扫码添加小助手，发送“Istio”加群

关注Istio知乎专栏，学技术不迷路

CloudNativeLives

Istio入门级实训

Istio 架构与技术

华为云容器团队核心架构师 & CNCF社区主要贡献者倾力打造

大纲

- Service Mesh
- Istio 架构基础
- Istio 基本概念
- Istio & Kubernetes：架构结合
- 运行第一个Istio集群

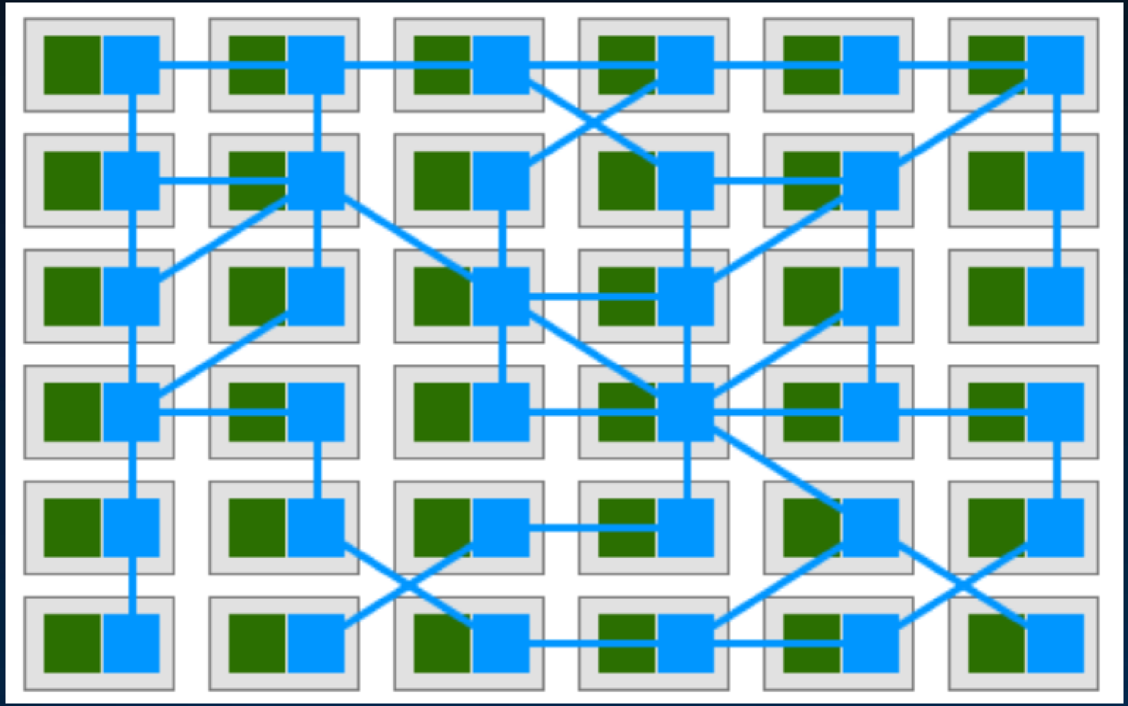
Kubernetes

- Kubernetes 提供平台基础设施层强大的容器编排与调度能力
  - 服务部署与弹性伸缩：Deployment
  - 服务拆分与服务发现：Service

