IBM Developer

IBM

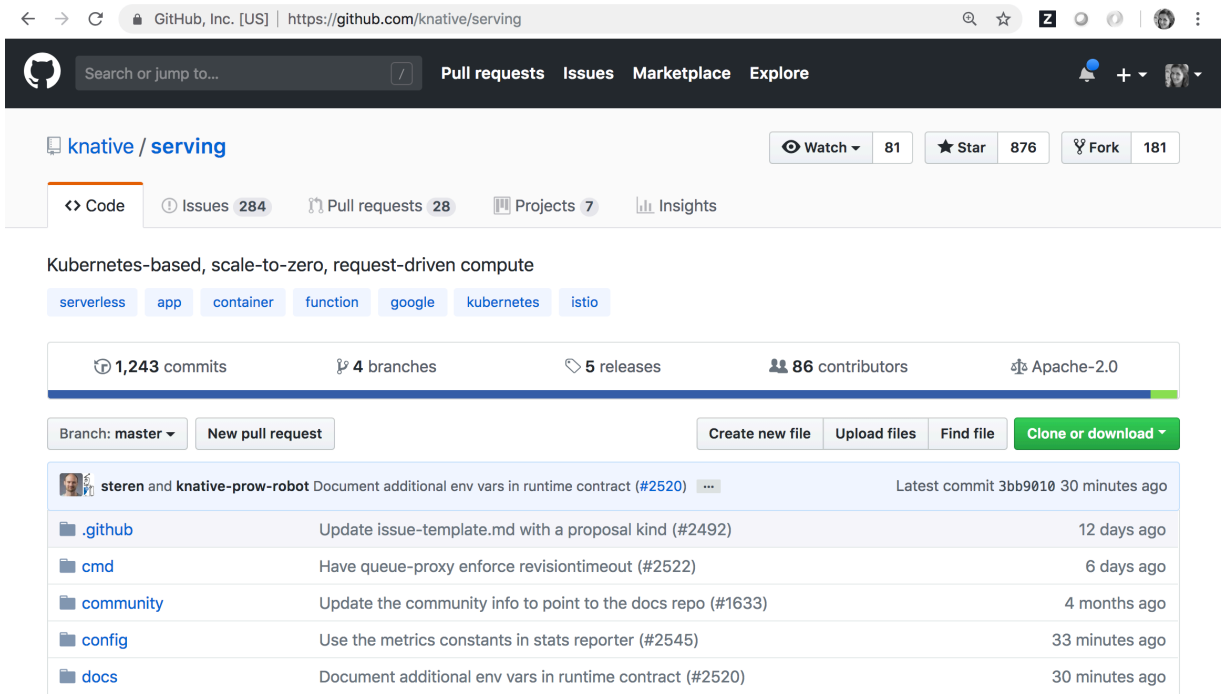
Knative Serving: 将微服务从0扩展到无限

邱见
软件工程师

IBM开发者技术沙龙

Serving

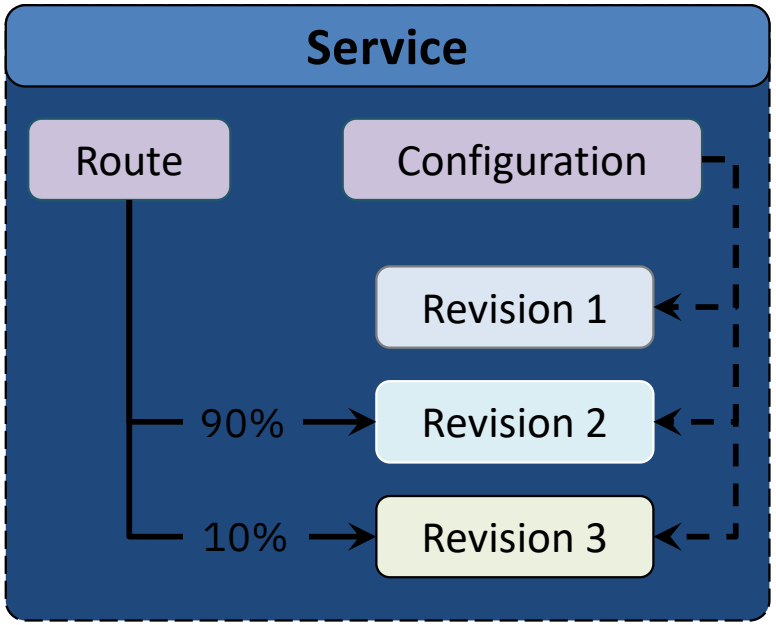
- 自动伸缩
- 路由和网络编程
- 实时指向部署的代码和配置
- 滚动升级，A/B测试

