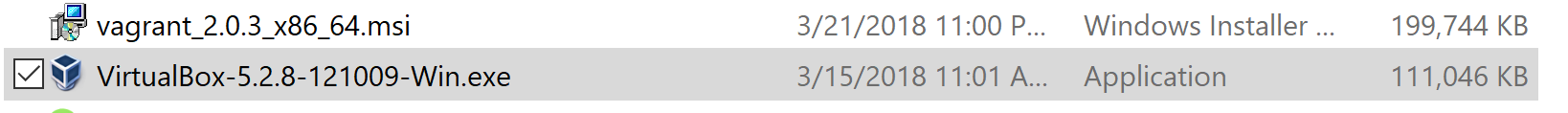
**根据此文档，我们将创建一个3节点的Kubernetes集群，包含1个master节点和2个node节点，默认信息如下：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Role** | **Hostname** | **IP** | **CPU** | **Memory** |
| Master | vg-k8s-master | 192.168.33.10 | 2 | 4G |
| Node | vg-k8s-node-1 | 192.168.33.11 | 2 | 4G |
| Node | vg-k8s-node-2 | 192.168.33.12 | 2 | 4G |

**1．下载并安装VirtualBox(5.2.8+)和Vagrant(2.0.3+)**



[*https://download.virtualbox.org/virtualbox/5.2.8/VirtualBox-5.2.8-121009-Win.exe*](https://download.virtualbox.org/virtualbox/5.2.8/VirtualBox-5.2.8-121009-Win.exe)

[*https://releases.hashicorp.com/vagrant/2.0.3/vagrant\_2.0.3\_x86\_64.msi*](https://releases.hashicorp.com/vagrant/2.0.3/vagrant_2.0.3_x86_64.msi)

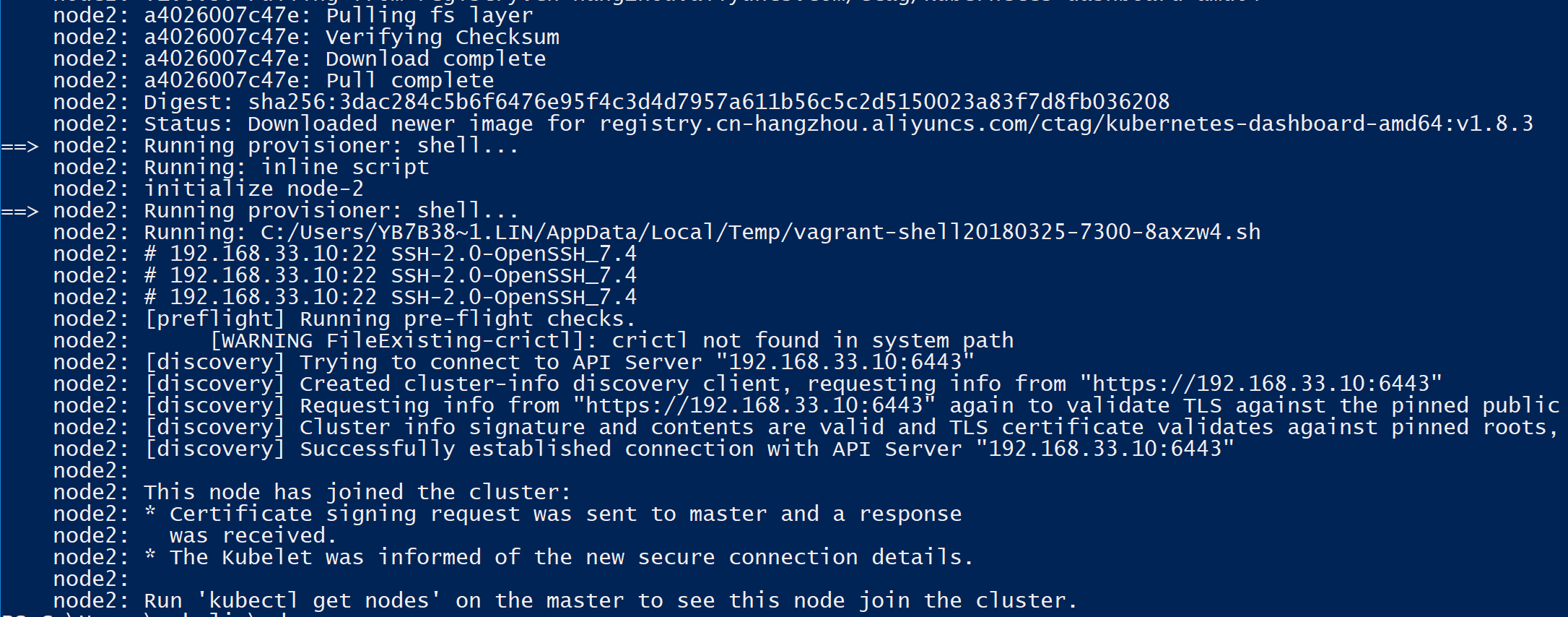
**2. 下载Vagrant脚本：**

*git clone* [*https://github.com/linyang0625/k8s-virtualbox*](https://github.com/linyang0625/k8s-virtualbox)