- Kubernetes 提供简单的负载均衡
  - 负载均衡：基于IPVS或Iptables的简单均衡机制

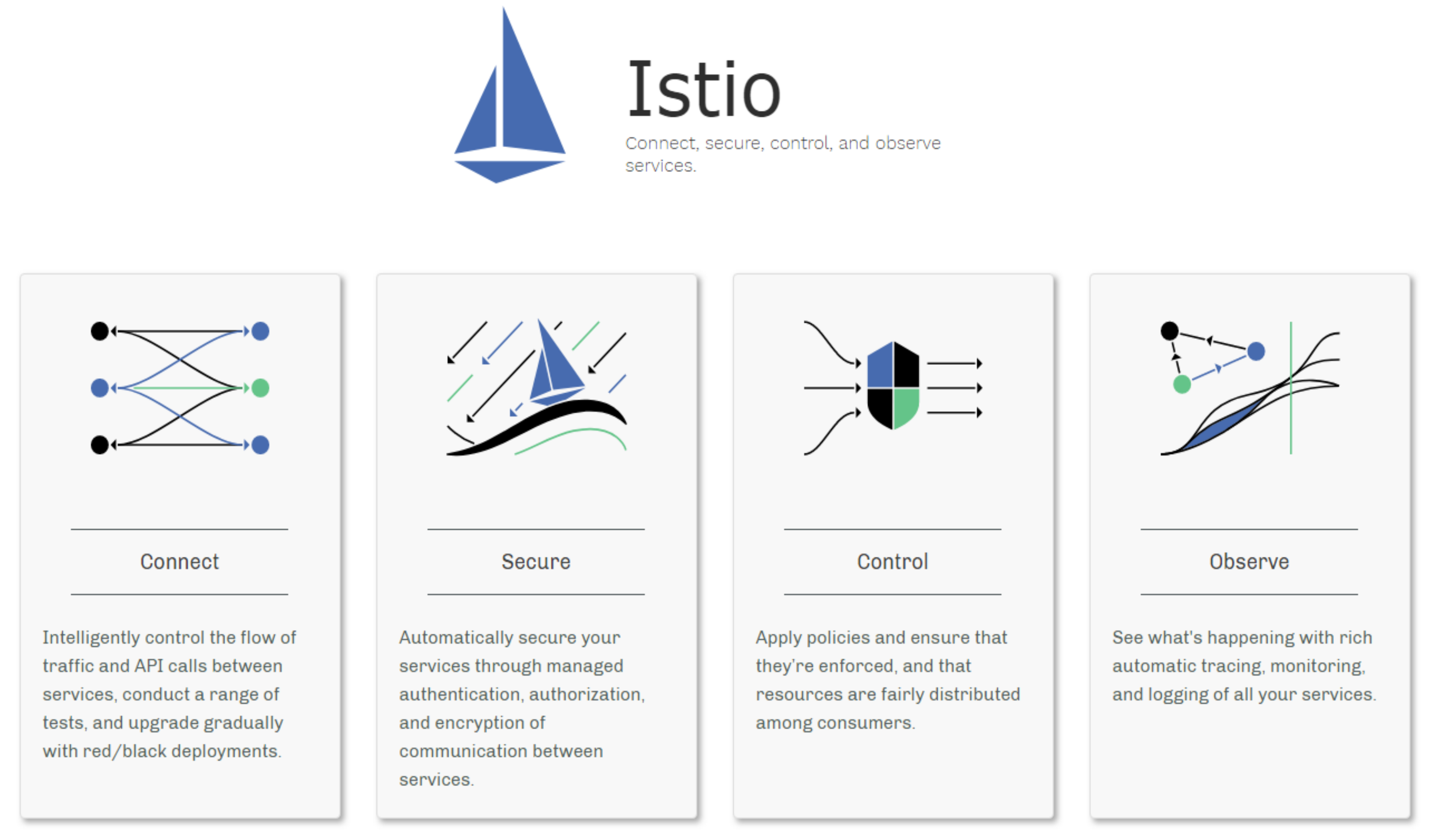
Service Mesh

- 治理能力独立（Sidecar）
- 应用程序无感知
- 服务通信的基础设施层



Istio问世

- 连接（Connect）
- 安全（Secure）
- 控制（Control）
- 观察（Observe）



Istio

Connect, secure, control, and observe services.

Connect

Intelligently control the flow of traffic and API calls between services, conduct a range of tests, and upgrade gradually with red/black deployments.

Secure

Automatically secure your services through managed authentication, authorization, and encryption of communication between services.

Control

Apply policies and ensure that they're enforced, and that resources are fairly distributed among consumers.

Observe

See what's happening with rich automatic tracing, monitoring, and logging of all your services.