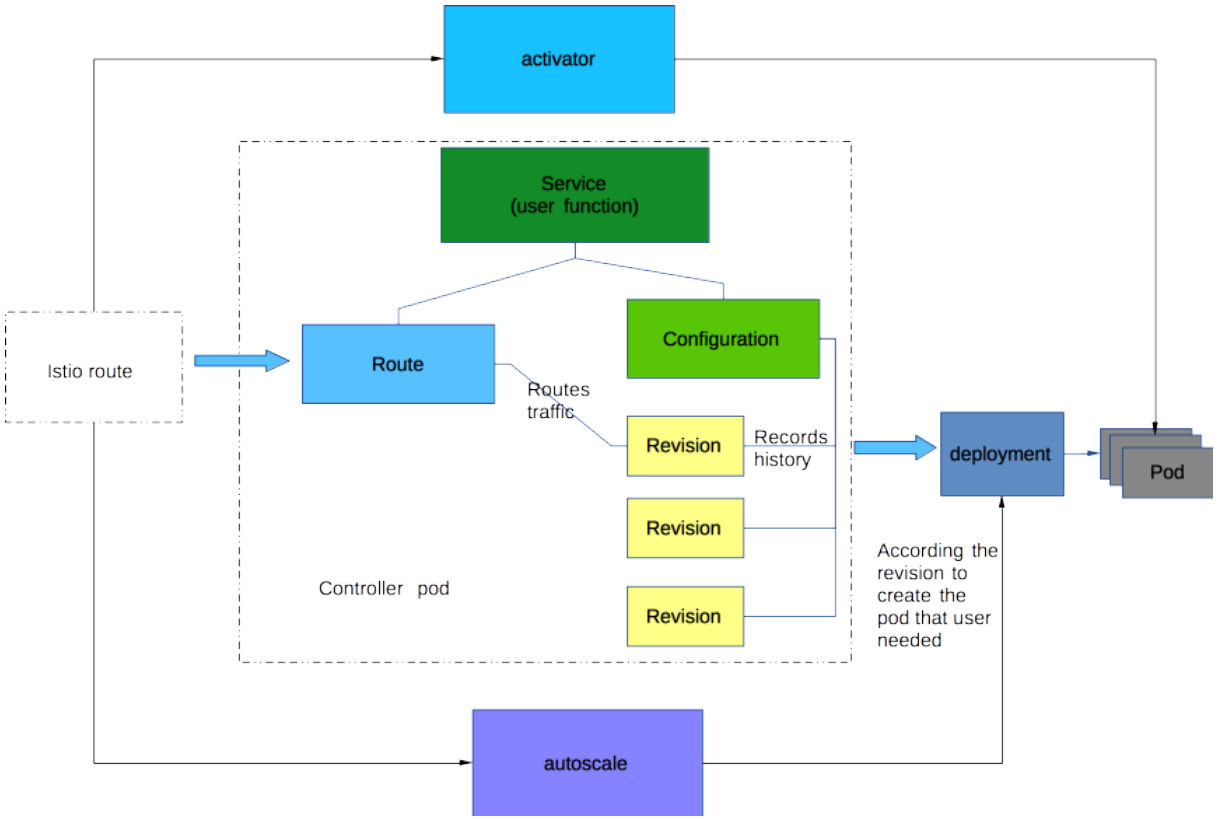


Source: https://github.com/knative/serving

API

- **Route**: 定义网络端口，映射到一个或多个revision
- **Revision**: 每次修改代码和配置产生的快照，不可更改。每个revision对应一次部署，并可以根据流量自动扩展
- **Configuration**: 维护部署的期望状态，每次修改configuration产生一个新的revision
- **Service**: 管理应用的生命周期，确保应用拥有configuration和route，并可以定义应用使请求导向特定的revision




activator
Service
(user function)
Istio route
Route
Configuration
Routes
traffic
Revision
Records
history
Revision
Revision
Controller pod
deployment
Pod
According the
revision to
create the
pod that user
needed
autoscale

