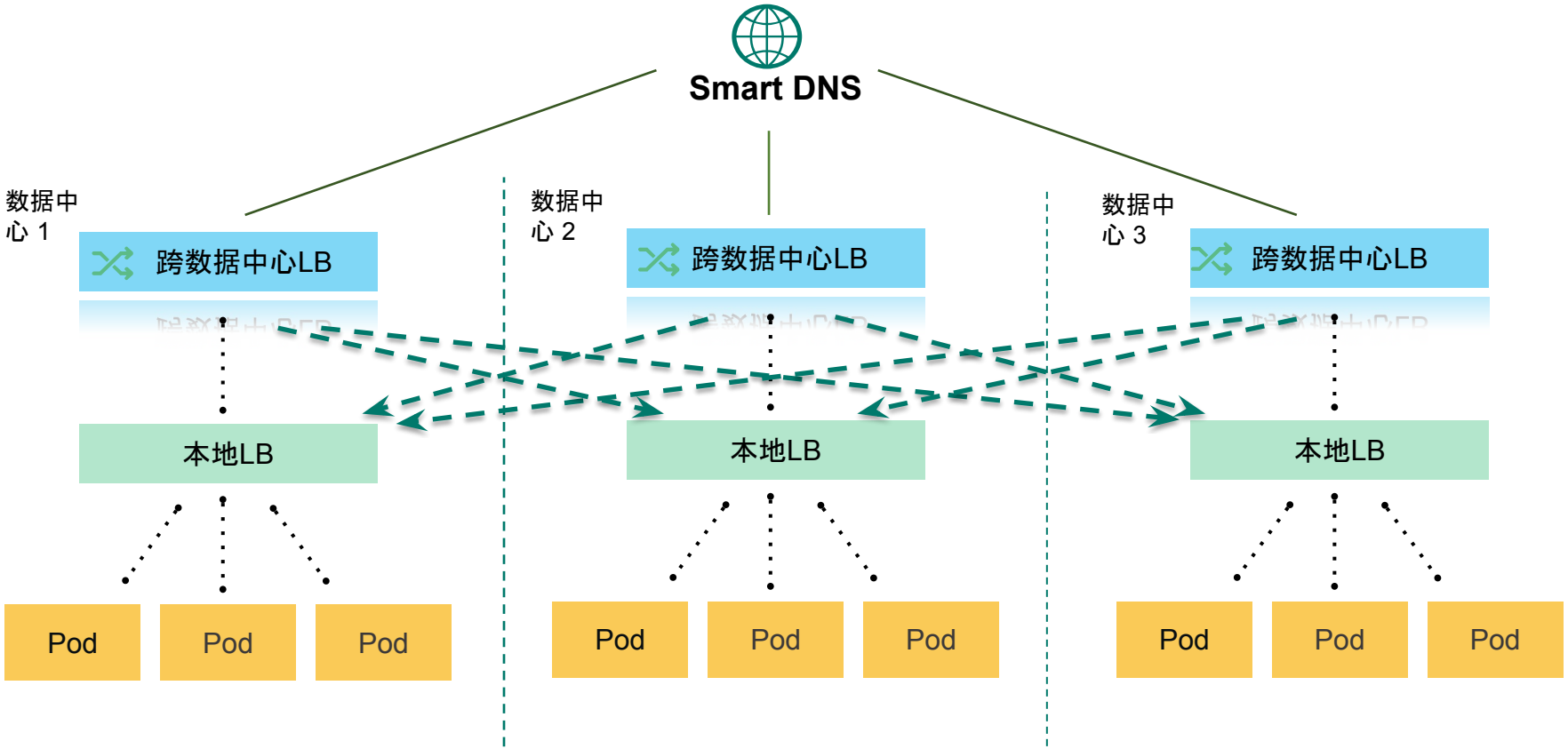
价格具有时效性

Kubernetes生产化实践之路

京东

长按或扫描查看

#IstioCon


Smart DNS
数据中心 1
跨数据中心LB
本地LB
Pod
Pod
Pod
数据中心 2
跨数据中心LB
本地LB
Pod
Pod
Pod
数据中心 3
跨数据中心LB
本地LB
Pod
Pod
Pod