Istio关键能力

功能

流量管理

- 负载均衡
- 动态路由
- 灰度发布
- 故障注入

可观察性

- 调用链
- 访问日志
- 监控

策略执行

- 限流
- ACL

服务身份和安全

- 认证
- 鉴权

扩展

平台支持

- Kubernetes
- CloudFoundry
- Eureka
- Consul

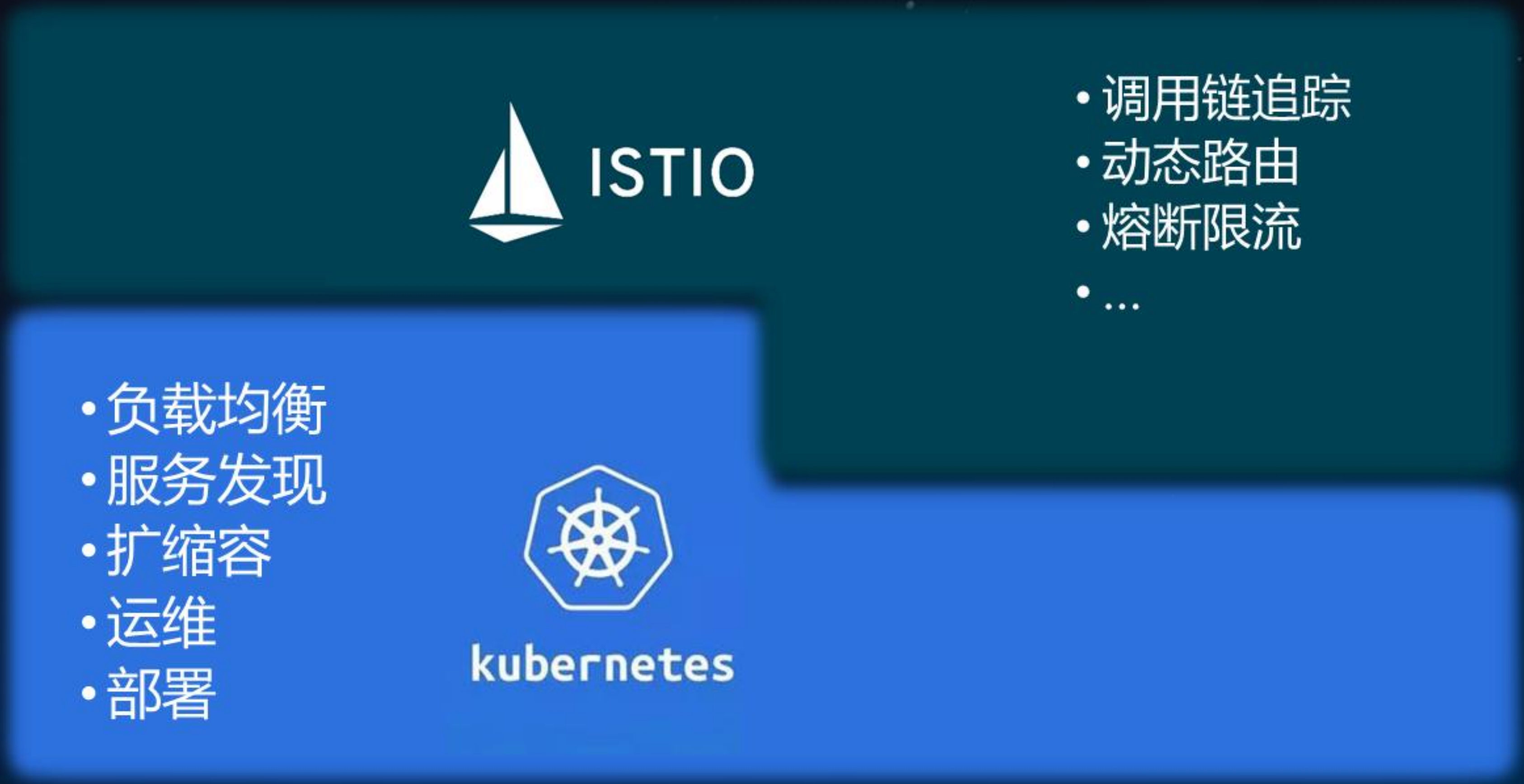
集成和定制

- ACL
- 日志
- 配额

Istio + Kubernetes：云原生应用治理 + 云原生应用设施

服务治理

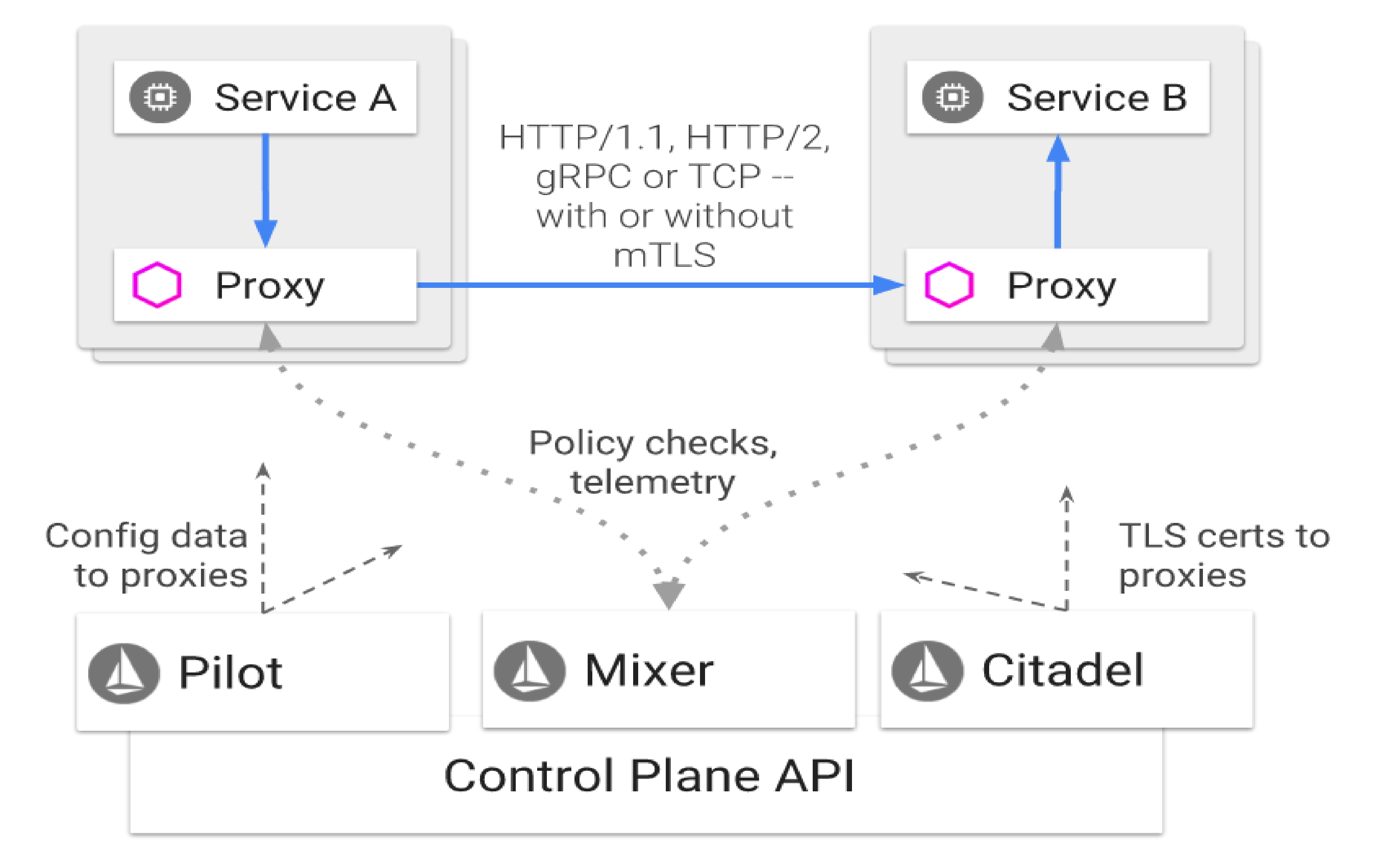
部署运维



- 调用链追踪
- 动态路由
- 熔断限流
- …

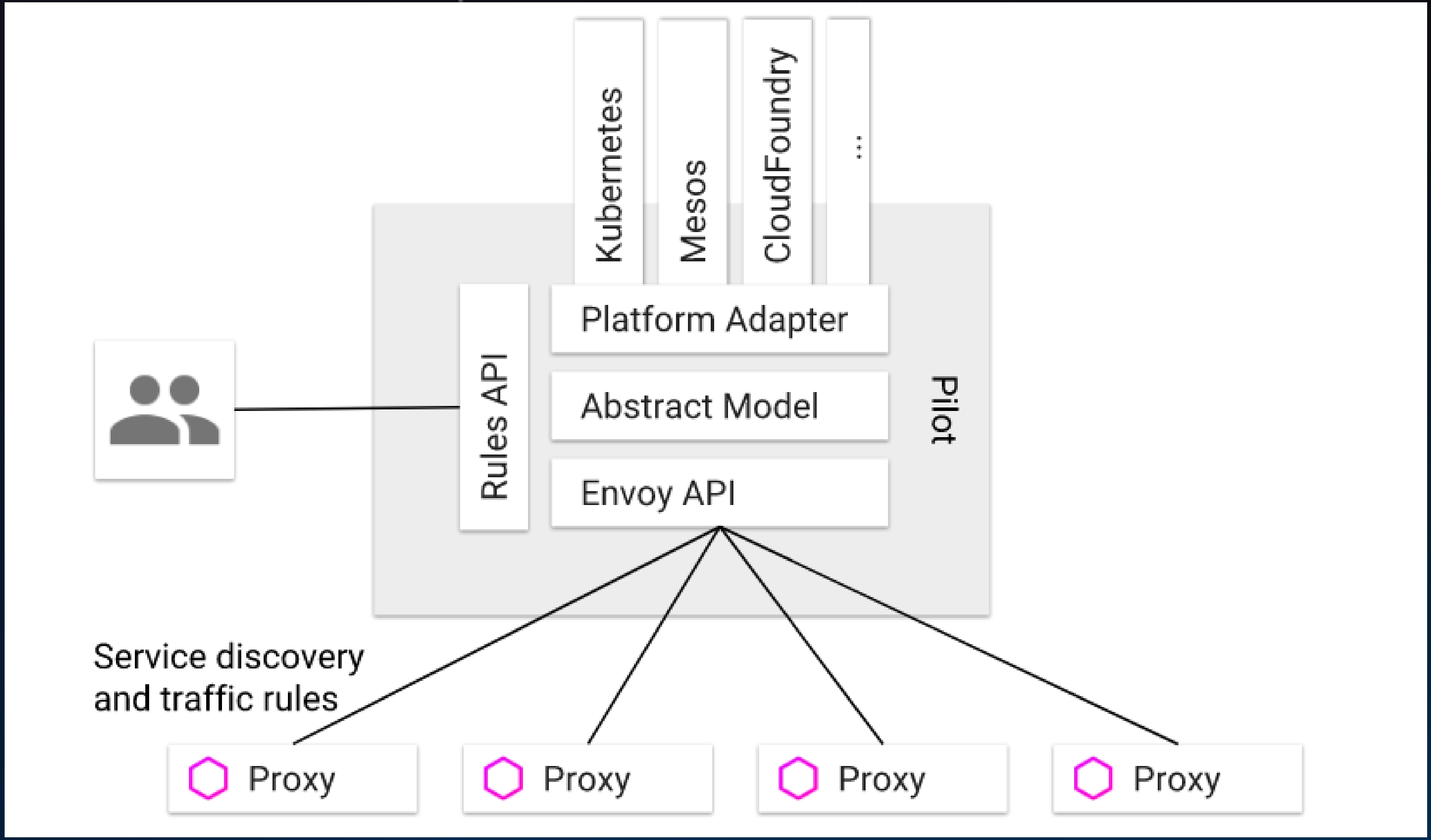
- 负载均衡
- 服务发现
- 扩缩容
