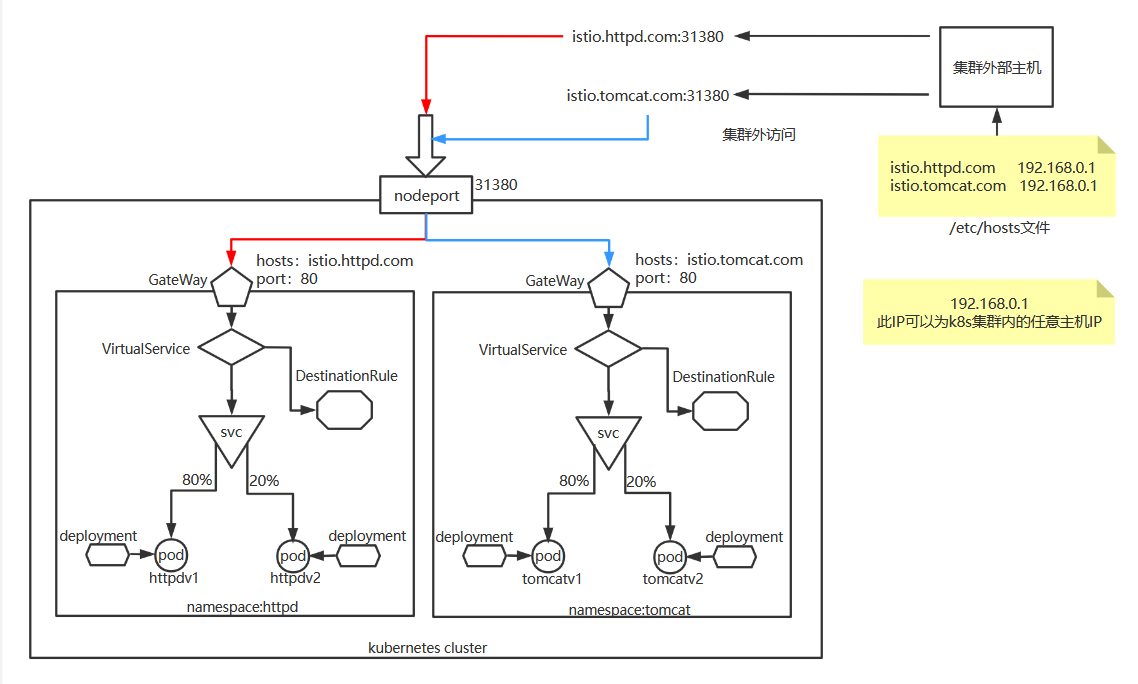
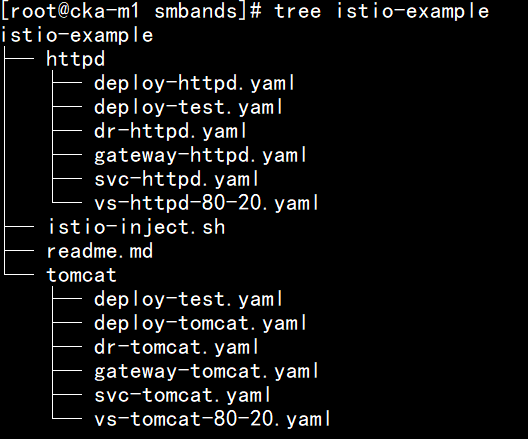
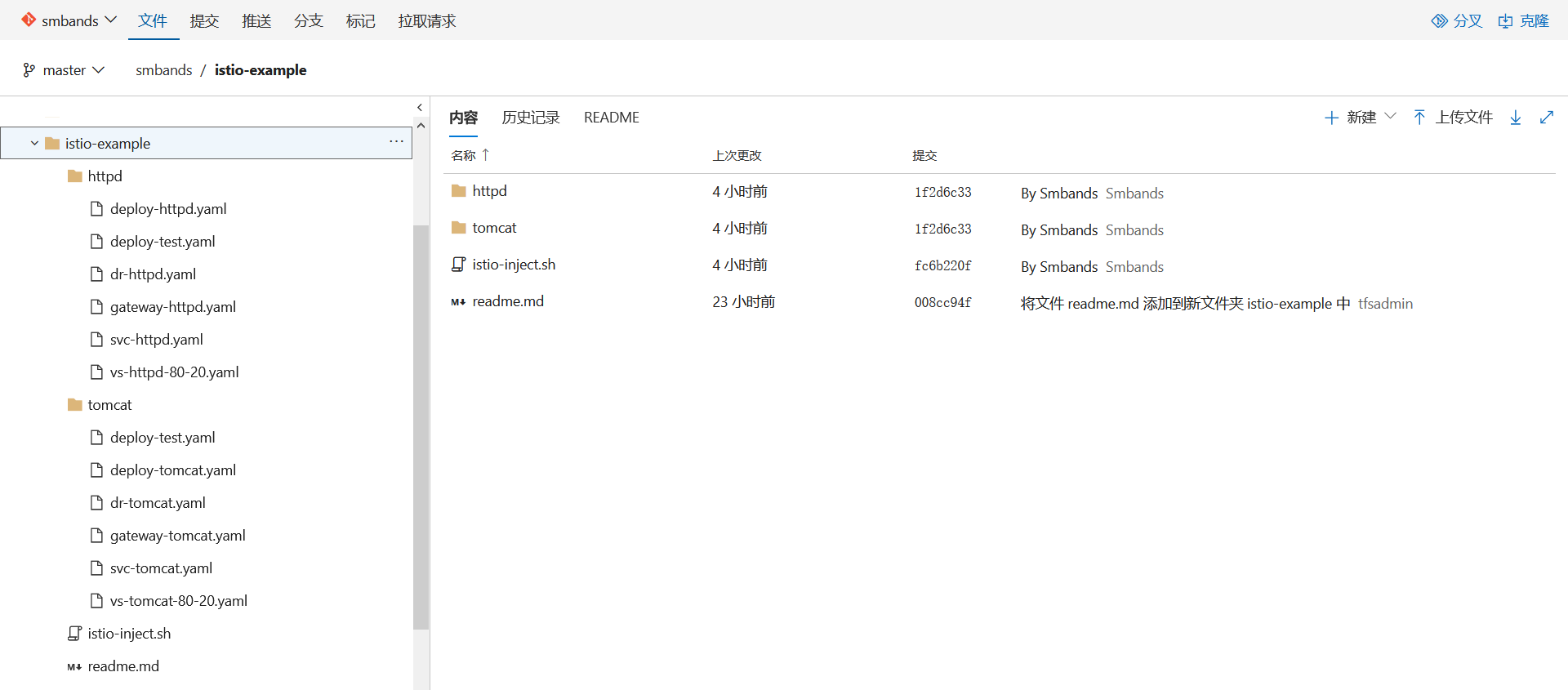
Istio灰度部署的简单实践

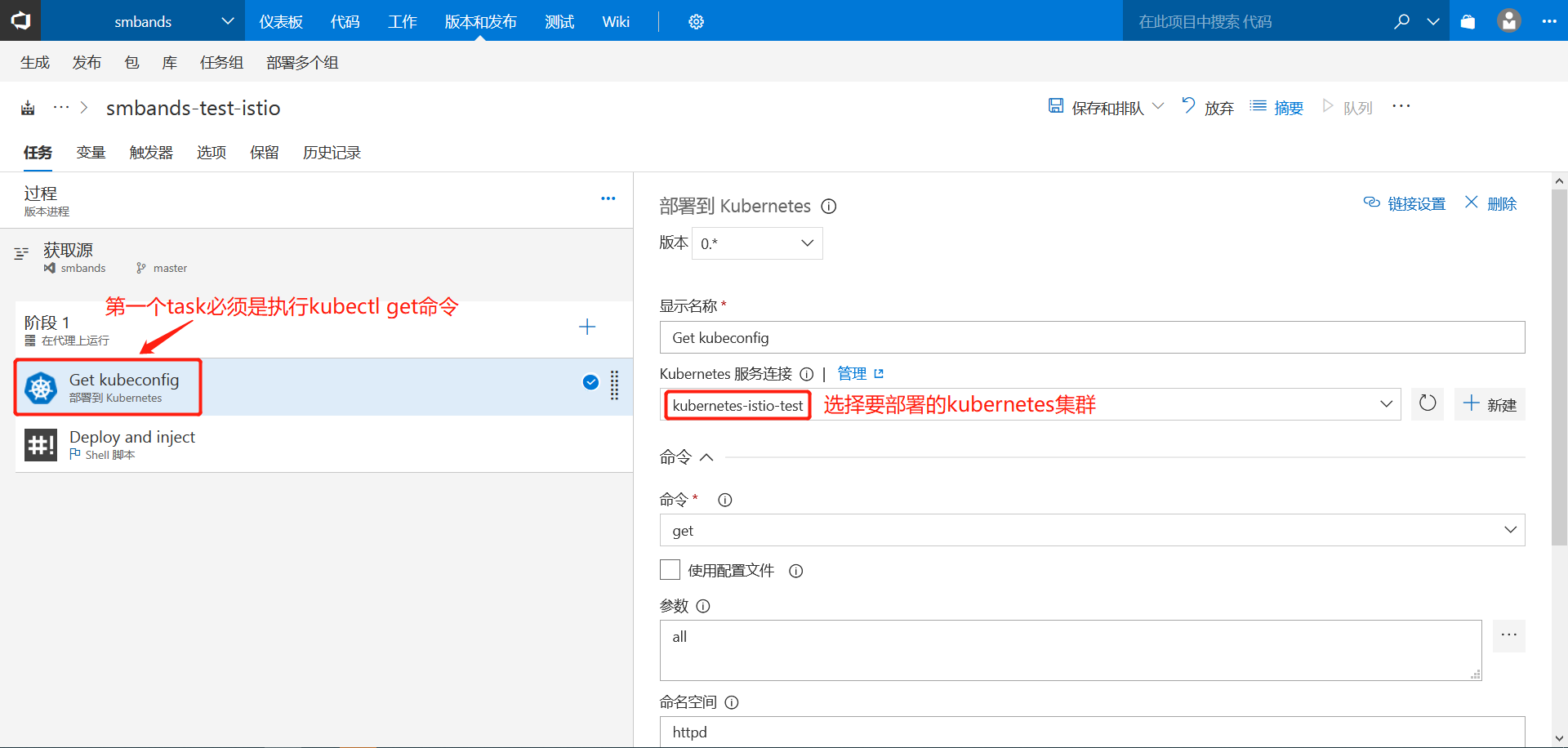
* 获取demo：
* 此示例将virtual service绑定在gateway上，以实现为不同服务单独创建gateway。
* Istio资源组件介绍：