**3. 进入clone下来的目录并运行：**

*vagrant up*

**4. 等待大约15~20分钟（第一次运行时会下载centos镜像，时间会稍长一些），脚本运行结束。**



**5. 可以在命令行通过以下命令登录到每个节点：**

*vagrant ssh master*

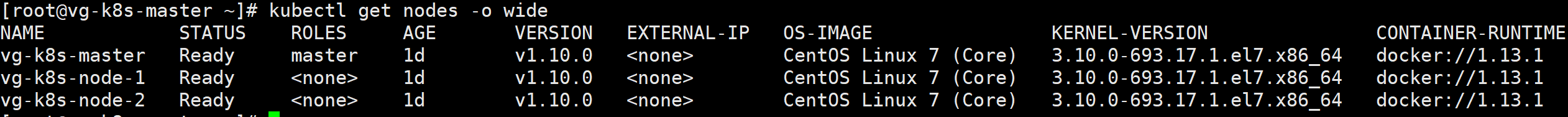
*vagrant ssh node1*

*vagrant ssh node2*

**6. 在master节点上查看所有节点的状态**

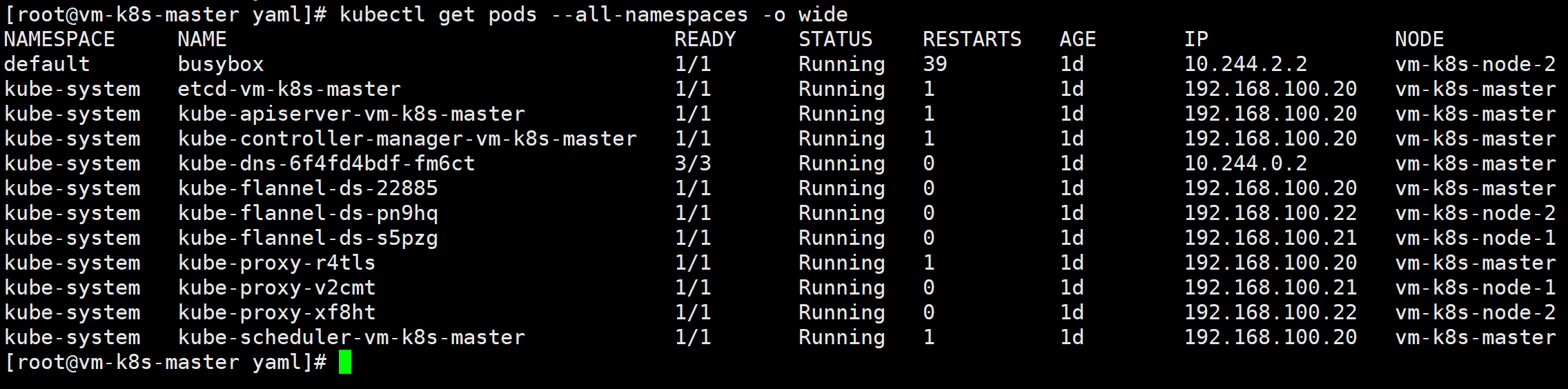
*sudo -i*

*kubectl get nodes -o wide*



**查看一下所有pod的运行情况，是否所有pod都是Running状态：**

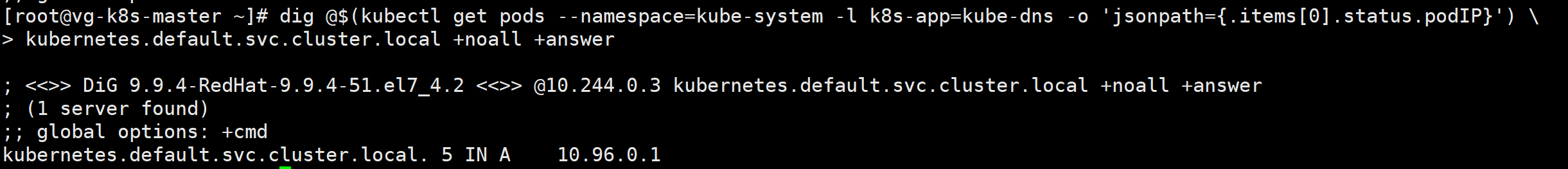
*kubectl get pods --all-namespaces -o wide*



**用下面命令测试一下DNS**

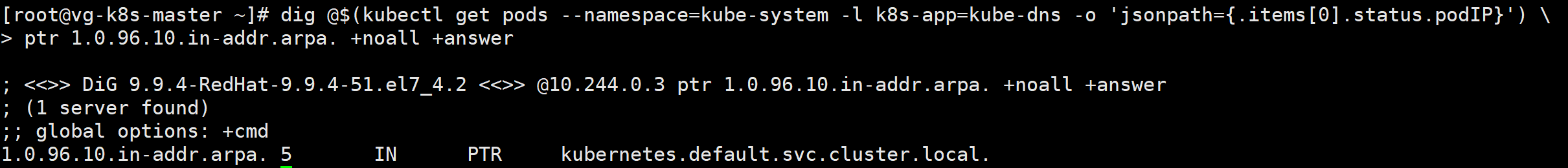
*dig @$(kubectl get pods --namespace=kube-system -l k8s-app=kube-dns -o 'jsonpath={.items[0].status.podIP}') \*

*kubernetes.default.svc.cluster.local +noall +answer*



*dig @$(kubectl get pods --namespace=kube-system -l k8s-app=kube-dns -o 'jsonpath={.items[0].status.podIP}') \*