IBM **Developer**

IBM

Hello world

```yaml
apiVersion: serving.knative.dev/v1beta1
kind: Service
metadata:
  name: helloworld-go
  namespace: default
spec:
  template:
    spec:
      containers:
      - image: gcr.io/knative-samples/helloworld-go
        env:
        - name: TARGET
          value: "Go Sample v1"
```

route

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Route
metadata:
  labels:
    serving.knative.dev/service: helloworld-go
  name: helloworld-go
  namespace: default
spec:
  traffic:
  - configurationName: helloworld-go
    latestRevision: true
    percent: 100
status:
  address:
    hostname: helloworld-go.default.svc.cluster.local
    url: http://helloworld-go.default.svc.cluster.local
  domain: helloworld-go.default.example.com
  domainInternal: helloworld-go.default.svc.cluster.local
  traffic:
  - latestRevision: true
    percent: 100
    revisionName: helloworld-go-6dgn9
  url: http://helloworld-go.default.example.com
```

configuration

```yaml
kind: Configuration
metadata:
  labels:
    serving.knative.dev/route: helloworld-go
    serving.knative.dev/service: helloworld-go
  name: helloworld-go
  namespace: default
spec:
  template:
    spec:
      containers:
      - env:
        - name: TARGET
          value: Go Sample v1
        image: gcr.io/knative-samples/helloworld-go
        name: ""
        resources: {}
      timeoutSeconds: 300
status:
  latestCreatedRevisionName: helloworld-go-6dgn9
  latestReadyRevisionName: helloworld-go-6dgn9
```

revision

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Revision
metadata:
  annotations:
    serving.knative.dev/lastPinned: "1561099012"
  labels:
    serving.knative.dev/configuration: helloworld-go
    serving.knative.dev/configurationGeneration: "1"
    serving.knative.dev/service: helloworld-go
  name: helloworld-go-6dgn9
  namespace: default
spec:
  containers:
  - env:
    - name: TARGET
      value: Go Sample v1
    image: gcr.io/knative-samples/helloworld-go
  timeoutSeconds: 300
status:
  imageDigest: gcr.io/knative-samples/helloworld-go@sha256:
  logUrl: http://localhost:8001/api/v1/namespaces/knative-m
  serviceName: helloworld-go-6dgn9
```

IBM开发者技术沙龙

revision

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Revision
metadata:
  annotations:
    serving.knative.dev/lastPinned: "1561099012"
  labels:
    serving.knative.dev/configuration: helloworld-go
    serving.knative.dev/configurationGeneration: "1"
    serving.knative.dev/service: helloworld-go
  name: helloworld-go-6dgn9
  namespace: default
spec:
  containers:
  - env:
    - name: TARGET
      value: Go Sample v1
    image: gcr.io/knative-samples/helloworld-go
  timeoutSeconds: 300
status:
  imageDigest: gcr.io/knative-samples/helloworld-go@sha256:
  logUrl: http://localhost:8001/api/v1/namespaces/knative-m
  serviceName: helloworld-go-6dgn9
```

deployment

```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  labels:
    app: helloworld-go-6dgn9
    serving.knative.dev/configuration: helloworld-go
    serving.knative.dev/configurationGeneration: "1"
    serving.knative.dev/revision: helloworld-go-6dgn9
    serving.knative.dev/revisionUID: 84062375-a56f-4e6c-97
    serving.knative.dev/service: helloworld-go
  name: helloworld-go-6dgn9-deployment
  namespace: default
spec:
  progressDeadlineSeconds: 120
  replicas: 0
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      serving.knative.dev/revisionUID: 84062375-a56f-4e6c-
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      annotations:
        sidecar.istio.io/inject: "true"
        traffic.sidecar.istio.io/includeOutboundIPRanges:
```

autoscaler

