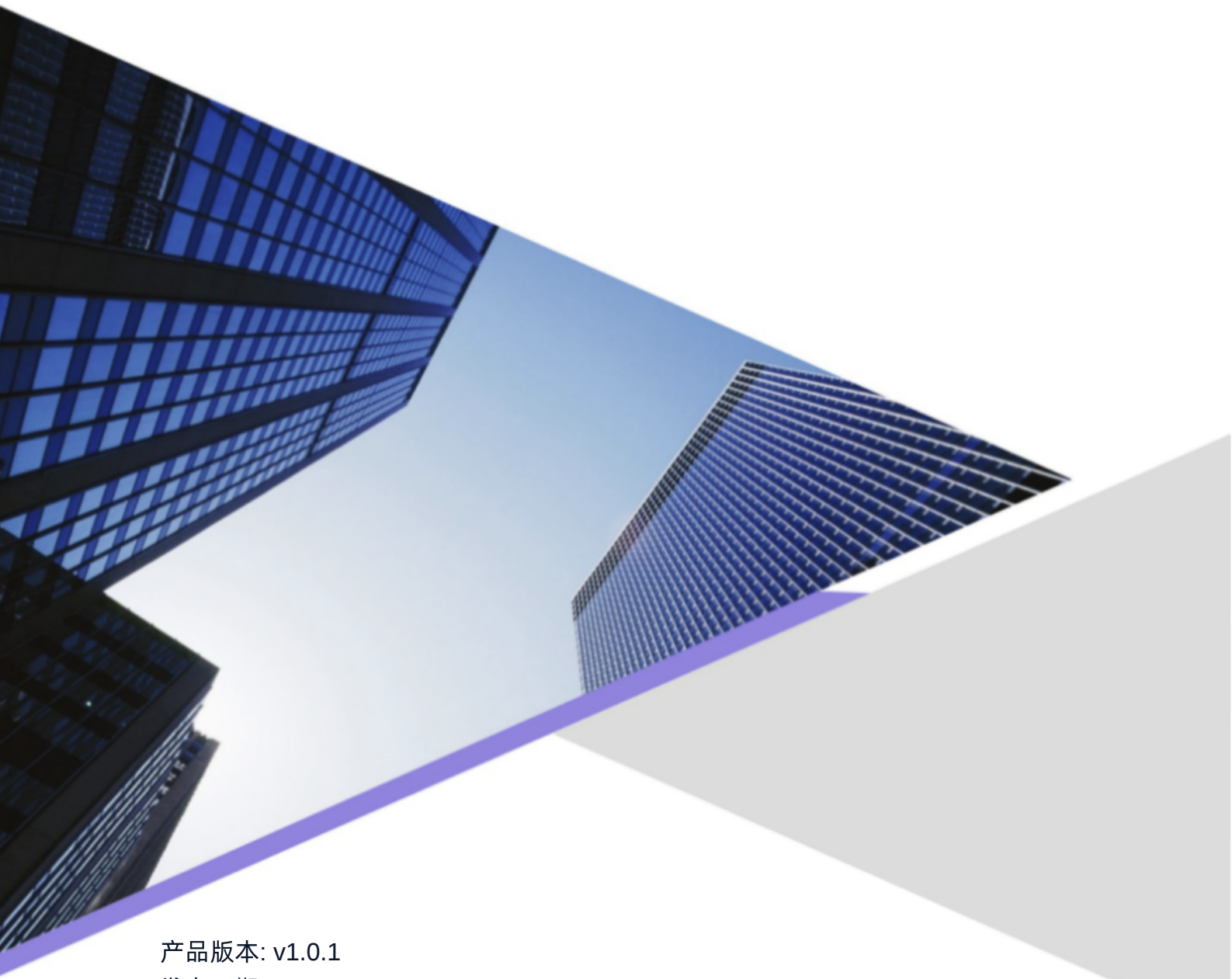
EasyStack
易捷行云

服务网格

常见问题



产品版本: v1.0.1

发布日期: 2024-06-05

目录

1 常见问题

1.1 为什么没注入

描述

为什么有些 pod 没有注入 sidecar?