  + Gateway：网格入口，主要定义了端口和hosts。Gateway中定义的端口会对应istio-ingressgateway的svc暴露的nodeport端口；hosts则为访问gateway的域名或IP。
  + VirtualService：虚拟服务。将k8s上面的svc连接到istio的网关，并且可以定义一些流量的路由规则等。
  + DestinationRule：目标规则。可以理解为书中的目录，哪个标题在哪一页（哪个版本用的哪个标签）此处版本在istio中称为subset。
* 架构图：
* 原理介绍：



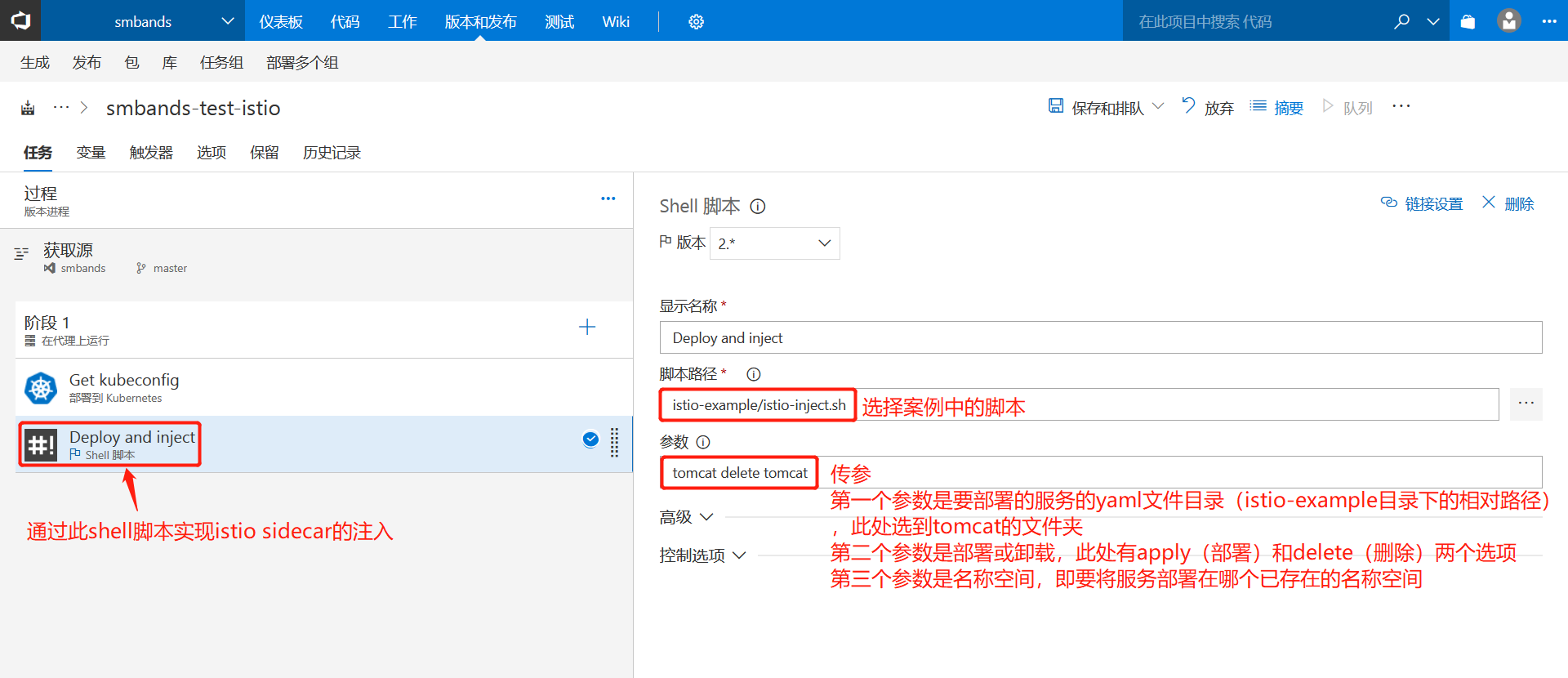
* + 集群外部主机通过域名和nodeport端口访问服务。
