CKA-考前辅导

讲师： 老段 RHCE/RHCA/COA/CKA

1.考试注册

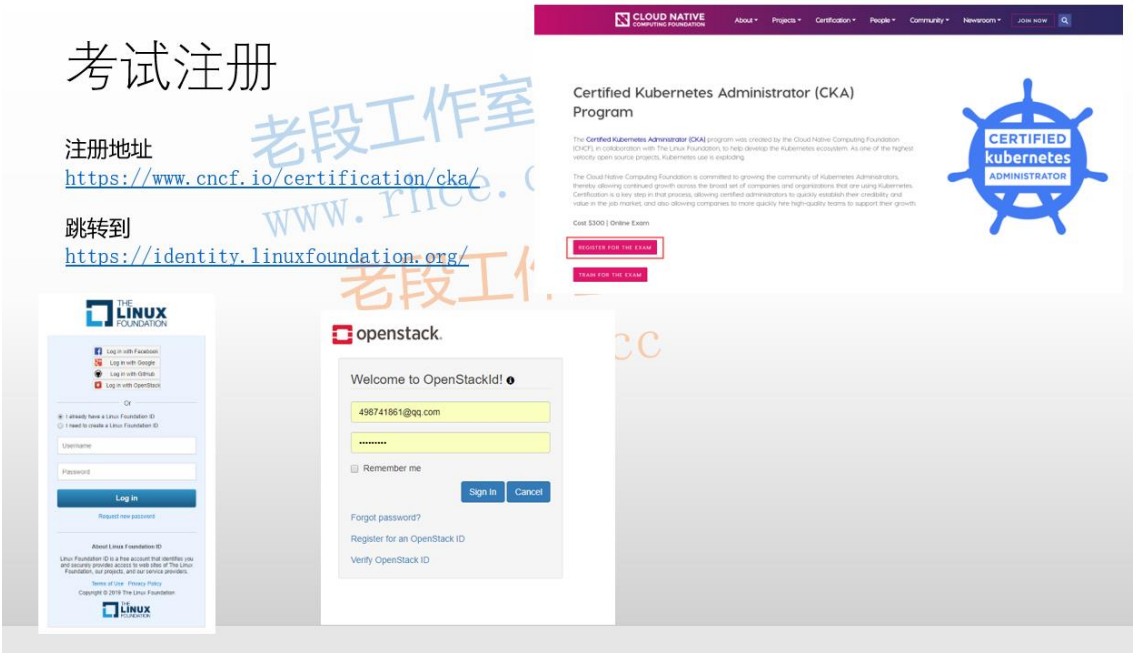
考试注册

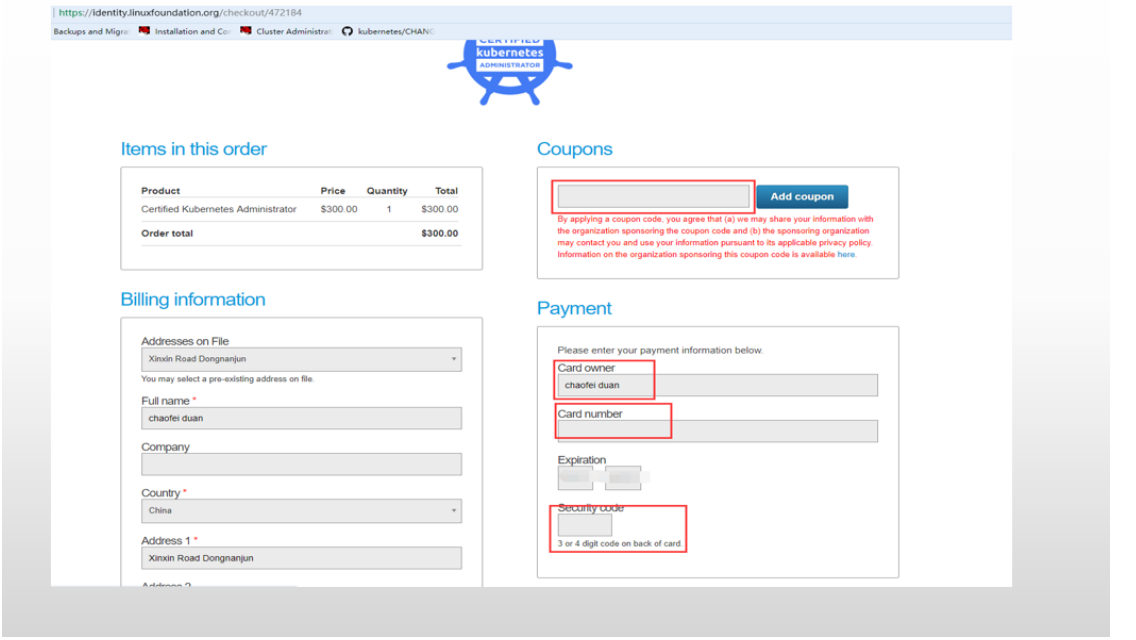
注册地址

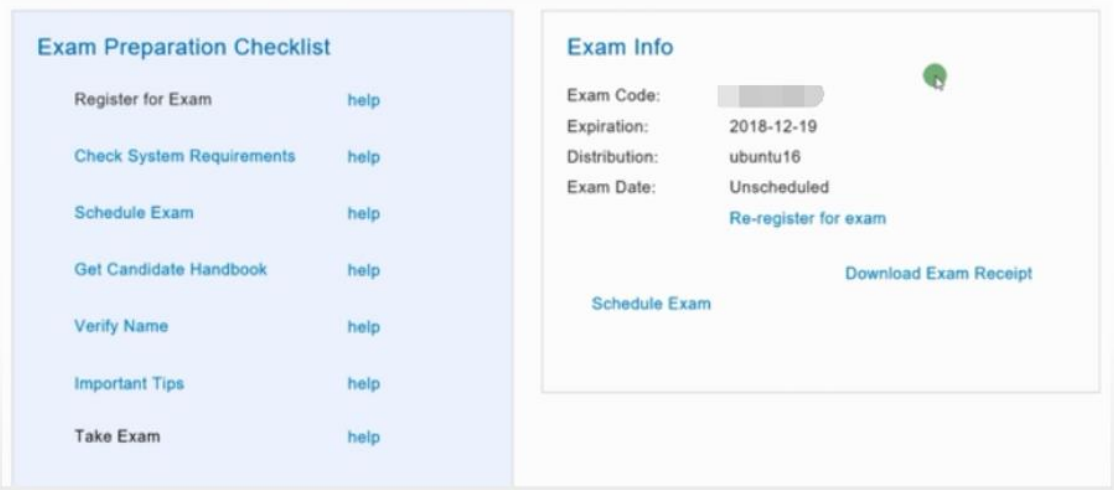
https://www.cncf.io/certification/cka/

跳转到

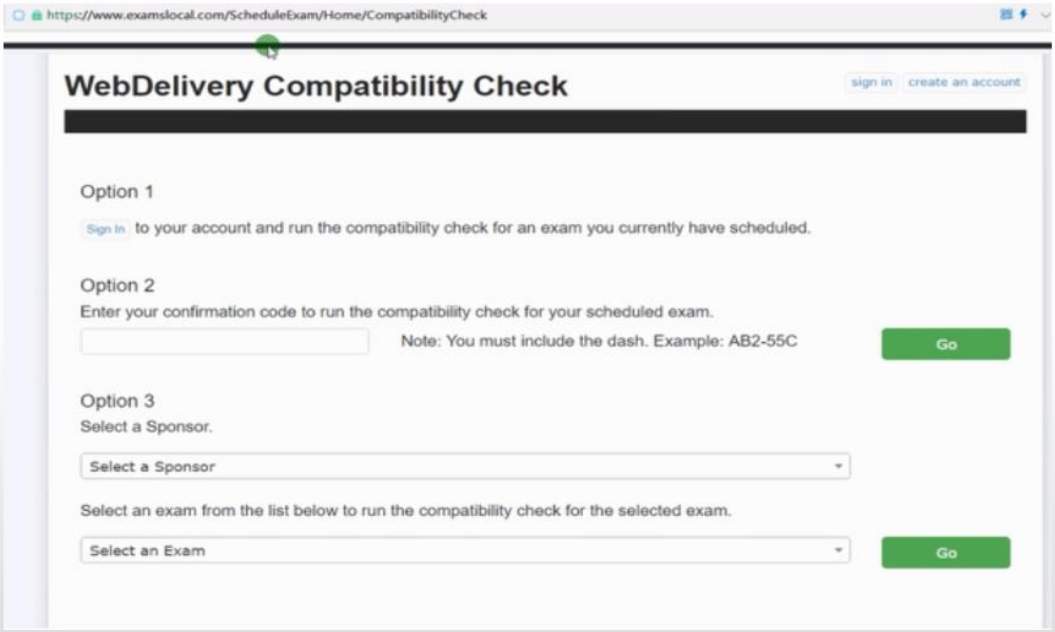