检查方法

1. 首先检查pod所在命名空间，是否有注解： `istio-injection: enabled` ，如没有注解，则不会主动注入pod
2. 检查pod是否有注解： `sidecar.istio.io/inject: "true"` ，如有该注解，即使命名空间不设置注解也会主动注入
3. 检查pod类型，当前job，cronjob类型控制器产生的pod不会注入，即使配置1/2 ，也不会注入 sidecar
4. 检查pod网络类型，主机网络类型的不会注入，即使配置1/2中的注解也不会注入sidecar
5. 检查pod的注解，当有注解： `sidecar.istio.io/inject: "false"` ， 即使配置1 ，也不会主动注入sidecar

1.2 Skywalking工作负载Crash

描述

使用Skywalking实现链路追踪需要自建ElasticSearch，在Skywaking工作负载中填入正确的ElasticSearch信息。如果ElasticSearch连接异常会导致Skywalking工作负载Crash。

解决方案

步骤一

修改工作负载的yaml文件 `kubectl edit deploy skywalking-oap -n servicemesh` 修改 `skywalking-oap` 容器的环境变量:

```
        containers:
        - name: skywalking-oap
          image: {{ tuple .Values.images.tags "skywalking_oap_server" . |
include "helm-toolkit.utils.update_image" }}
          env:
            - name: SW_STORAGE_ES_CLUSTER_NODES
              value: "elasticsearch.servicemesh:9200"
            - name: SW_STORAGE_ES_HTTP_PROTOCOL
              value: "HTTP"
            - name: SW_ES_USER
              value: ""
            - name: SW_ES_PASSWORD
              value: ""
```

变量含义:

- `SW_STORAGE_ES_CLUSTER_NODES` :ElasticSearch服务地址，默认为 `elasticsearch.servicemesh:9200`
- `SW_STORAGE_ES_HTTP_PROTOCOL` :ElasticSearch连接协议，默认为HTTP
- `SW_ES_USER` :ElasticSearch用户，默认为空
- `SW_ES_PASSWORD` :ElasticSearch密码，默认为空 确保ElasticSearch相关信息填写正确且网络可达。

步骤二

保存退出后工作负载滚动升级，等待并观察Skywalking工作负载是否处于Running状态且Ready字段为1/1

kubectl get po -n servicemesh|grep skywalking-oap

```
skywalking-oap-xxxxxxxxxx-xxxxx                1/1     Running   0
1h
```

1.3 访问跨命名空间的服务没有度量和链路信息

描述

在某个命名空间a的客户端应用，访问另一个命名空间b的服务端应用后，通过kiali等页面查看时，发现没有度量信息和链路信息。

解决办法

为了减少服务网格下发到envoy sidecar的服务配置数量，在istio层面默认做了命名空间隔离，只能看到当前命名空间和指定的服务kubernetes和tracing-oap，可通过以下命令查看：

```
kubectl -nservicemesh get sidecar default -oyaml
apiVersion: networking.istio.io/v1beta1
kind: Sidecar
metadata:
  name: default
  namespace: servicemesh
spec:
  egress:
  - hosts:
    - ./*
    - default/kubernetes.default.svc.cluster.local
    - servicemesh/tracing-oap.servicemesh.svc.cluster.local
```

此时如果不在a命名空间设置sidecar资源，a空间下的客户端服务是没有经过sidecar的，因此没有度量和链路信息。需要在a命名空间下设置sidecar资源，让a空间下的sidecar能够访问b空间的服务端应用，示例设置如下，a空间默认可以访问b空间的服务b：

```
apiVersion: networking.istio.io/v1beta1
kind: Sidecar
metadata:
  name: default
  namespace: a
spec:
  egress:
```

```
  - hosts:
    - "b/svcb"
```

1.4 添加第三方注册中心后没生成服务对应的ServiceEntry资源

描述

部署RegistryHub资源后，返回 `registryhub.ecns.easystack.cn/xxx created` ，但是在导出的命名空间一直没有服务端对应的ServiceEntry资源。

解决方案

1. 通过查看RegistryHub资源描述，如果出现错误，在 `status.phase` 状态会置为 `failed` ，同时在 `status.conditions` 中会标注出错误原因，比如下面的错误是zookeeper地址设置错误，通过修改为正确地址后，再重新部署一下资源即可：

```
kubectl get registryhub registryhub-sample -oyaml
apiVersion: ecns.easystack.cn/v1alpha1
kind: RegistryHub
metadata:
  name: registryhub-sample
  namespace: default
spec:
  address: zookeeper1.default:2181
  generate_to: default
  type: zookeeper
status:
  address: zookeeper1.default:2181
  conditions:
  - failed_reason: 'lookup zookeeper1.default on 10.43.0.10:53: no such
host'
    record_time: "2024-02-04T02:53:25Z"
  generate_to: default
  phase: failed
  serviceentries: {}
  type: zookeeper
```