*ptr 1.0.96.10.in-addr.arpa. +noall +answer*



**再使用busybox测试一下网络的情况：**

*kubectl create -f /vagrant/yaml/busybox.yaml*

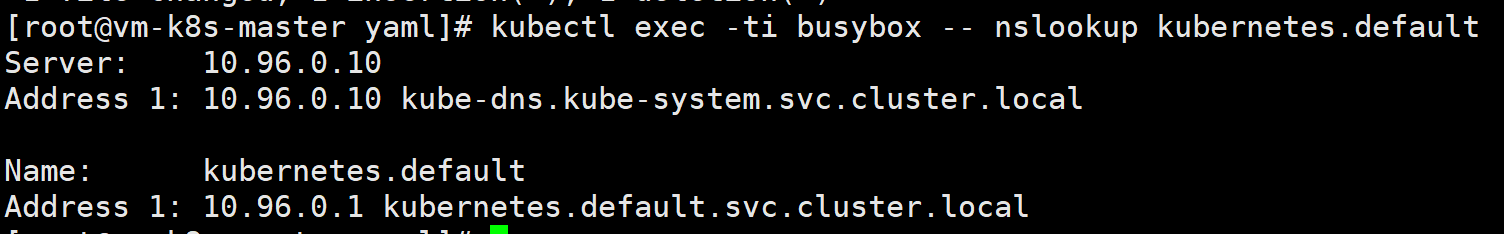
**通过**

*kubectl get pods -o wide*

**查看busybox的状态，待busybox状态变为Running后，运行：**

*kubectl exec -ti busybox -- nslookup kubernetes.default*

**如果看到下面输出，整个集群即可认为正常运行：**



**7. 本机如果是Windows系统，修改C:\Windows\System32\drivers\etc\hosts，加入以下几行：**

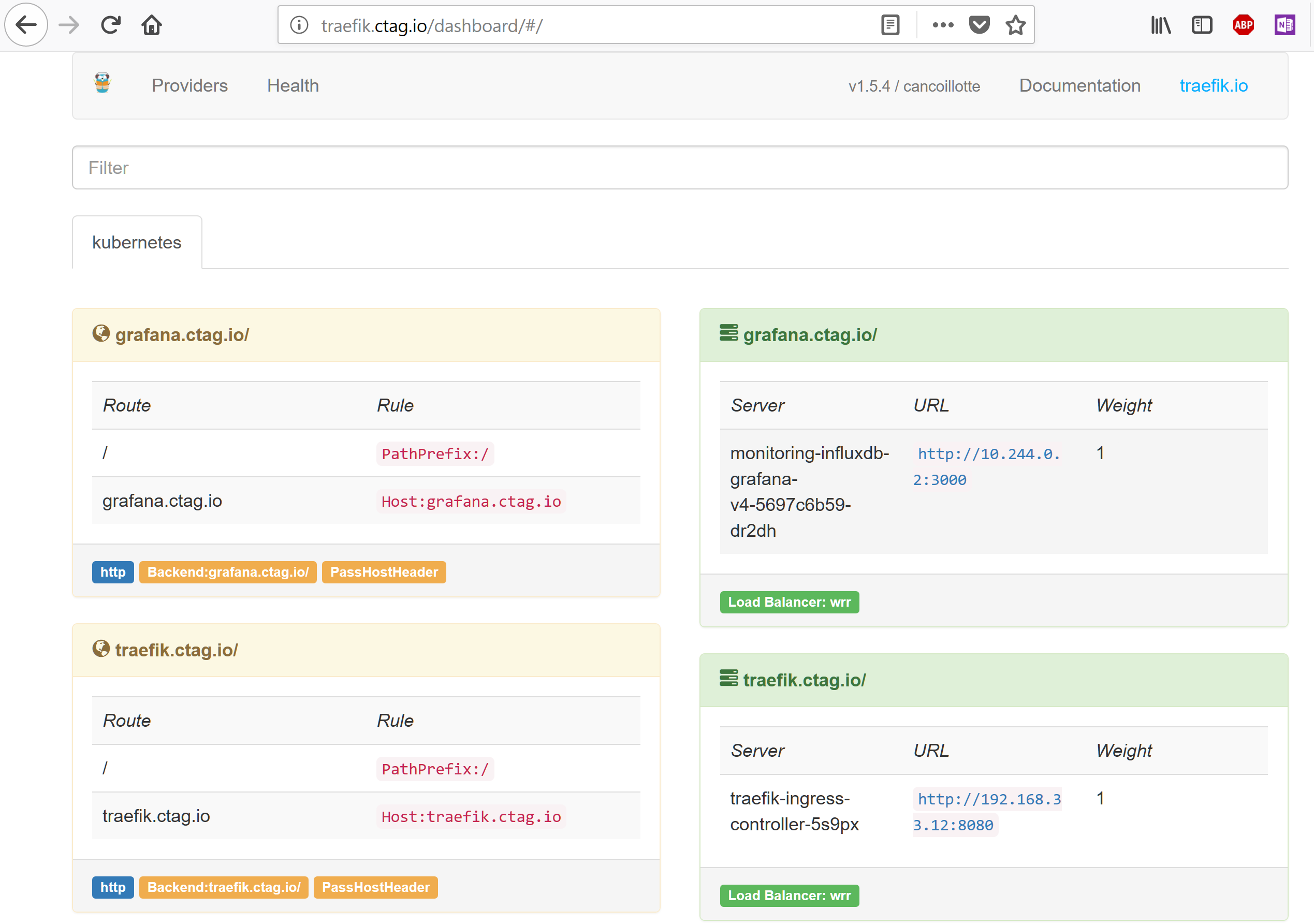