- 运维
- 部署

Istio架构



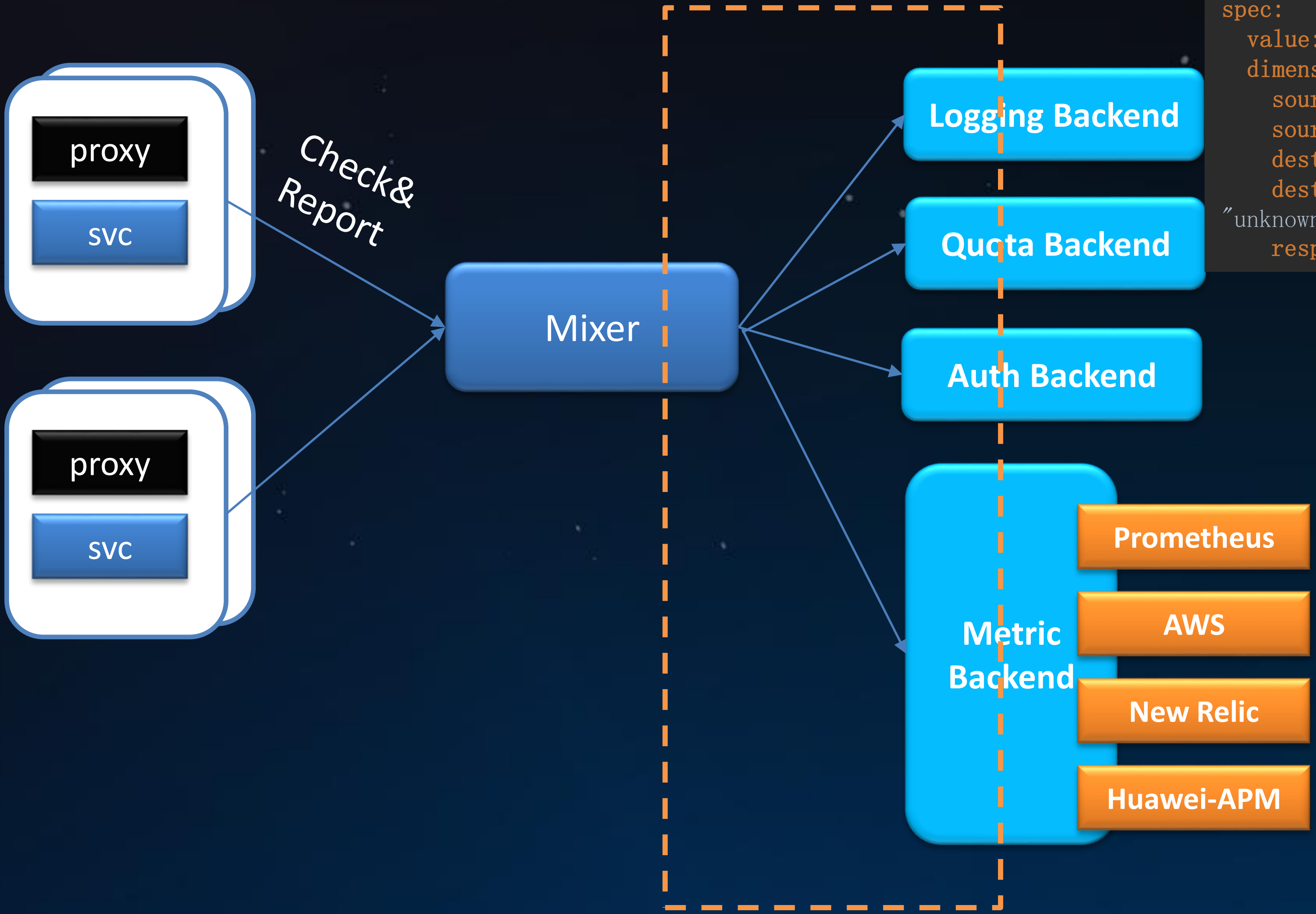
Service A
Proxy
HTTP/1.1, HTTP/2, gRPC or TCP -- with or without mTLS
Service B
Proxy
Policy checks, telemetry
Config data to proxies
TLS certs to proxies
Pilot
Mixer
Citadel
Control Plane API

Pilot


Kubernetes
Mesos
CloudFoundry
...
Platform Adapter
Abstract Model
Envoy API
Rules API
Pilot
Service discovery
and traffic rules
Proxy
Proxy
Proxy
Proxy