2. 查看RegistryHub资源描述，`status.phase` 显示为 `success` ，但仍然没有生成服务端对应的ServiceEntry资源，`status.serviceentryies` 只有几条 `ServiceEntry` 记录：

```
status:
  address: zookeeper.default:2181
  generate_to: default
  phase: success
  serviceentries:
    aeraki-org-apache-dubbo-samples-api-greetingservice-1-0-0-default:
"2024-02-04T03:04:20Z"
  type: zookeeper
```

可从以下两个方面排查：

- 是不是刚部署registryhub资源：如果是，可以继续等待几分钟，同时关注 `status.serviceentryies` 列表是否在持续增加，如果在增加，说明还在同步，稍后客户端再重试即可。
- 注册中心是否有服务端信息：如果registryhub的 `status.serviceentryies` 没有增加了，也没有看到服务端对于的ServiceEntry资源，可以排查注册中心是否存在服务端信息。

1.5 Envoy默认会将Header转换为小写

问题描述

Envoy缺省会把http header的key转换为小写，例如有一个http `header Test-Upper-Case-Header: some-value` ，经过envoy代理后会变成 `test-upper-case-header: some-value` 。这个在正常情况下没问题，RFC 2616 规范也说明了处理HTTP Header是大小写不敏感的

通常 header转换为小写不存在规范问题，但有些情况下对header大小写敏感会存在以下问题，例如：

- 业务解析header依赖大小写。
- 使用的SDK对Header大小写敏感，如读取 `Content-Length` 来判断response长度时依赖首字母大写。

解决方案

这种情况强制指定为TCP协议。将服务声明为TCP协议，不让istio进行七层处理，也就不会更改http header大小写了，但需要注意的是同时也会丧失istio的七层能力

当是集群内服务时，可以配置service使用TCP协议

```
kind: Service
metadata:
  name: myservice
spec:
  ports:
  - number: 80
    name: tcp-web # 指定该端口协议为 tcp
```

当是集群外服务时，可以配置serviceEntry数据结构

```
apiVersion: networking.istio.io/v1alpha3
kind: ServiceEntry
metadata:
  name: cos
spec:
  hosts:
  - "private.cos.guangzhou.mycloud.com"
```

```
  location: MESH_INTERNAL
  addresses:
  - 169.254.0.47
  ports:
  - number: 80
    name: tcp
    protocol: TCP
  resolution: DNS
```

1.6 第三方注册中心Zookeeper、Nacos生成ServiceEntry资源后，仍然不能访问

描述

使用 `RegistryHub` 资源，注册了基于 `dubbo` 协议的第三方注册中心Zookeeper、Nacos，也生成 `ServiceEntry` 资源，但是客户端仍然无法访问，比如生成的 `ServiceEntry` 如下：

```
kubectl -ndefault get serviceentry
NAME                                                                    HOSTS
LOCATION        RESOLUTION   AGE
aeraki-org-apache-dubbo-samples-api-greetingservice-1-0-0-default
["org.apache.dubbo.samples.api.greetingservice-1.0.0.default"]
MESH_INTERNAL   STATIC       32s
```

解决办法

使用基于 `dubbo` 协议的第三方注册中心Zookeeper、Nacos，因istio官方不知道 `dubbo` 协议，我们服务网格产品还会在 `servicemesh` 命名空间生成 `envoyfilter` ，来将下方的协议替换为 `envoy` 支持的 `dubbo` 协议，可通过以下命令查看是否生成对于的 `envoyfilter` ：

```
kubectl -nservicemesh get envoyfilter
```

对于上面的 `serviceentry` `aeraki-org-apache-dubbo-samples-api-greetingservice-1-0-0-default` 会生成以下两条记录：

```
aeraki-outbound-org.apache.dubbo.samples.api.greetingservice-1.0.0.default-
240.240.0.1-20880   3m53s
aeraki-inbound-org.apache.dubbo.samples.api.greetingservice-1.0.0.default-
20880               3m53s
```

如果查看没有生成对于的 `envoyfilter` ，可重启服务网格中的 `dubbo-controller` 服务，稍后查看是否解决。

```
kubectl -nservicemesh rollout restart deployment/dubbo-controller
```

1.7 Headless service相关问题

问题描述

服务间通过注册中心调用响应404，在Kubernetes的服务发现中，会使用service域名方式注册，如服务间调用不经过域名解析，直接向获取到的目的IP发起调用会导致404问题