*192.168.33.12 grafana.ctag.io*

*192.168.33.12 traefik.ctag.io*

*192.168.33.12 jaeger.ctag.io*

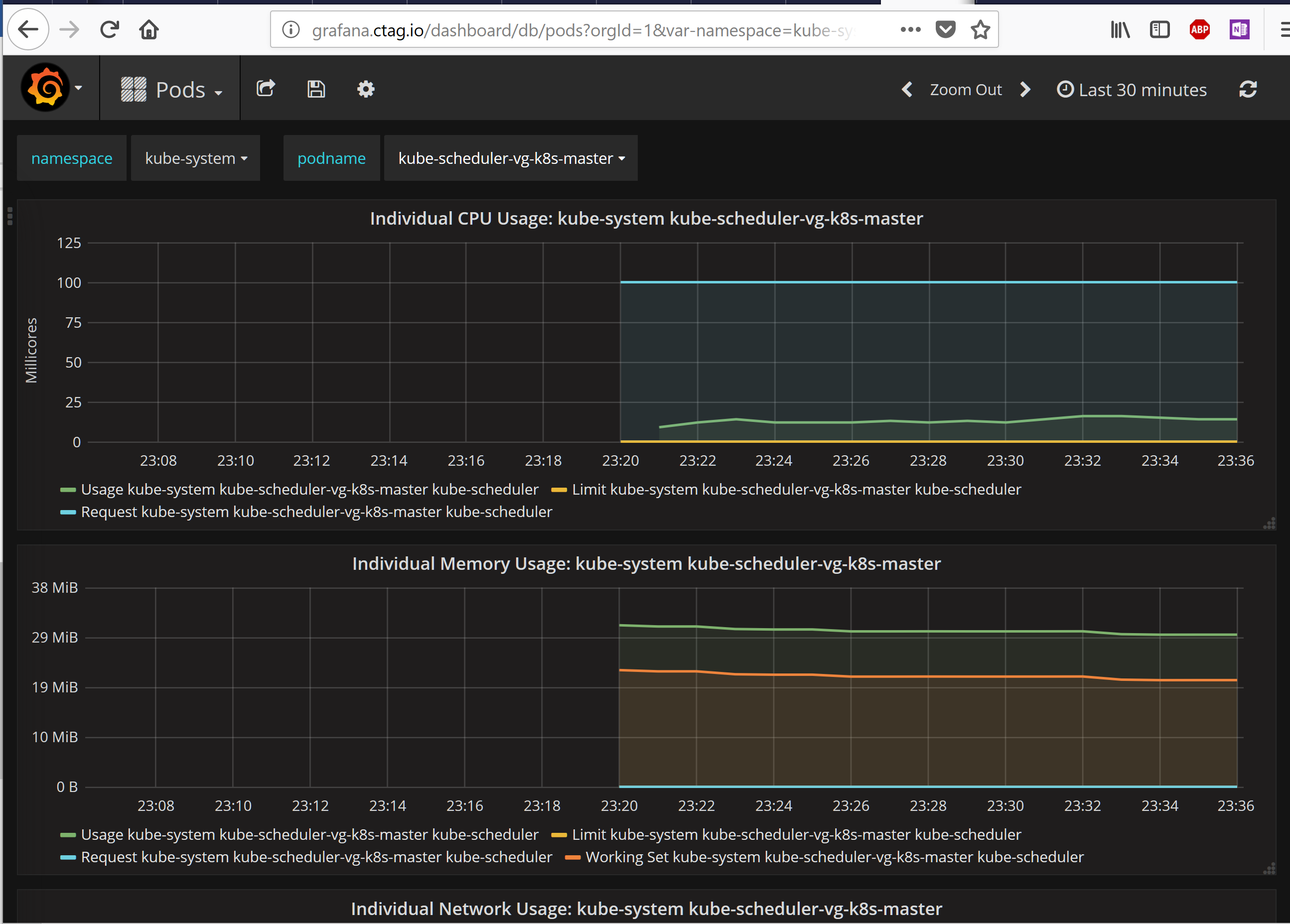
*192.168.33.12 weave.ctag.io*

**8. 访问**[*http://traefik.ctag.io*](http://traefik.ctag.io)**， 可以看到如下Ingress的控制台页面：**



**9. 访问grafana控制台：**

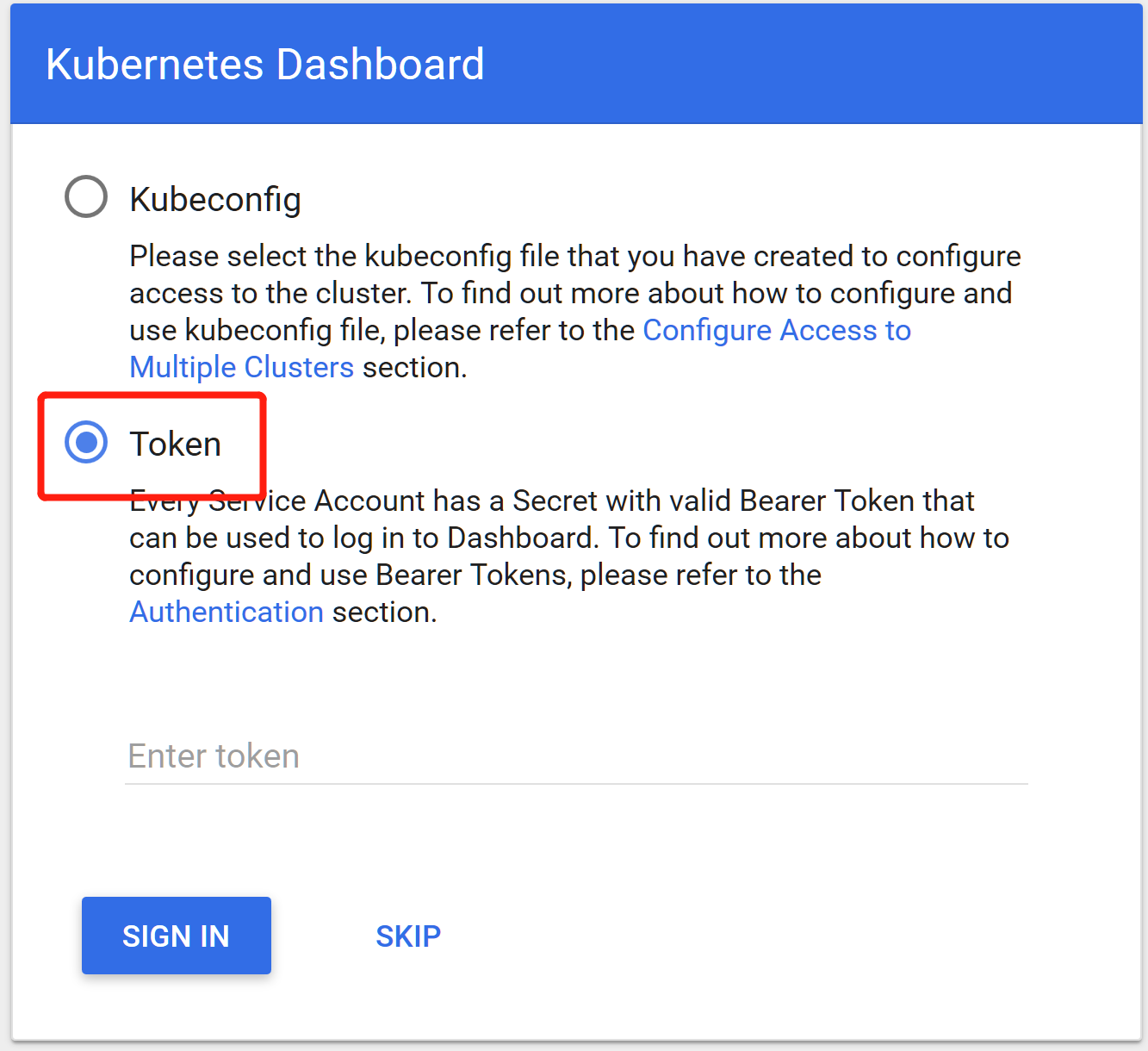
[*http://grafana.ctag.io*](http://grafana.ctag.io)