https://identity.linuxfoundation.org/

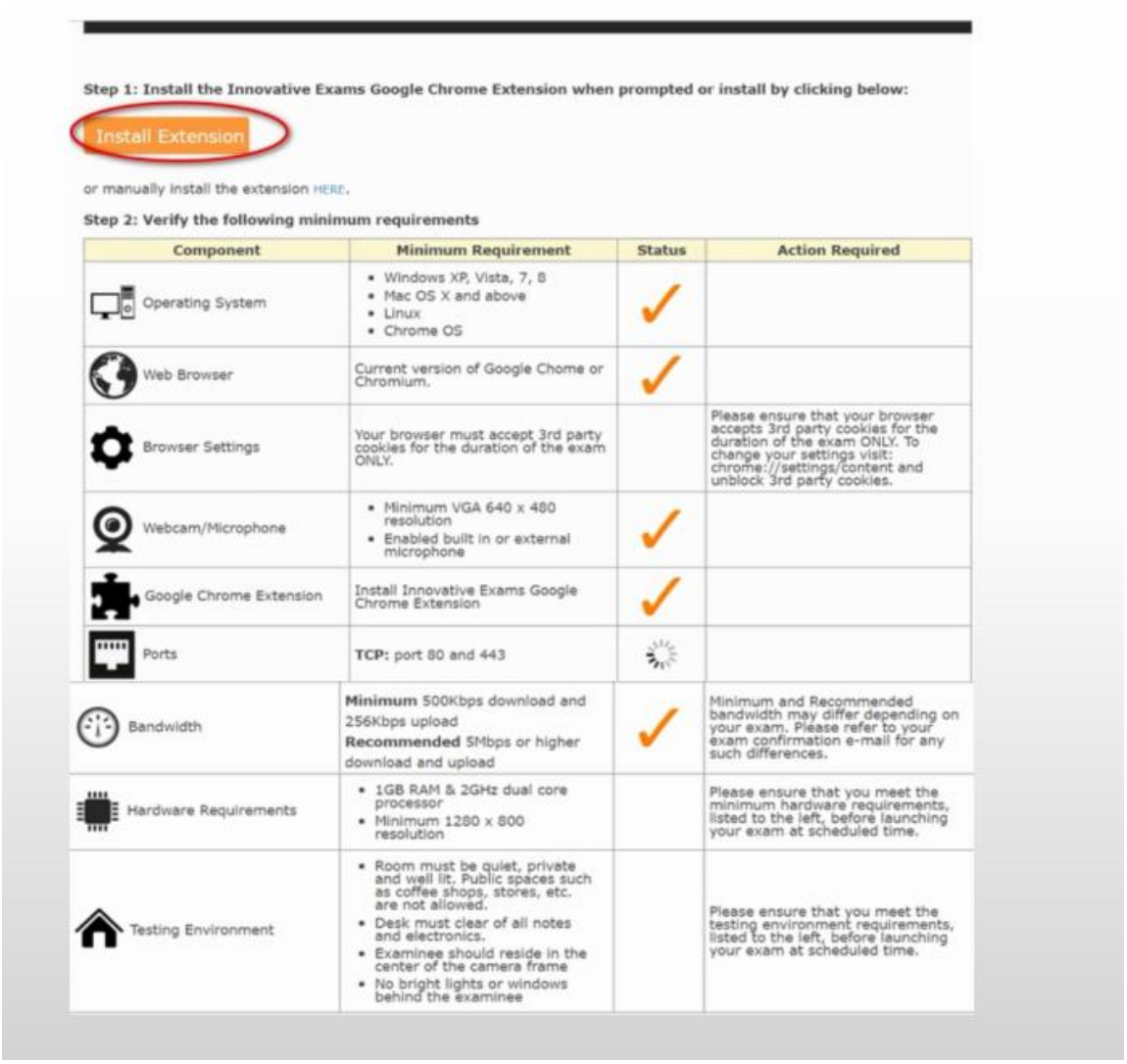






上图左侧的 "Check System Requirements" 是用来检测考试用的机器是否满足考试条件的，点击它，会打开一个新的页面：





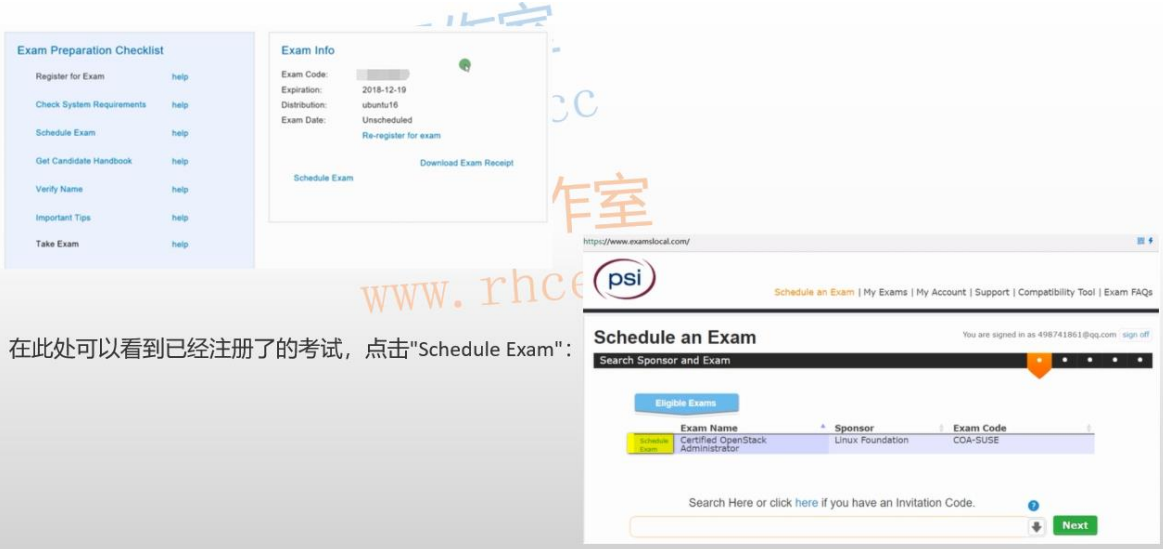