解决方案

注册中心不直接注册Pod IP地址，注册service域名。 客户端请求时带上hosts（需要更新代码）。

1.8 Virtual Service 路由匹配顺序问题

问题描述

在写 VirtualService 路由规则时，通常会 match 各种不同路径转发到不同的后端服务，有时候不小心命名冲突了，导致始终只匹配到前面的服务，如下例子:

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: test
spec:
  gateways:
  - default/example-gw
  hosts:
  - 'test.example.com'
  http:
  - match:
    - uri:
        prefix: /usrv
    rewrite:
      uri: /
    route:
    - destination:
        host: usrv.default.svc.cluster.local
        port:
          number: 80
  - match:
    - uri:
        prefix: /usrv-expand
    rewrite:
      uri: /
    route:
    - destination:
        host: usrv-expand.default.svc.cluster.local
        port:
          number: 80
```

istio匹配是按顺序匹配，不像nginx那样使用最长前缀匹配。这里使用prefix进行匹配，第一个是 `/usrv` ，表示只要访问路径前缀含 `/usrv` 就会转发到第一个服务，由于第二个匹配路径 `/usrv-expand` 本身也属于带 `/usrv` 的前缀，所以永远不会转发到第二个匹配路径的服务

解决方案

这种情况可以调整下匹配顺序，如果前缀有包含的冲突关系，越长的放在越前面

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: test
spec:
  gateways:
  - default/example-gw
  hosts:
  - 'test.example.com'
  http:
  - match:
    - uri:
        prefix: /usrv-expand
    rewrite:
      uri: /
    route:
    - destination:
        host: usrv-expand.default.svc.cluster.local
        port:
          number: 80
  - match:
    - uri:
        prefix: /usrv
    rewrite:
      uri: /
    route:
    - destination:
        host: usrv.default.svc.cluster.local
        port:
          number: 80
```

1.9 重试策略导致服务异常

问题描述

Istio为Envoy设置了缺省的重试策略，会在connect-failure,refused-stream, unavailable, cancelled, retriable-status-codes等情况下缺省重试两次。出现错误时，可能已经触发了服务器逻辑，在操作不是幂等（任意多次执行所产生的影响均与一次执行的影响相同）的情况下，可能会导致错误

解决方案

可以通过配置 VS 关闭重试

```
apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: ratings
spec:
  hosts:
  - ratings
  http:
  - retries:
      attempts: 0
```

1.10 如何调整istio-proxy容器resources requests取值

描述

istio-proxy容器资源占用大小的默认配置如下。如果不符合要求，可按照实际需求进行修改。

```
resources:
  requests:
    cpu: 500m
    memory: 128Mi
  limits:
    cpu: 2000m
    memory: 1024Mi
```

解决方案

调整网格中的某个服务

步骤一：

修改服务的yaml文件。 `kubectl edit deploy <nginx> -n <namespace>`

步骤二：

在 `spec.template.metadata.annotations` 下添加如下配置（大小仅供参考，请自行替换）。

```
sidecar.istio.io/proxyCPU: 500m
sidecar.istio.io/proxyCPULimit: 500m
sidecar.istio.io/proxyMemoryLimit: 1024Mi
sidecar.istio.io/proxyMemory: 1024Mi
```