  + 访问流量被ingressgateway的pod服务监听，此pod通过gateway中定义的域名将流量转发到对应的gateway。
  + 被绑定在此gateway上的virtual service接收到流量后，先“查询”DestinationRule得知定义的“目录”（版本），然后通过自身定义的流量规则，将流量按规则转发到不同版本的pod。
* 安装实践：
  + 解压istio-example.tar.gz，文件名称根据资源类型和具体服务命名，功能与上面的架构图匹配。目录结构如下：
  + 将此目录上传至tfs的git中：



* + 创建生成：



说明：第一个task执行的是kubectl get命令，主要目的是日志中获取kubectl命令后的--kubeconfig的路径，从而下面的脚本可以获取此kubernetes（cke）集群的config文件，所以要指定好是哪个kubernetes(cke)集群，也可以执行kubectl的其他命令，但是为了不对集群造成影响，此处最好使用get。

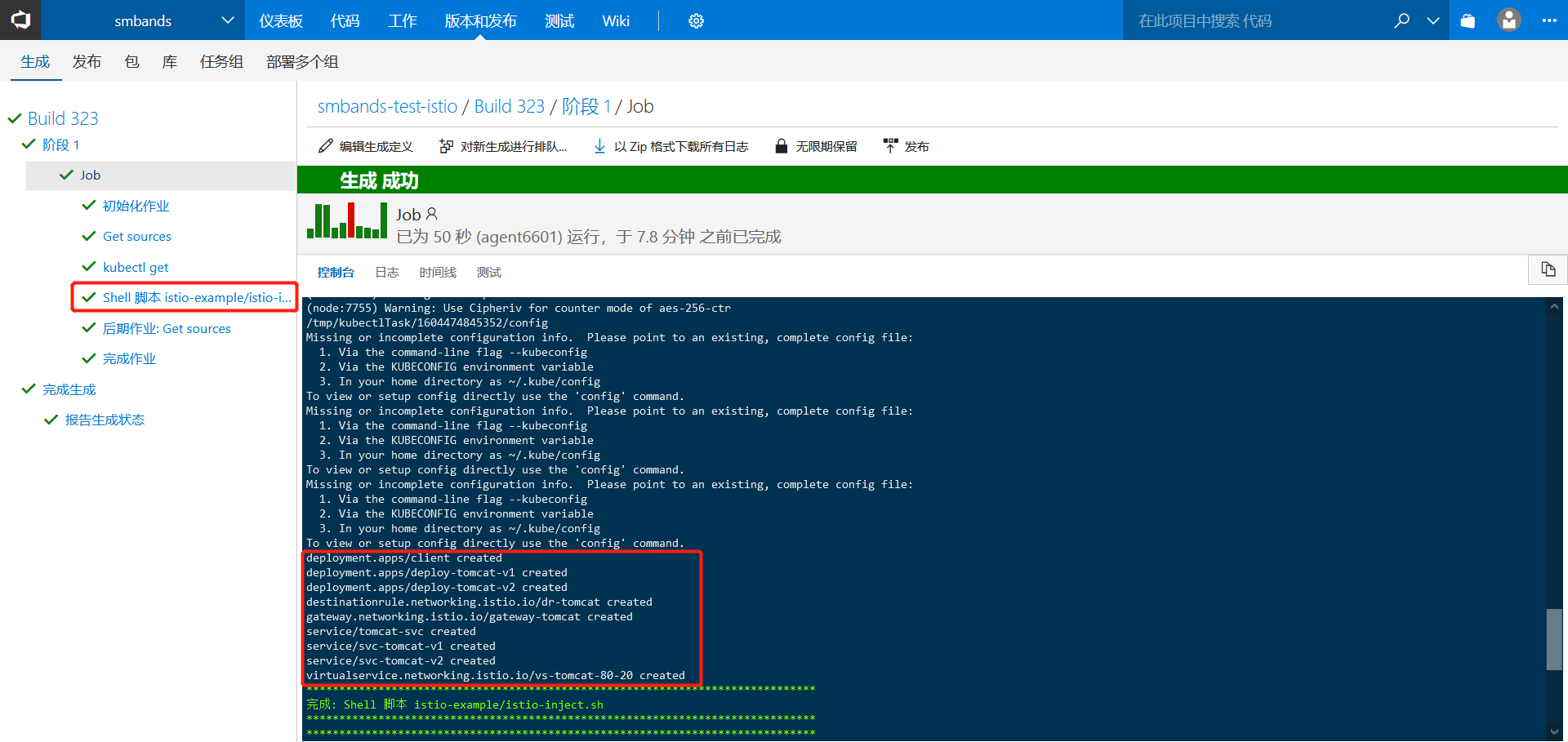
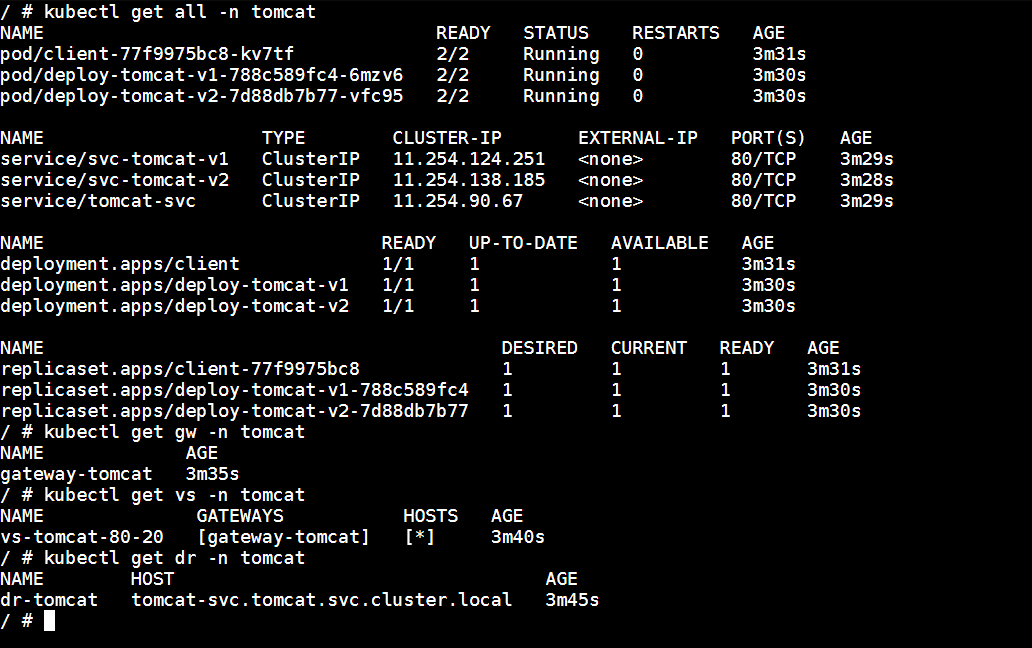
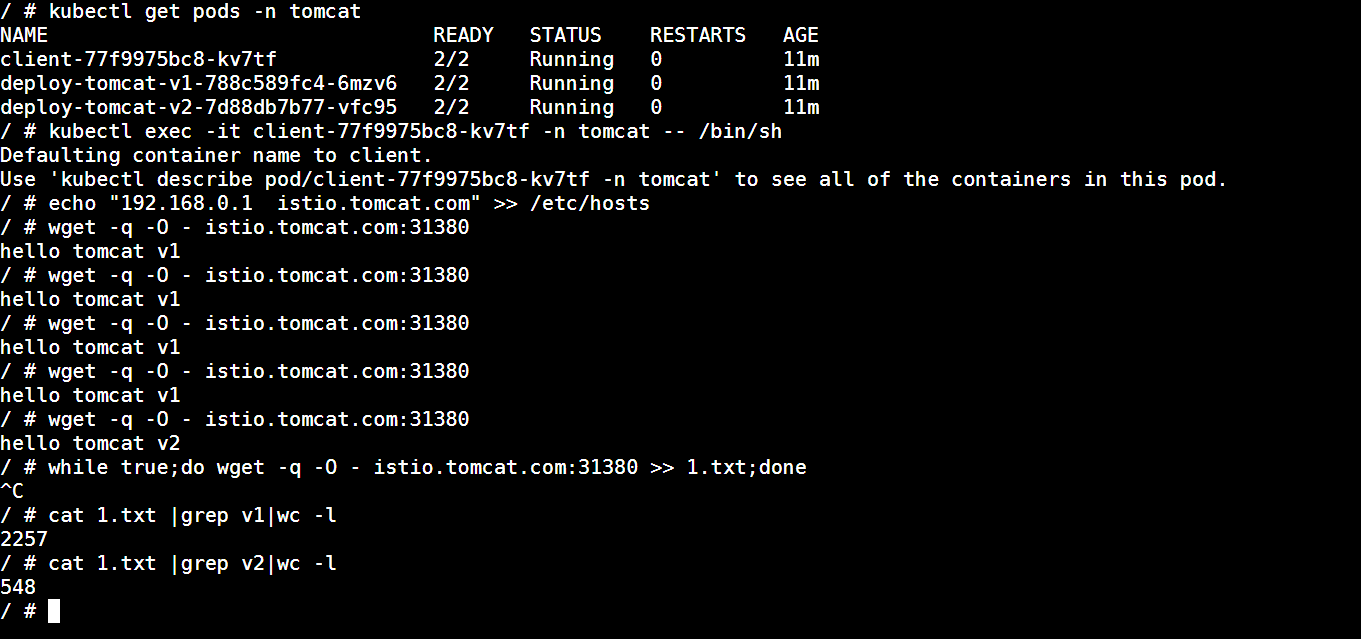


说明：此脚本主要实现的是获取上个kubectl命令带出来的config文件，然后通过传递参数完成注入。

参数说明：第一个参数：要部署的yaml文件所在目录，此目录是相对于istio-inject.sh脚本文件的相对目录，因为执行脚本时的当前目录为脚本的目录，脚本的目录下有httpd和tomcat的文件夹，所以就直接写httpd或tomcat文件夹即可。

第二个参数：部署或卸载的动作命令，即补充kubectl apply/delete，部署或更新时直接将istio的sidecar进行注入，此处无需考虑服务是否已经部署，因为使用apply时，不存在则创建，存在则更新。此参数只能填写apply或delete。

第三个参数：即kubectl 的-n参数，指定要将服务部署在哪个已存在的名称空间下。

* + Task和相关参数填写完成后即可保存和排队：可以看到资源已经被创建。
  +  在cke的命令行中查看部署的状态：
  + 由上图pod的READY状态可以看出，pod中已经是两个容器在运行，说明sidecar已经成功注入。
* 访问：由于cke中没有curl或wget请求命令，在此设计了一个client，可以通过命令行进入client的pod中进行验证：
* 
* 注：client的pod中需要配置一下hosts文件，然后通过域名+nodeport访问。从访问的结果看，v1出现的次数大概是v2的4倍，即80%的流量进入了v1的pod，20%的流量进入了v2的pod。