```yaml
apiVersion: autoscaling.internal.knative.dev/v1alpha1
kind: PodAutoscaler
metadata:
  labels:
    app: helloworld-go-6dgn9
    serving.knative.dev/configuration: helloworld-go
    serving.knative.dev/configurationGeneration: "1"
    serving.knative.dev/revision: helloworld-go-6dgn9
    serving.knative.dev/revisionUID: 84062375-a56f-4e6c-9704-f
    serving.knative.dev/service: helloworld-go
  name: helloworld-go-6dgn9
  namespace: default
spec:
  ProtocolType: http1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: helloworld-go-6dgn9-deployment
  serviceName: ""
status:
  serviceName: helloworld-go-6dgn9
```

serverlessservice

```yaml
apiVersion: networking.internal.knative.dev/v1alpha1
kind: ServerlessService
metadata:
  annotations:
    autoscaling.knative.dev/class: kpa.autoscaling.knative.de
  creationTimestamp: "2019-06-21T06:35:52Z"
  generation: 2
  labels:
    app: helloworld-go-6dgn9
    serving.knative.dev/configuration: helloworld-go
    serving.knative.dev/configurationGeneration: "1"
    serving.knative.dev/revision: helloworld-go-6dgn9
    serving.knative.dev/revisionUID: 84062375-a56f-4e6c-9704-
    serving.knative.dev/service: helloworld-go
  name: helloworld-go-6dgn9
  namespace: default
spec:
  ProtocolType: http1
  mode: Proxy
  objectRef:
    apiVersion: apps/v1
    kind: Deployment
    name: helloworld-go-6dgn9-deployment
status:
  privateServiceName: helloworld-go-6dgn9-priv
  serviceName: helloworld-go-6dgn9
```

route

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Route
metadata:
  labels:
    serving.knative.dev/service: helloworld-go
  name: helloworld-go
  namespace: default
spec:
  traffic:
  - configurationName: helloworld-go
    latestRevision: true
    percent: 100
status:
  address:
    hostname: helloworld-go.default.svc.cluster.local
    url: http://helloworld-go.default.svc.cluster.local
  domain: helloworld-go.default.example.com
  domainInternal: helloworld-go.default.svc.cluster.local
  traffic:
  - latestRevision: true
    percent: 100
    revisionName: helloworld-go-6dgn9
  url: http://helloworld-go.default.example.com
```

clusteringress

```yaml
apiVersion: networking.internal.knative.dev/v1alpha1
kind: ClusterIngress
metadata:
  annotations:
    networking.knative.dev/ingress.class: istio.ingress.netwo
    serving.knative.dev/creator: kubernetes-admin
    serving.knative.dev/lastModifier: kubernetes-admin
  labels:
    serving.knative.dev/route: helloworld-go
    serving.knative.dev/routeNamespace: default
  name: route-3c799700-121b-4f18-a746-a02e5a04cb6d
spec:
  rules:
  - hosts:
    - helloworld-go.default.example.com
    - helloworld-go.default.svc.cluster.local
    http:
      paths:
      - appendHeaders:
          knative-serving-namespace: default
          knative-serving-revision: helloworld-go-6dgn9
        retries:
          attempts: 3
          perTryTimeout: 10m0s
        splits:
        - percent: 100
          serviceName: helloworld-go-6dgn9
```