**10. 访问Kubernetes的dashboard：**

<https://192.168.33.11:30001>

**在登录时候选Token：**

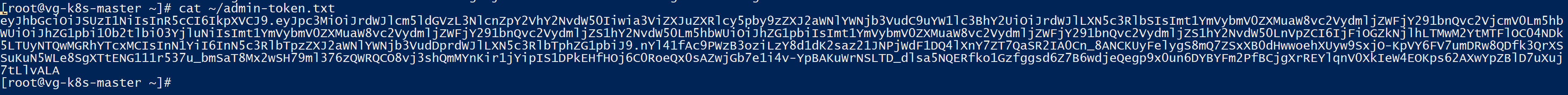


**在master节点上运行：**

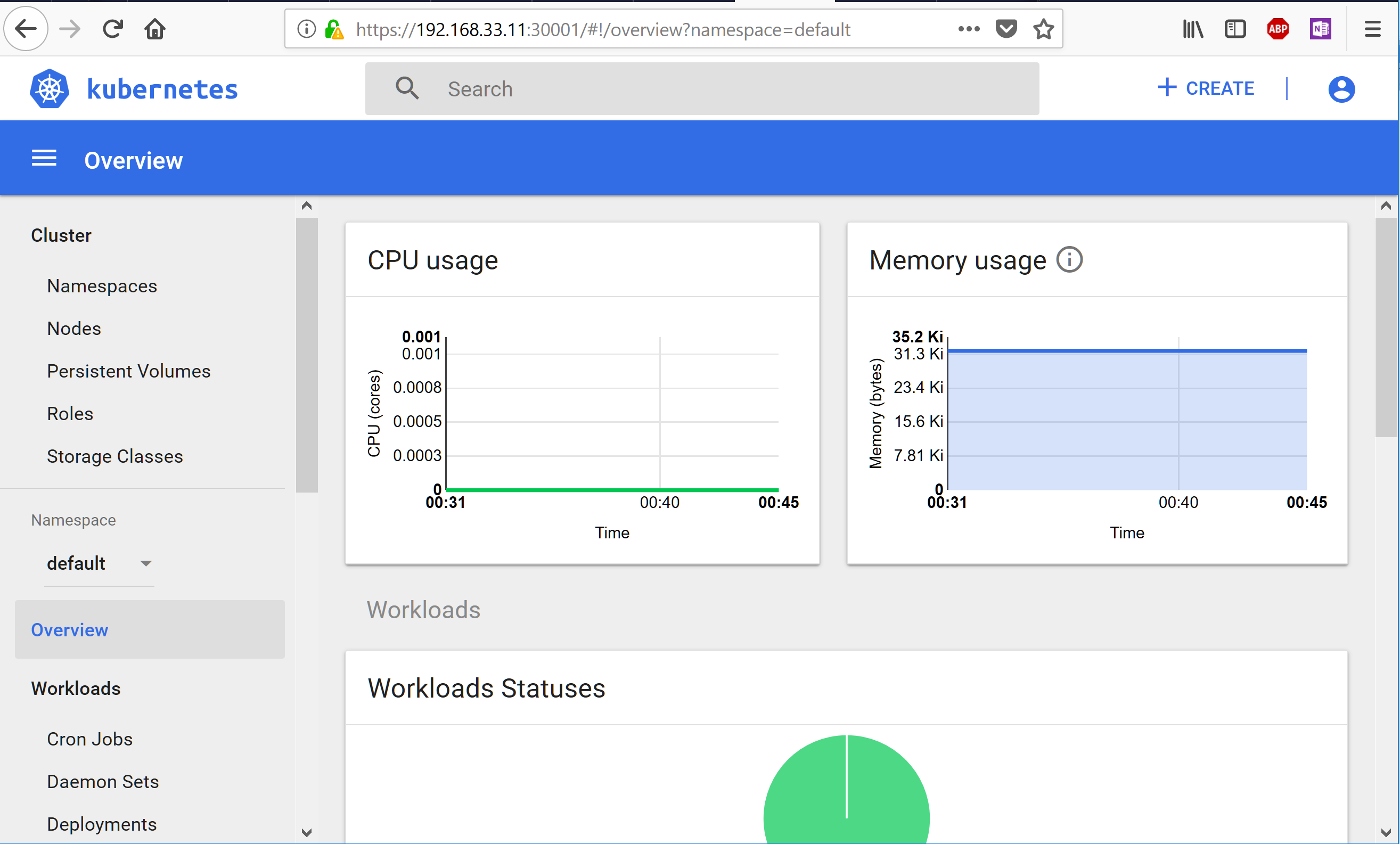
*sudo -i*

*cat ~/admin-token.txt*

**可以得到用来登录的token：**



**把此token贴入上面的登录窗口，登录之后即可看到dashboard：**

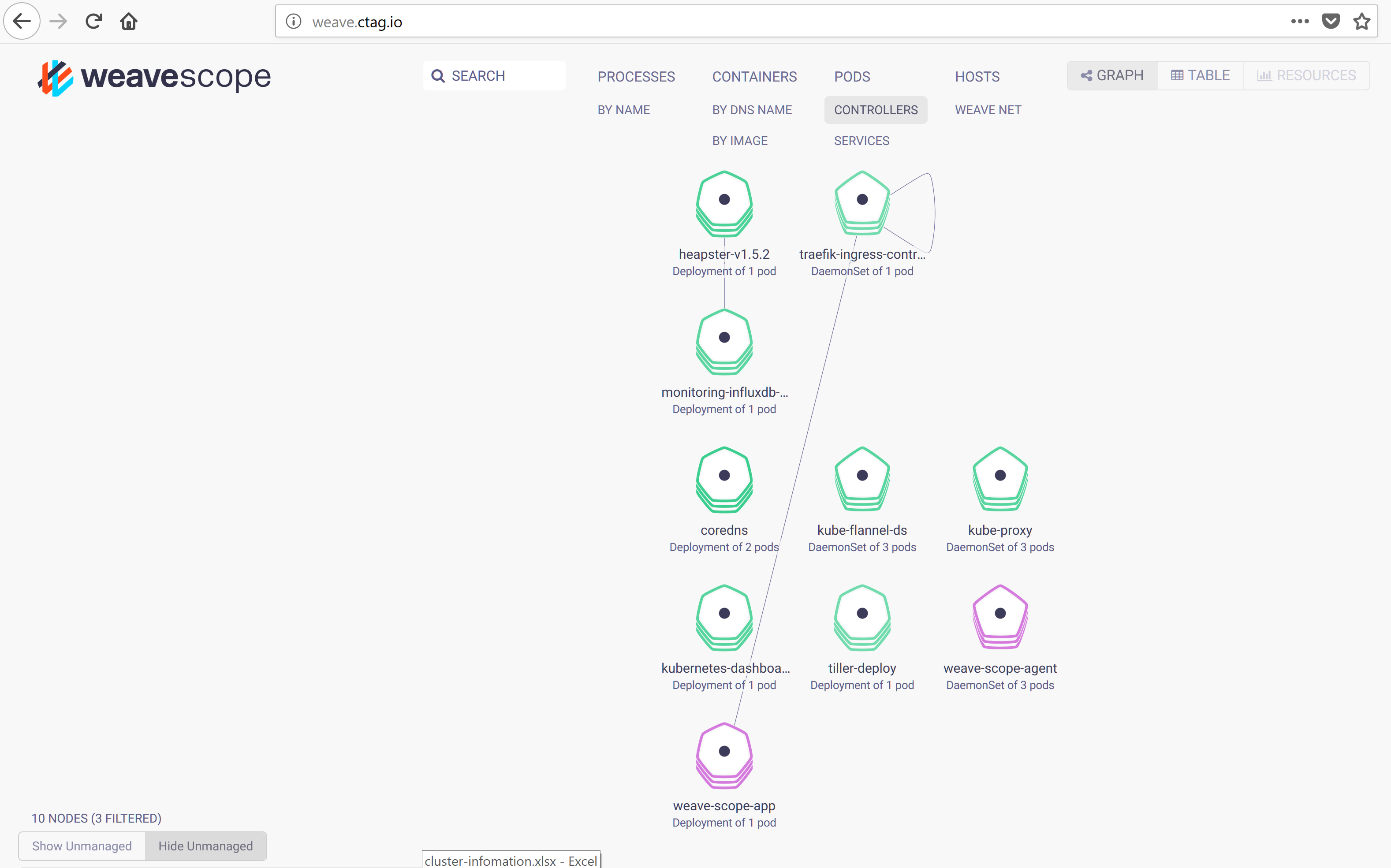


**如果登录token有问题，可以通过如下命令获得token：**

*kubectl -n kube-system describe secret `kubectl -n kube-system get secret|grep admin-token|cut -d " " -f1`|grep "token:"|tr -s " "|cut -d " " -f2*