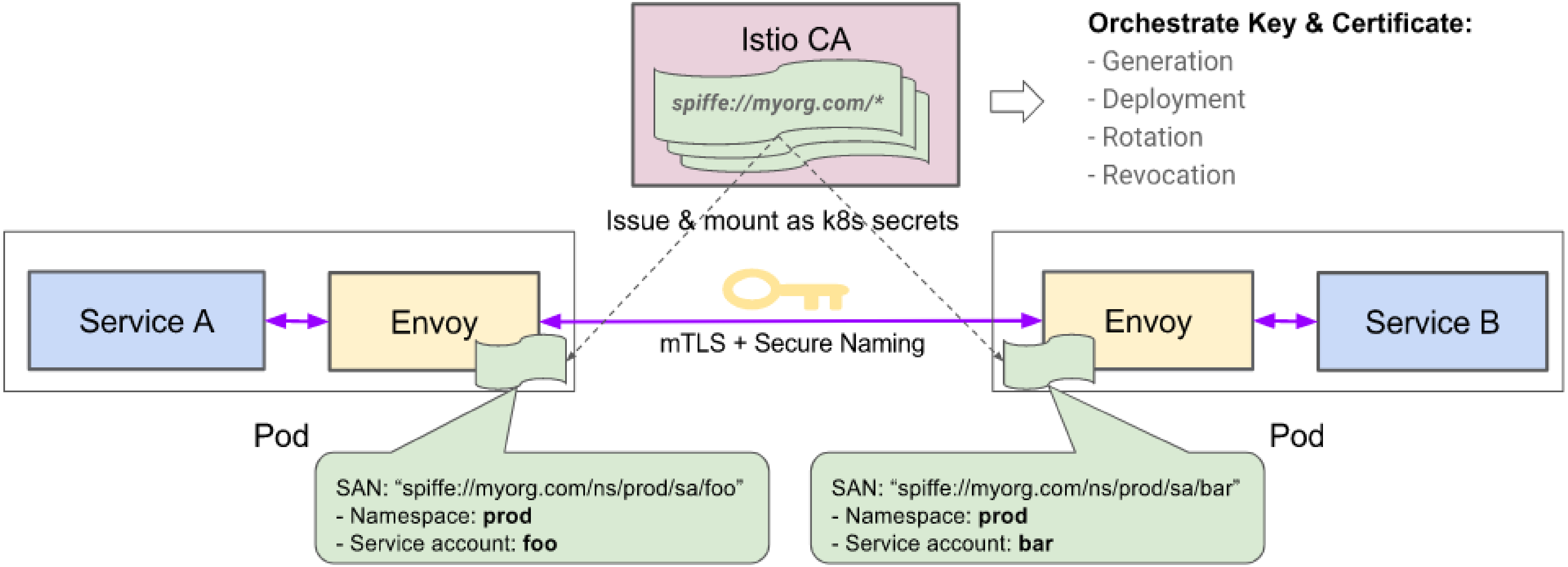
Mixer

```
apiVersion: "config.istio.io/v1alpha2"
kind: metric
metadata:
  name: requestduration
  namespace: istio-system
spec:
  value: response.duration | "0ms"
  dimensions:
    source_service: source.service | "unknown"
    source_version: source.labels["version"] | "unknown"
    destination_service: destination.service | "unknown"
    destination_version: destination.labels["version"] |
"unknown"
    response_code: response.code | 200
```



Citadel

HUAWEI

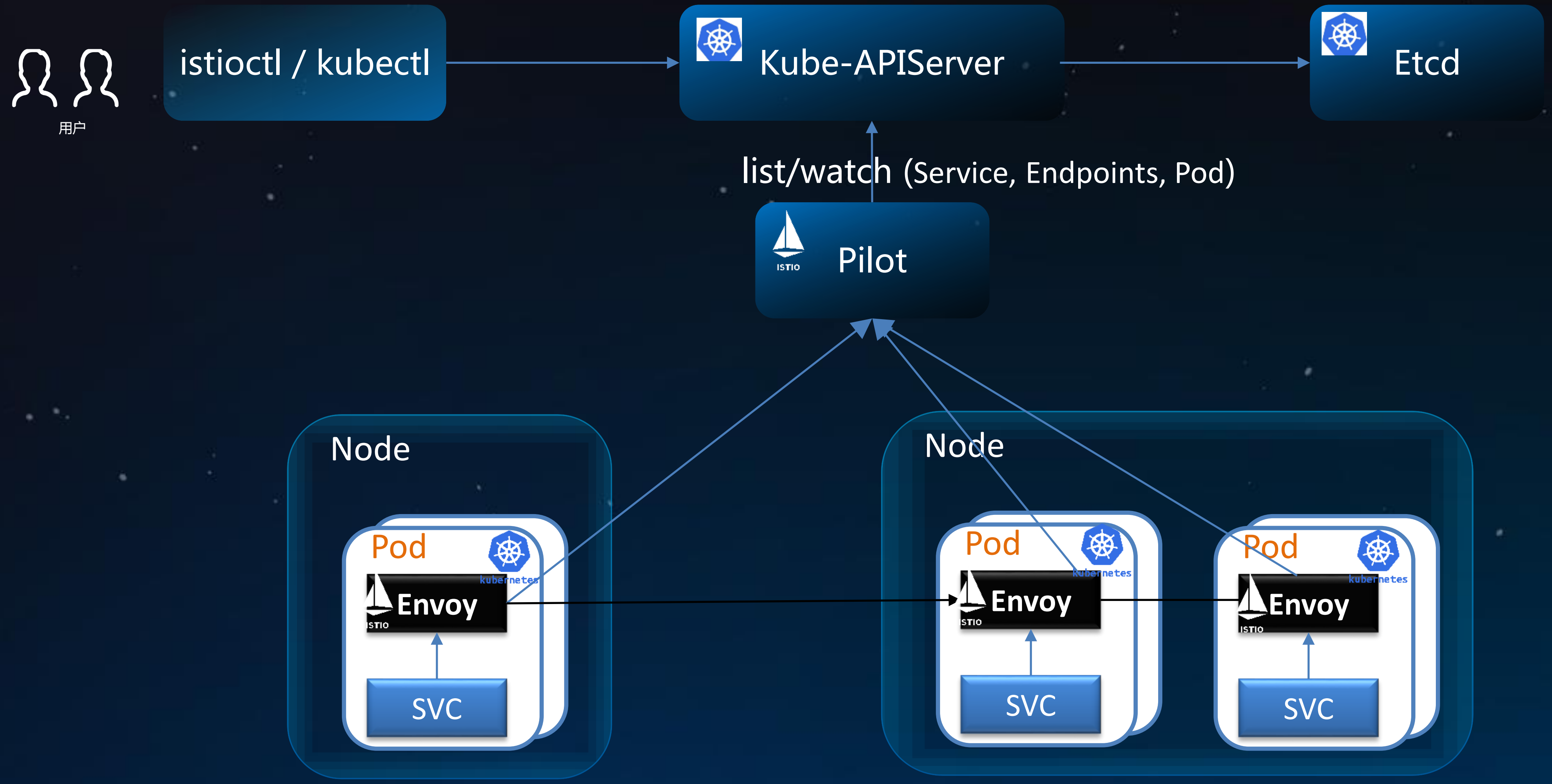


Orchestrate Key & Certificate:
- Generation
- Deployment
- Rotation
- Revocation

SAN: "spiffe://myorg.com/ns/prod/sa/foo"
- Namespace: **prod**
- Service account: **foo**