virtualservice

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  networking.knative.dev/ingress.class: istio.ingress.
    serving.knative.dev/creator: kubernetes-admin
    serving.knative.dev/lastModifier: kubernetes-admin
  labels:
    networking.internal.knative.dev/clusteringress: ro
    serving.knative.dev/route: helloworld-go
    serving.knative.dev/routeNamespace: default
  name: route-3c799700-121b-4f18-a746-a02e5a04cb6d
  namespace: knative-serving
spec:
  gateways:
  - knative-ingress-gateway
  - mesh
  hosts:
  - helloworld-go.default.example.com
  - helloworld-go.default.svc.cluster.local
  http:
  - appendHeaders:
      knative-serving-namespace: default
      knative-serving-revision: helloworld-go-6dgn9
    match:
    - authority:
        regex: ^helloworld-go\.default(?::\d{1,5})?$
```

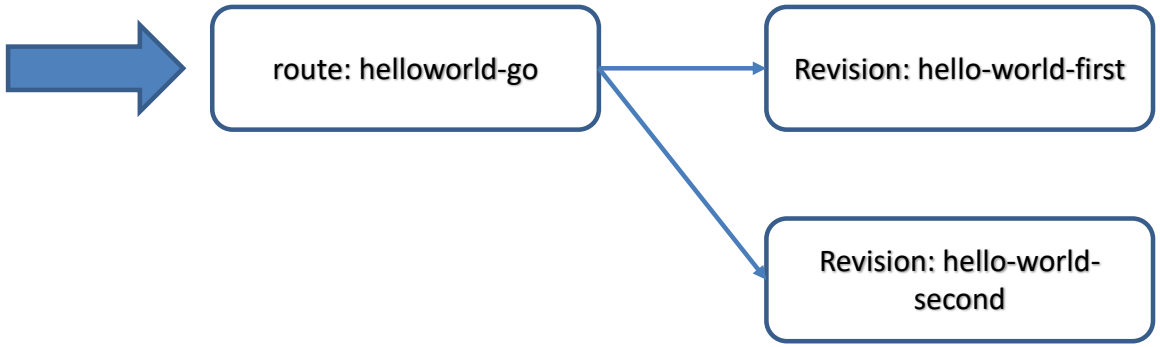
Traffic splitting

Helloworld v1

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: helloworld-go
  namespace: default
spec:
  template:
    metadata:
      name: helloworld-go-first
    spec:
      containers:
      - image: gcr.io/knative-samples/helloworld-go
        env:
        - name: TARGET
          value: "Go Sample v1"
  traffic:
  - tag: current
    revisionName: helloworld-go-first
    percent: 100
  - tag: latest
    latestRevision: true
    percent: 0
```

Helloworld v2

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: helloworld-go
  namespace: default
spec:
  template:
    metadata:
      name: helloworld-go-second
    spec:
      containers:
      - image: gcr.io/knative-samples/helloworld-go
        env:
        - name: TARGET
          value: "Go Sample v2"
  traffic:
  - tag: current
    revisionName: helloworld-go-first
    percent: 50
  - tag: candidate
    revisionName: helloworld-go-second
    percent: 50
  - tag: latest
    latestRevision: true
    percent: 0
```



How does scale happen?

- Autoscaler metrics
  - 每个revision通过metrics service暴露自己的metrics
  - Autoscaler 分析metrics值修改revision对应的deployment replica数量
- KPA & HPA
  - KPA 可以根据并发数进行伸缩

```
annotations:
  # Knative concurrency-based autoscaling (default).
  autoscaling.knative.dev/class: kpa.autoscaling.knative.dev
  autoscaling.knative.dev/metric: concurrency
  # Target 10 requests in-flight per pod.
  autoscaling.knative.dev/target: "10";
  # Disable scale to zero with a minScale of 1.
  autoscaling.knative.dev/minScale: "1";
  # Limit scaling to 100 pods.
  autoscaling.knative.dev/maxScale: "100";
```

  - HPA 根据HPA metrics 如CPU进行伸缩