**11. 访问weave scope控制台，可以查看依赖关系：**

[*http://weave.ctag.io*](http://weave.ctag.io)



**11. Service Mesh，默认并未安装Istio以及附带的BookInfo例子，可以登录到master节点然后运行：**

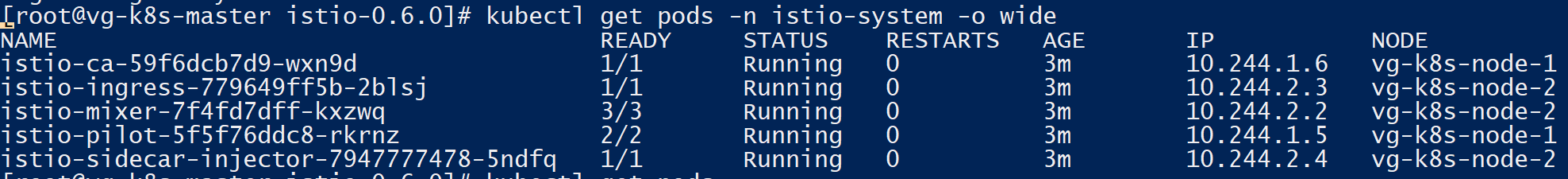
*sudo -i* (如果还未切换到root用户)

*cd /vagrant/yaml/istio-0.6.0*

*./install-service-mesh.sh*

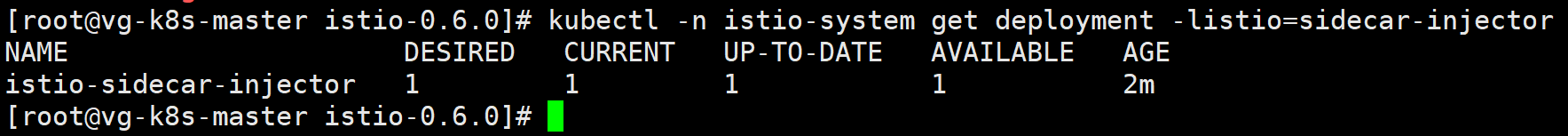
**运行完命令之后，通过下面的命令查询istio所有pod的状态：**

*kubectl get pods -n istio-system -o wide*



**当所有pod都变成Running状态后，确认一下sidecar injector webhook已经正常运行：**

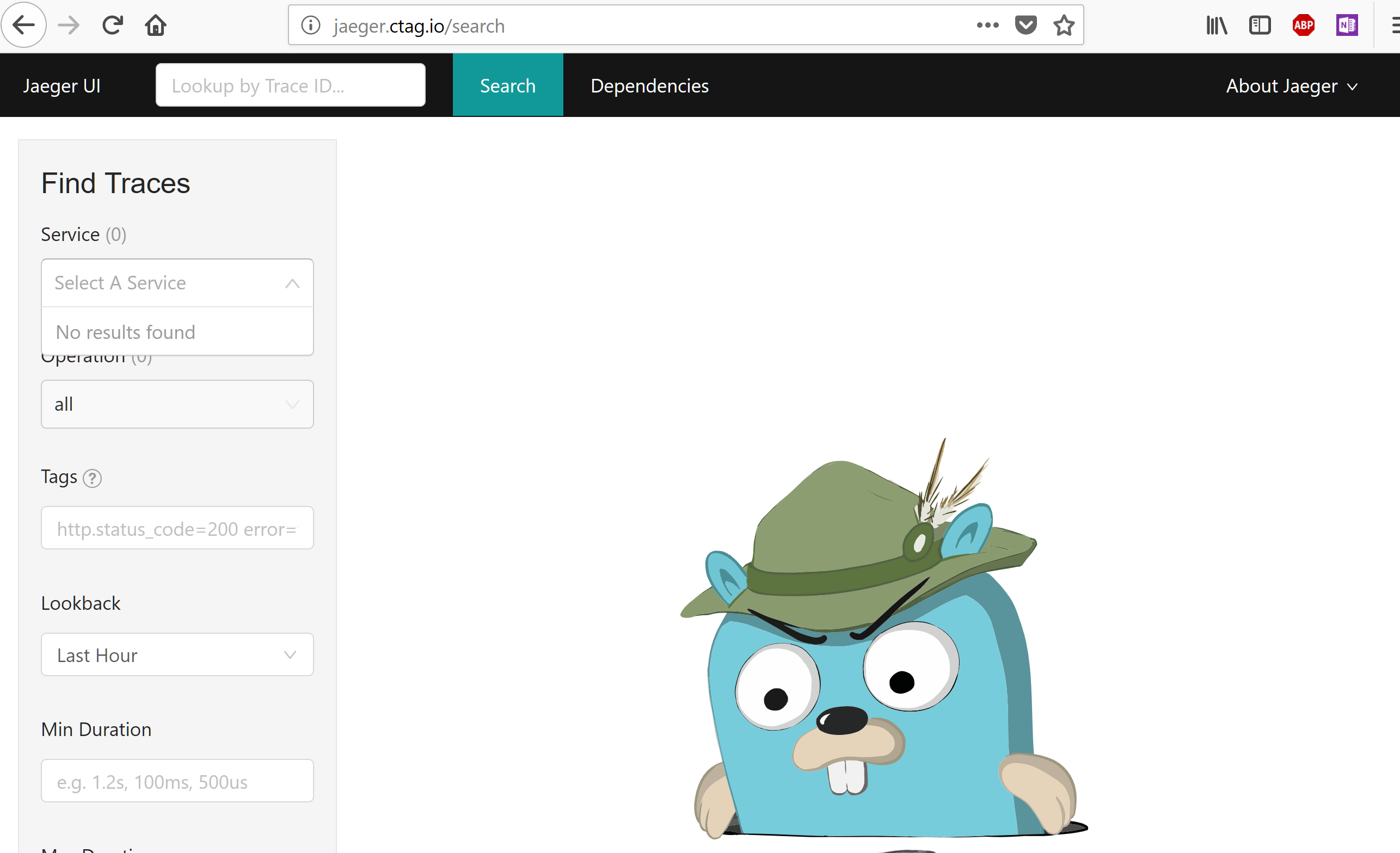
*kubectl -n istio-system get deployment -listio****=****sidecar-injector*



**12. 安装Jaeger（分布式追踪）**

*kubectl apply -f /vagrant/yaml/jaeger/jaeger-all-in-one-template.yml -n istio-system*