SAN: "spiffe://myorg.com/ns/prod/sa/bar"
- Namespace: **prod**
- Service account: **bar**

Istio & Kubernetes：架构结合

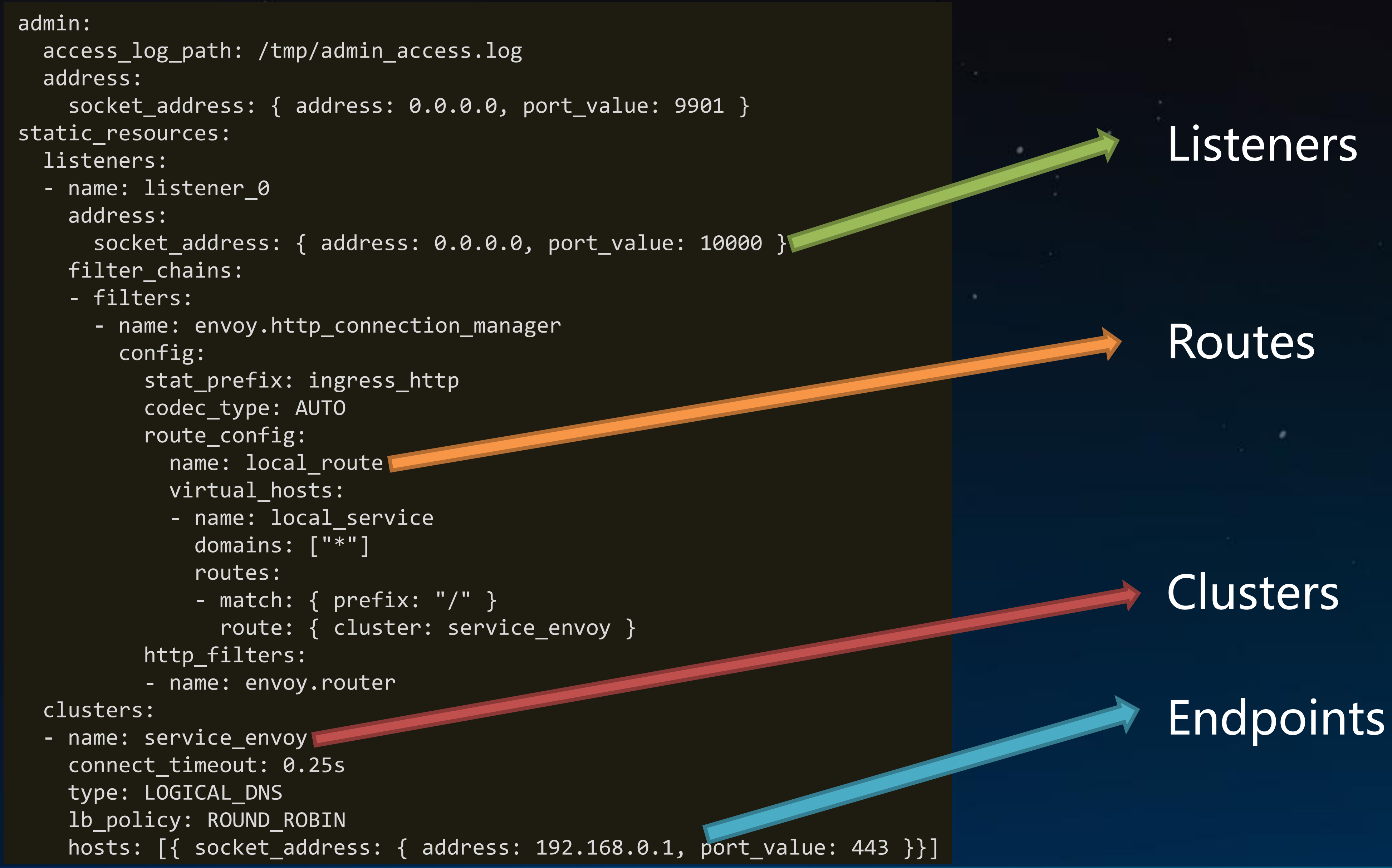


Envoy

- 基于C++的 L4/L7 Proxy转发器
- CNCF第三个毕业的项目

• Listeners	(LDS)
• Routes	(RDS)
• Clusters	(CDS)
• Endpoints	(EDS)