步骤三

修改后服务滚动升级，确保不会断服。

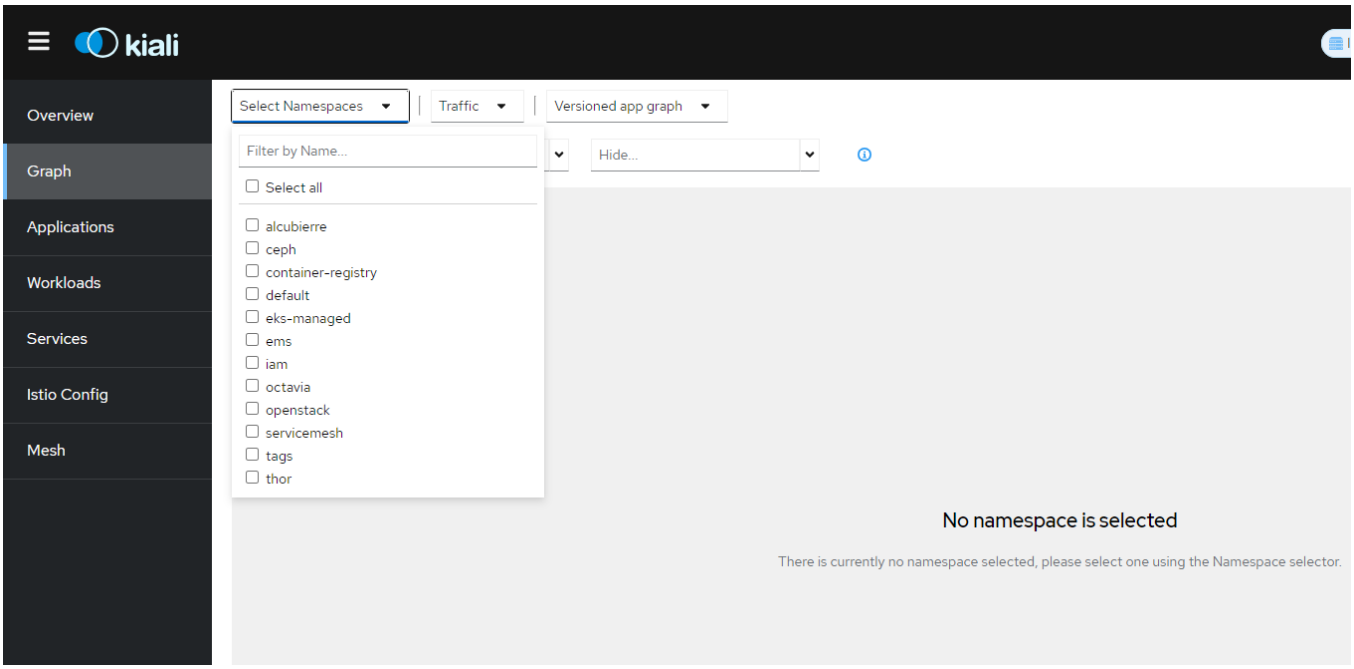
1.11 Kiali流量监控拓扑图中找不到服务

描述

Kiali页面流量拓扑图未展示用户部署的服务。

解决方案

步骤一：确认命名空间是否选择正确



确保选择服务部署对应的命名空间。

步骤二：确认服务是否有网络流量

确认服务有发起或者接收到请求，可结合pod日志进行查看：`kubectl logs <服务对应pod名称> -n <服务所在命名空间>`

步骤三：确认命名空间sidecar注入标签

```
kubectl get namespace <服务所在命名空间> -oyaml
```

```
apiVersion: v1
kind: Namespace
metadata:
  ...
  labels:
    kubernetes.io/metadata.name: <服务所在命名空间>
    istio-injection: enabled
  name: <服务所在命名空间>
...
```

如果命名空间不存在标签 `istio-injection=enabled` ，执行以下命令给命名空间打上标签。 `kubectl label namespace <服务所在命名空间> istio-injection=enabled` 或者如果只希望单独为服务添加注入标签开启sidecar注入，请参考步骤四中的场景2。

步骤四：确认sidecar注入相关的工作负载标签和注解

场景一：所在命名空间已有sidecar注入标签

确认pod模板没有禁止sidecar注入的标签或者注解（注解方式不推荐使用） `kubectl get deploy <服务对应的deployment名称> -n <服务所在命名空间>`

```
...
spec:
  ...
  template:
    metadata:
      ...
      labels:
        sidecar.istio.io/inject: false
        ...
      annotations:
        sidecar.istio.io/inject: false #注解方式不推荐使用，与标签二选一配置即可
    spec:
      ...
...
```

如标签或者注解为 `sidecar.istio.io/inject: false` ，编辑 `deployment` ，删除 `pod` 模板中的 `sidecar.istio.io/inject` 标签或者注解即可。 `kubectl edit deploy <服务对应的deployment名称> -n <`

服务所在命名空间>

场景二：所在命名空间没有sidecar注入标签，希望单独为服务添加注入标签

`kubectl edit deploy <服务对应的deployment名称> -n <服务所在命名空间>` 编辑 `deployment` 中的 `pod` 模板，为 `pod` 添加注入标签

```
...
spec:
  ...
  template:
    metadata:
      ...
      labels:
        sidecar.istio.io/inject: true #推荐用法
        ...
      annotations:
        sidecar.istio.io/inject: true #注解方式不推荐使用，与标签二选一配置即可
    spec:
      ...
...
```