**安装完成后访问：**[*http://jaeger.ctag.io*](http://jaeger.ctag.io) **可以看到Jaeger的控制台**



**运行下面的命令来安装BookInfo例子：**

*cd /vagrant/yaml/istio-0.6.0*

*./install-bookinfo.sh*

**查询所有BookInfo的pod的状态：**

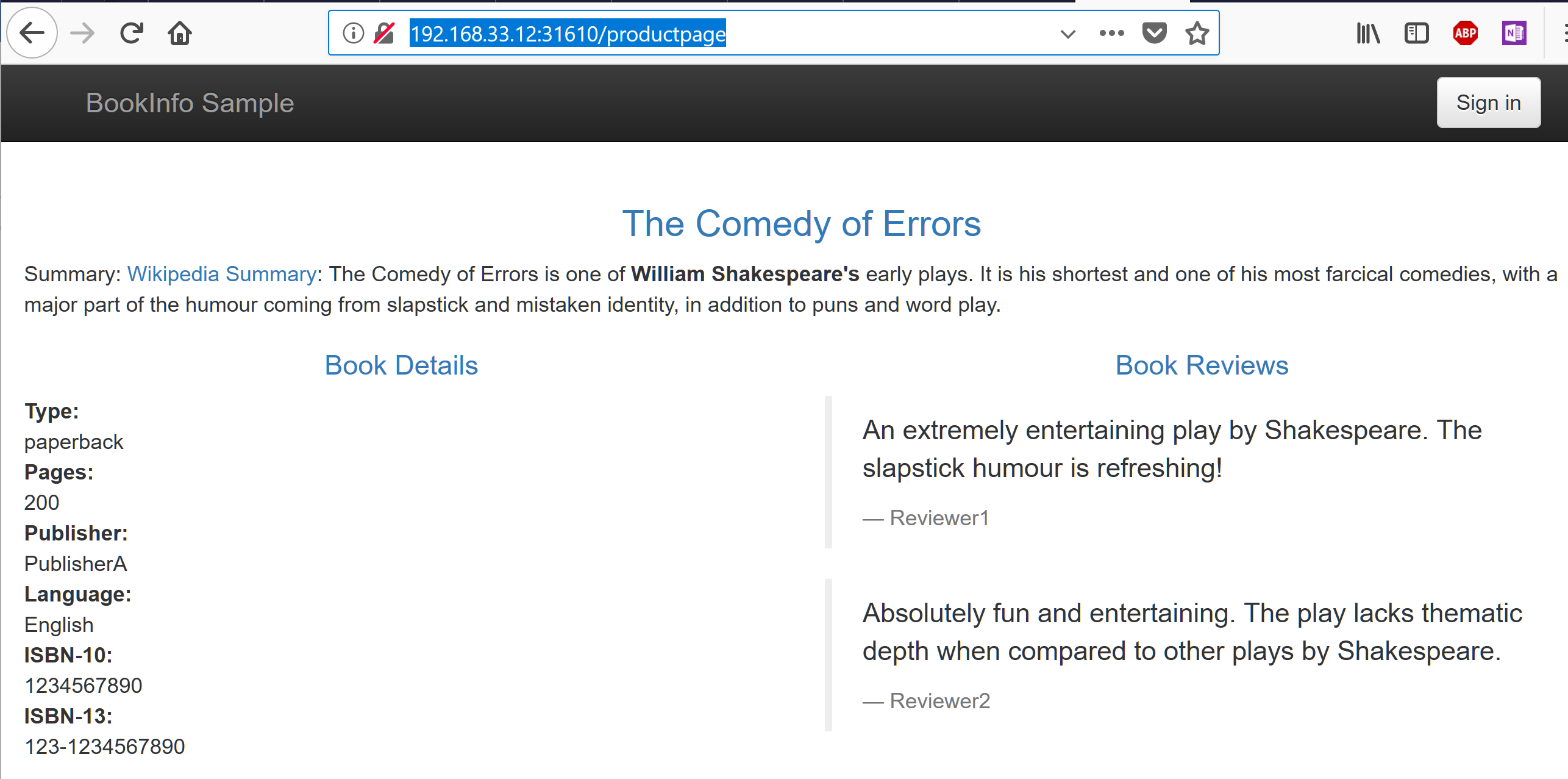
*kubectl get pods -o wide*



**当所有pod的状态变成Running后，可以通过如下命令查询product page的地址及端口：**

*echo $(kubectl get po -l istio=ingress -n istio-system -o 'jsonpath={.items[0].status.hostIP}'):$(kubectl get svc istio-ingress -n istio-system -o 'jsonpath={.spec.ports[0].nodePort}')*

**例如，上述命令的输出结果为：**192.168.33.12:31610**，则访问**[*http://192.168.33.12:31610/productpage*](http://192.168.33.12:31610/productpage)