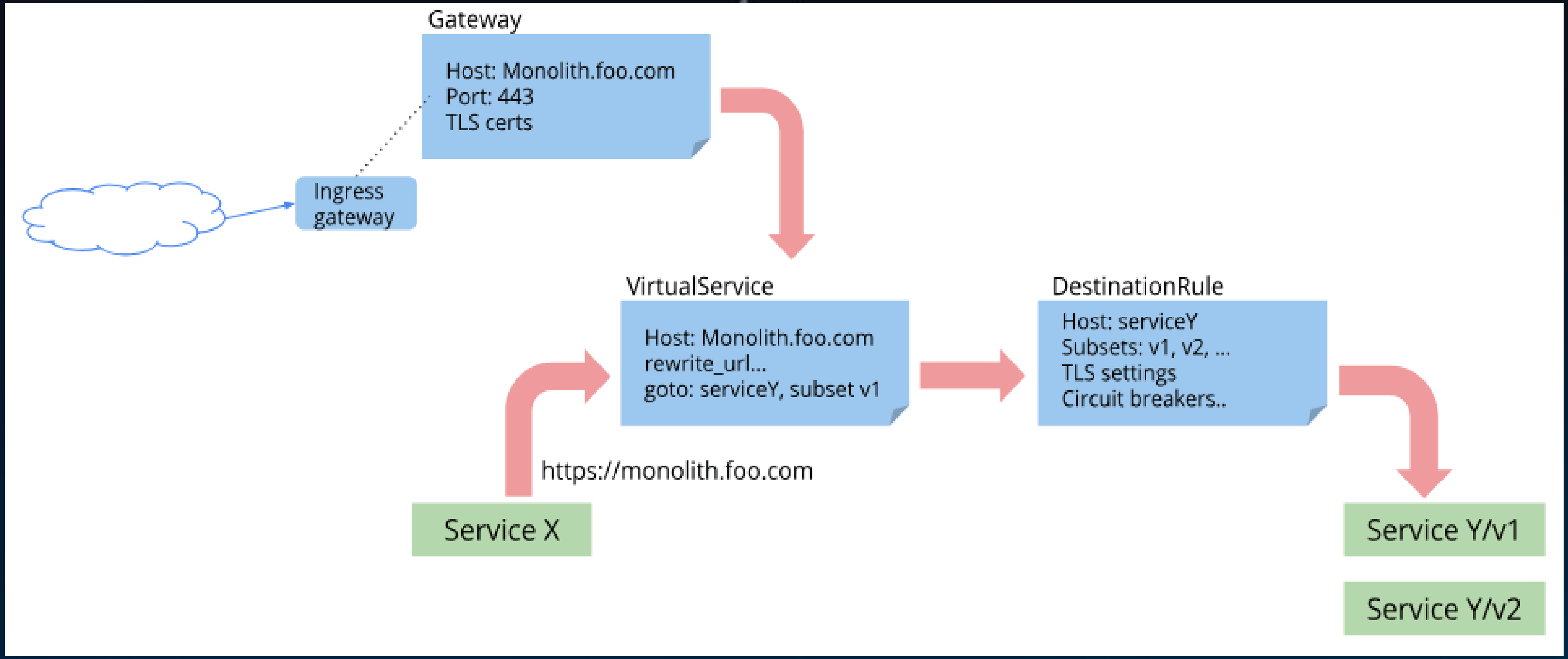
Envoy 配置文件



```
admin:
  access_log_path: /tmp/admin_access.log
  address:
    socket_address: { address: 0.0.0.0, port_value: 9901 }
static_resources:
  listeners:
  - name: listener_0
    address:
      socket_address: { address: 0.0.0.0, port_value: 10000 }
    filter_chains:
    - filters:
      - name: envoy.http_connection_manager
        config:
          stat_prefix: ingress_http
          codec_type: AUTO
          route_config:
            name: local_route
            virtual_hosts:
            - name: local_service
              domains: ["*"]
              routes:
              - match: { prefix: "/" }
                route: { cluster: service_envoy }
          http_filters:
          - name: envoy.router
  clusters:
  - name: service_envoy
    connect_timeout: 0.25s
    type: LOGICAL_DNS
    lb_policy: ROUND_ROBIN
    hosts: [{ socket_address: { address: 192.168.0.1, port_value: 443 }}]
```

Istio 基础概念

- Gateway
- VirtualService
- DestinationRule
- ServiceEntry



VirtualService

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews-route
  namespace: foo
spec:
  hosts:
  - reviews
  http:
  - match:
    - uri:
        prefix: "/wpcatalog"
    - uri:
        prefix: "/consumercatalog"
    rewrite:
      uri: "/newcatalog"
    route:
    - destination:
        host: reviews
        subset: v2
  - route:
    - destination:
        host: reviews
        subset: v1
```

最核心的配置接口，定义指定服务的所有路由规则

- Hosts
- Gateways
- Http
- Tcp
- Tls

DestinationRule

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: bookinfo-ratings
spec:
  host: ratings
  trafficPolicy:
    loadBalancer:
      simple: LEAST_CONN
  subsets:
  - name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
  - name: v3
    labels:
      version: v3
    trafficPolicy:
      loadBalancer:
        simple: ROUND_ROBIN
```