保存退出，等待服务滚动升级完成。

步骤五：标签配置正确的情况，检查pod中是否有sidecar

如果前面步骤检查都没问题，但Kiali页面仍未展示相关的流量拓扑，需要检查sidecar实际是否注入。

`kubectl get pod <服务对应的pod> -o jsonpath={.spec.containers[*].name} -n <服务所在命名空间>`

```
<服务对应的容器> ... istio-proxy
```

输出的容器名称有 `istio-proxy` 即已经注入了 `sidecar` 。 如果 `istio-proxy` 容器不存在，可以选择对相应的 `deployment` 进行滚动升级， `sidecar` 会自动注入。 `kubectl edit deploy <服务对应的deployment名称> -n <服务所在命名空间>` 编辑服务对应的 `deployment` ，推荐通过添加自定义 `pod` 模板标签的方式进行滚动升级。

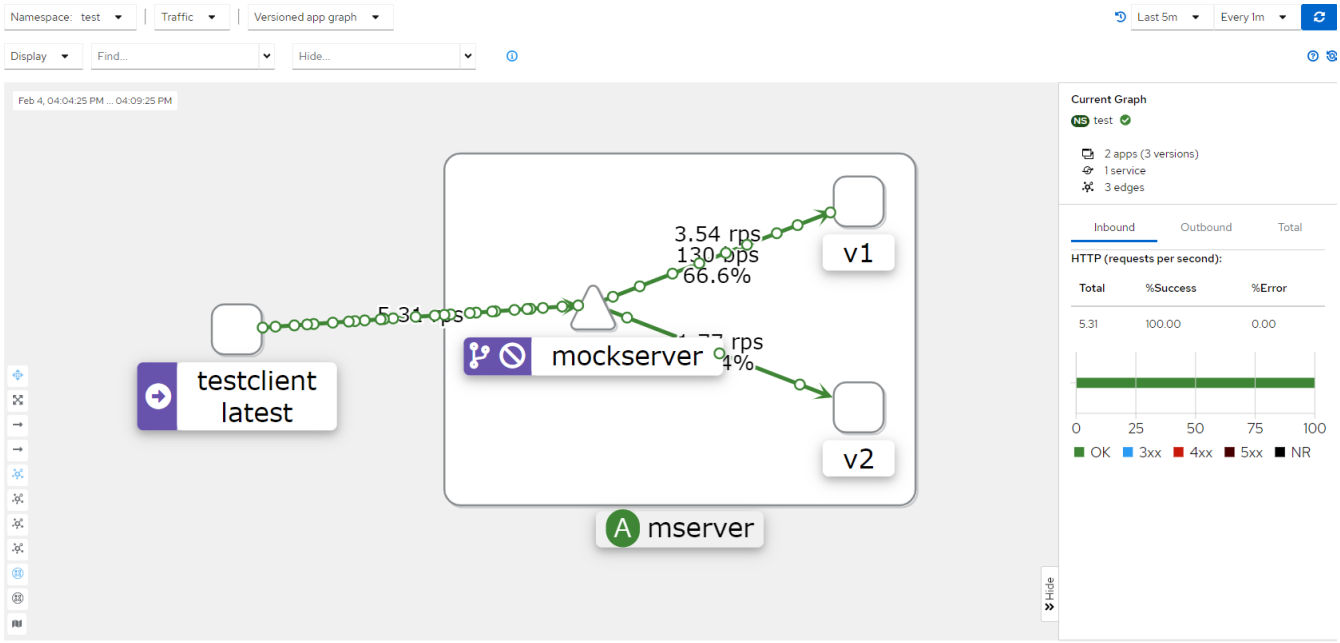
```
...
spec:
```

```
  ...
  template:
    metadata:
      ...
      labels:
        try-inject: '12345678'    #无特殊要求，可自定义
        ...
    spec:
      ...
...
```

等待滚动升级完成。

步骤六：完成所有检查后，打开Kiali页面确认流量拓扑正确展示

需要注意选择查询的时间长度以及刷新周期。也可点击刷新按钮手动刷新流量拓扑。



咨询热线：400-100-3070

北京易捷思达科技发展有限公司：
北京市海淀区西北旺东路10号院东区1号楼1层107-2号
南京易捷思达软件科技有限公司：
江苏省南京市雨花台区软件大道168号润和创智中心4栋109-110

邮箱：
contact@easystack.cn (业务咨询)
partners@easystack.cn(合作伙伴咨询)
marketing@easystack.cn (市场合作)