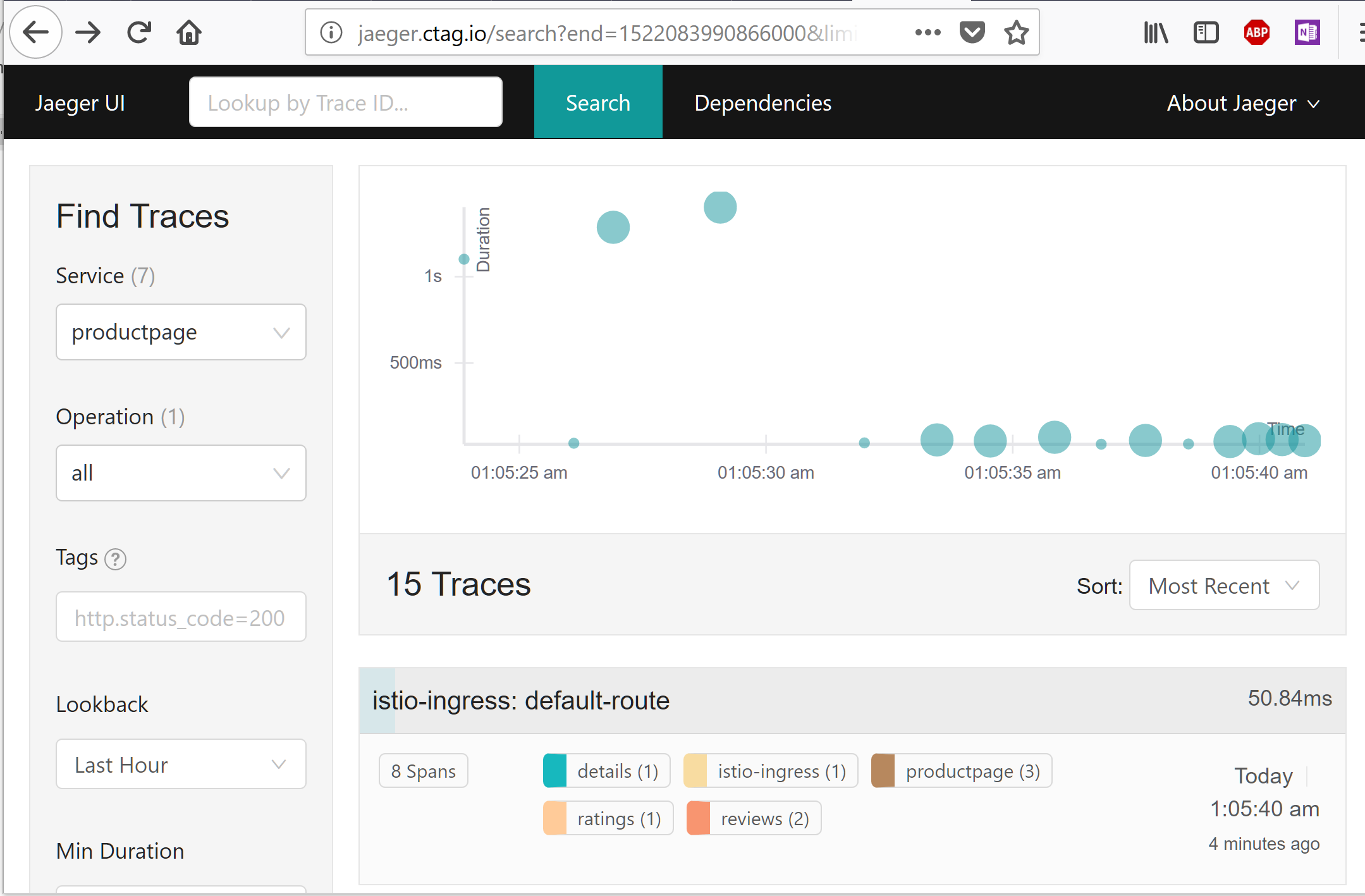


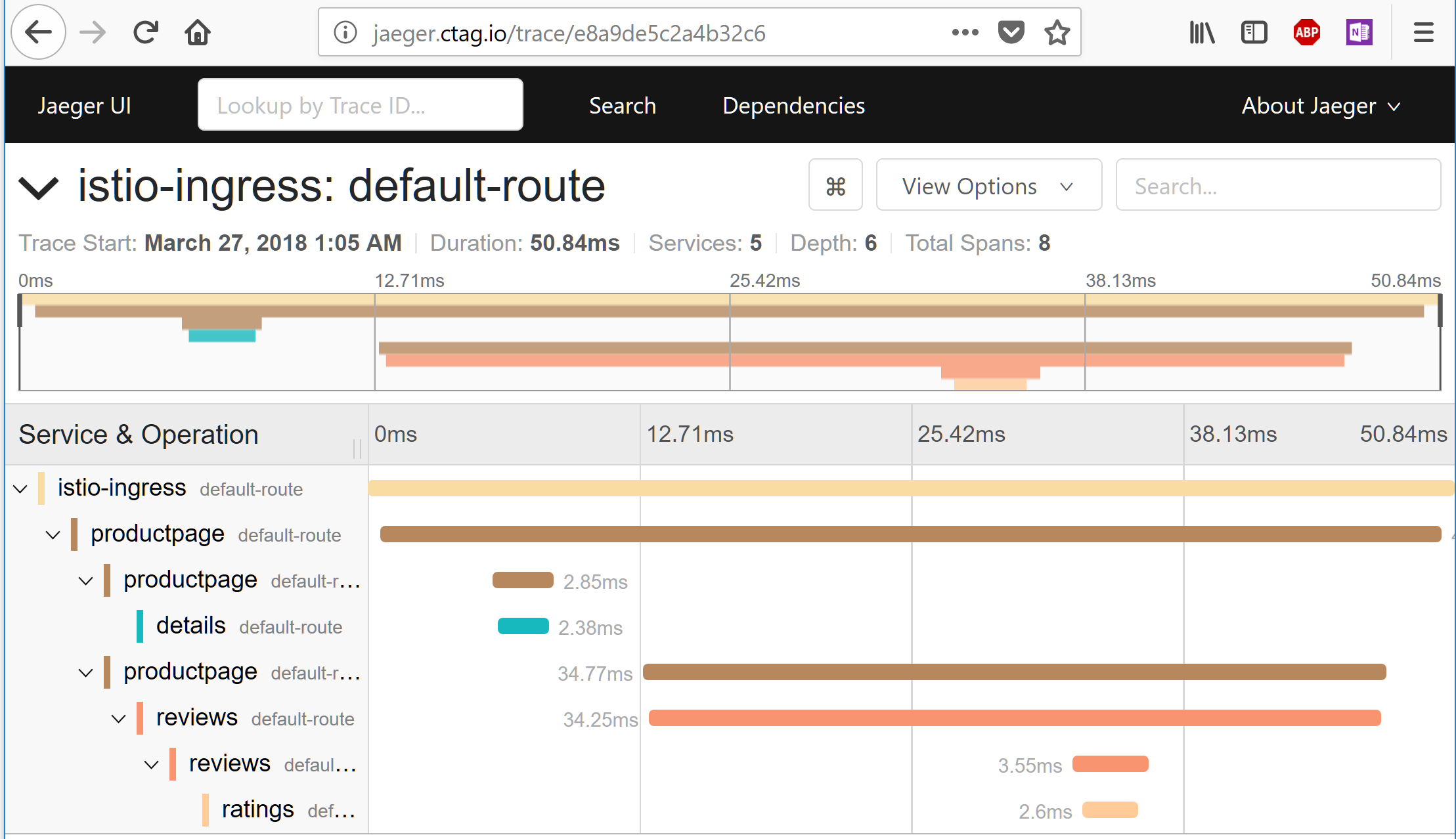
**有关BookInfo例子的详细信息，可以参考：**<https://istio.io/docs/guides/bookinfo.html>

**可以通过命令行来访问上面的product page或者在浏览器里刷新页面：**

*curl -o /dev/null -s -w "%{http\_code}\n"* [*http://192.168.33.12:31610/productpage*](http://192.168.33.12:31610/productpage)

**多调用或者多刷新几次，然后在Jaeger控制台里面可以查询Tracing log。**





**13. 想停掉整个集群的话，在命令行运行：**

*vagrant halt*

**14. 想彻底销毁掉整个集群的话：在命令行运行：**

*vagrant destroy*