其定义的策略，决定了路由处理之后的流量访问策略。负载均衡设置，断路器，TLS设置等

- Host
- Subset
- TrafficPolicy

Gateway

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: bookinfo-gateway
spec:
  selector:
    istio: ingressgateway # use istio default controller
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "*"
---
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: bookinfo
spec:
  hosts:
  - "*"
  gateways:
  - bookinfo-gateway
  http:
  - match:
    - uri:
        exact: /productpage
  route:
    - destination:
        ...
```

提供外部服务访问接入，可发布任意内部端口的服务，供外部访问。配合VirtualService使用，使用标准Istio规则治理

- Servers
- Selector

ServiceEntry

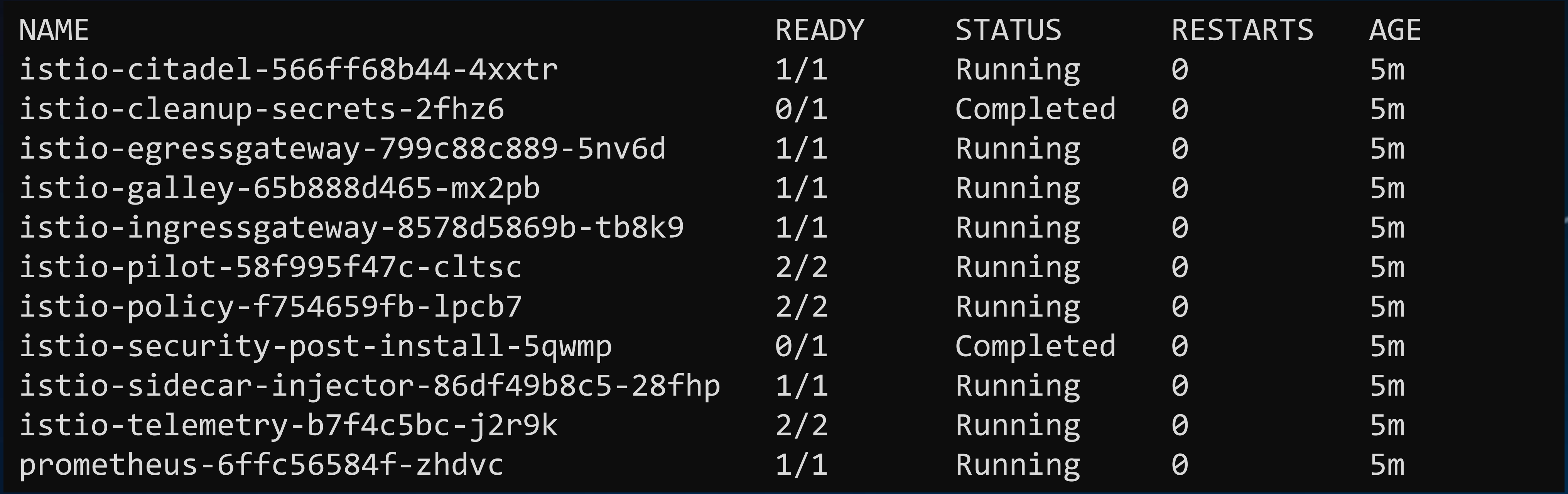
```
apiVersion: networking.istio.io/v1alpha3
kind: ServiceEntry
metadata:
  name: external-svc-mongocluster
spec:
  hosts:
  - mymongodb.somedomain
  addresses:
  - 192.192.192.192/24 # VIPs
  ports:
  - number: 27018
    name: mongodb
    protocol: MONGO
  location: MESH_INTERNAL
  resolution: STATIC
  endpoints:
  - address: 2.2.2.2
  - address: 3.3.3.3
----
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: mtls-mongocluster
spec:
  host: mymongodb.somedomain
  trafficPolicy:
    tls:
      mode: MUTUAL
      clientCertificate: /etc/certs/myclientcert.pem
      privateKey: /etc/certs/client_private_key.pem
      caCertificates: /etc/certs/rootcacerts.pem
```

将外部服务接入到服务注册表中，让Istio中自动发现的服务能够访问和路由到这些手工加入的服务。与VirtualService或DestinationRule配合使用

- Hosts
- Addresss
- Ports
- Location
- Resolution
- Endpoints

基于K8s运行Istio集群

- kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml
- helm template install/kubernetes/helm/istio --name istio --namespace istio-system > $HOME/istio.yaml
- kubectl create namespace istio-system
- kubectl apply -f $HOME/istio.yaml

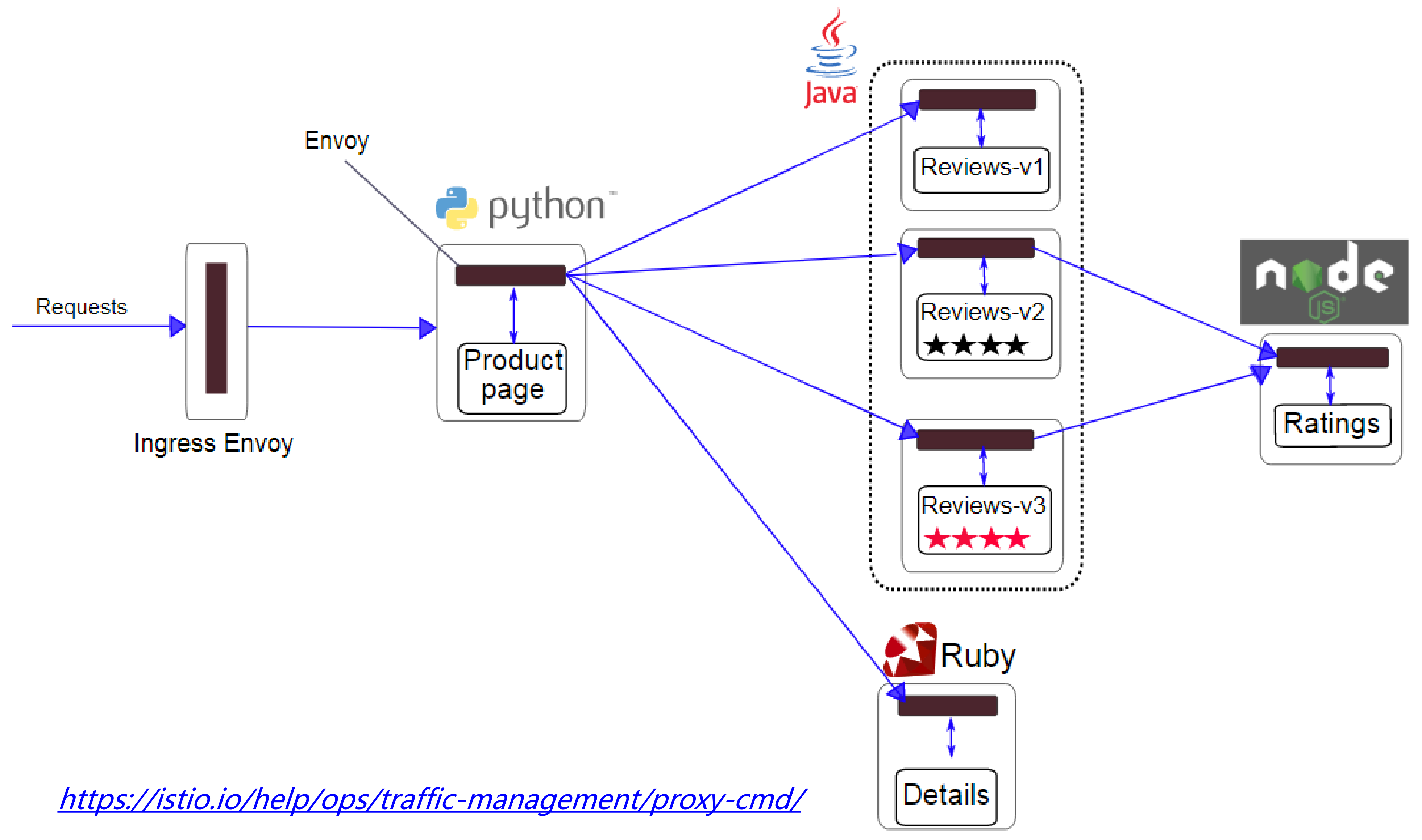


```
NAME                                      READY     STATUS      RESTARTS   AGE
istio-citadel-566ff68b44-4xxtr            1/1       Running     0          5m
istio-cleanup-secrets-2fhz6               0/1       Completed   0          5m
istio-egressgateway-799c88c889-5nv6d      1/1       Running     0          5m
istio-galley-65b888d465-mx2pb             1/1       Running     0          5m
istio-ingressgateway-8578d5869b-tb8k9     1/1       Running     0          5m
istio-pilot-58f995f47c-cltsc              2/2       Running     0          5m
istio-policy-f754659fb-lpcb7              2/2       Running     0          5m
istio-security-post-install-5qwmp         0/1       Completed   0          5m
istio-sidecar-injector-86df49b8c5-28fhp   1/1       Running     0          5m
istio-telemetry-b7f4c5bc-j2r9k            2/2       Running     0          5m
prometheus-6ffc56584f-zhdvc               1/1       Running     0          5m
```

Istioctl

- proxy-status：状态同步情况
- proxy-config：envoy中具体规则查询
  - listener
  - route
  - cluster
  - endpoint
- kube-inject
- ……

Demo



https://istio.io/help/ops/traffic-management/proxy-cmd/

HUAWEI

Thank You

直播　每周四 晚20:00