```
annotations:
  # Standard Kubernetes CPU-based autoscaling.
  autoscaling.knative.dev/class: hpa.autoscaling.knative.dev
  autoscaling.knative.dev/metric: cpu
```

Scale to 0

- 当没有request后， Autoscaler将deployment的replica缩减为0
  - scale-to-zero-grace-period
  - stable-window

- Activator
  - Replica 为 0 时， activator修改service endpoint将流量导入activator
  - 新请求到达时， activator增加deployment的replica并转发流量
  - 当容器完全拉起后， activator修改endpoint将流量直接导入service

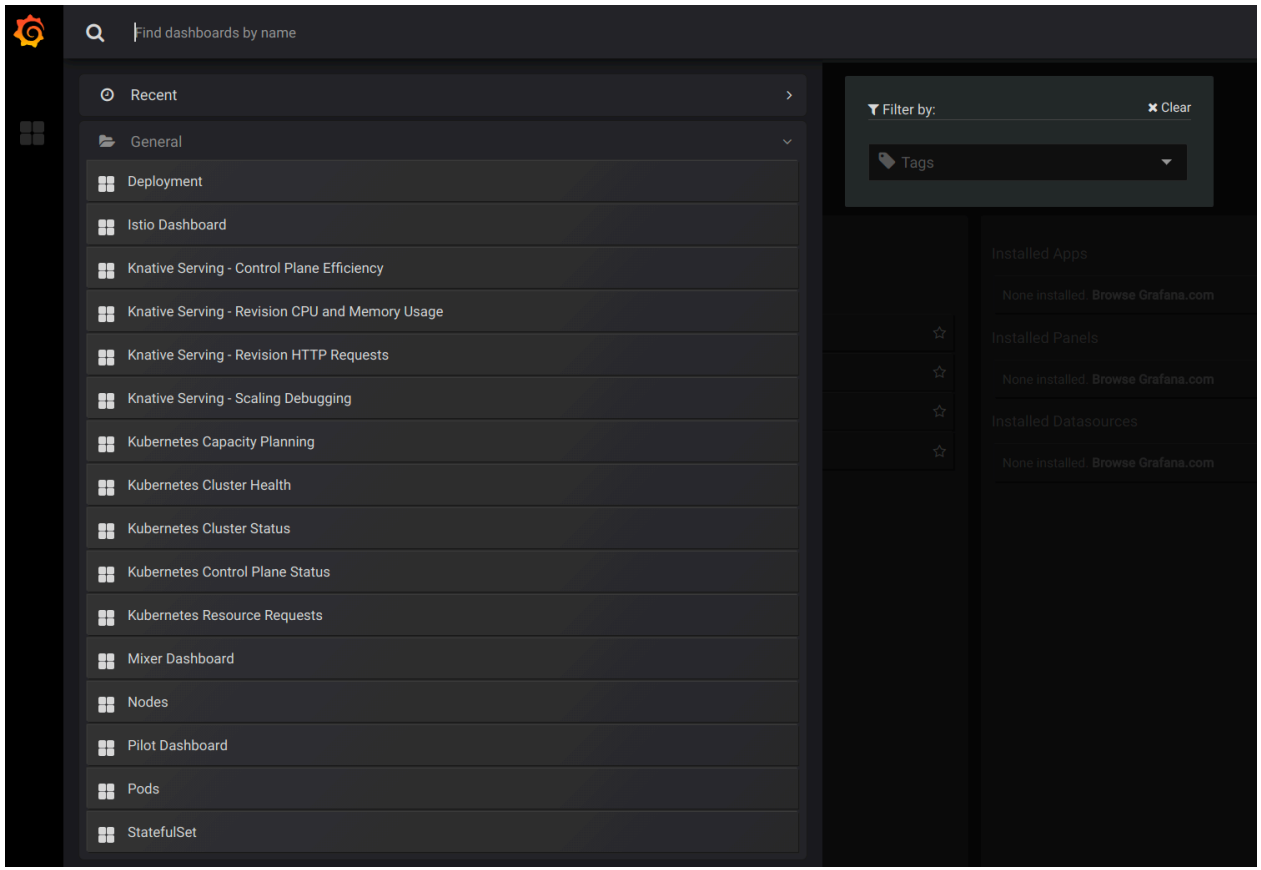
Observability

- Logging: Elasticsearch & Kibana, Stackdriver
- Monitoring: Prometheus & Grafana
- Tracing: Zipkin, Jaeger

Logging

- Request log
  - tag: "requestlog.logentry.istio-system”
- Configuration log
  - kubernetes.labels.serving_knative_dev\/configuration: <CONFIGURATION_NAME>
- Revision log
  - kubernetes.labels.serving_knative_dev\/revision: <REVISION_NAME>

Monitoring



- **Revision HTTP Requests:** HTTP request count, latency, and size metrics per revision and per configuration
- **Nodes:** CPU, memory, network, and disk metrics at node level
- **Pods:** CPU, memory, and network metrics at pod level
- **Deployment:** CPU, memory, and network metrics aggregated at deployment level
- **Istio, Mixer and Pilot:** Detailed Istio mesh, Mixer, and Pilot metrics
- **Kubernetes:** Dashboards giving insights into cluster health, deployments, and capacity usage

Thank You

谢　谢