Step 1: Install the Innovative Exams Google Chrome Extension when prompted or install by clicking below:

Install Extension

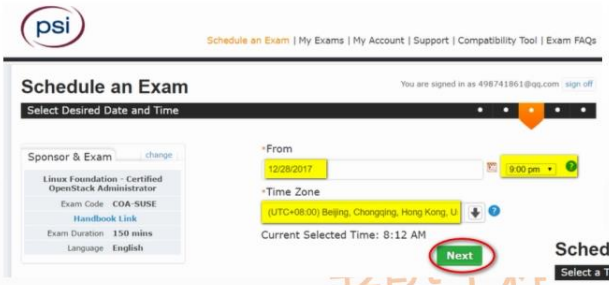
or manually install the extension HERE.

Step 2: Verify the following minimum requirements

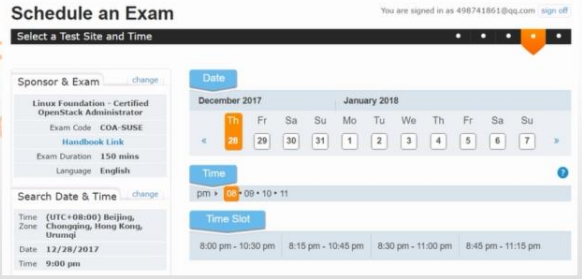
Component	Minimum Requirement	Status	Action Required
Operating System	• Windows XP, Vista, 7, 8 • Mac OS X and above • Linux • Chrome OS	✓	
Web Browser	Current version of Google Chome or Chromium.	✓	
Browser Settings	Your browser must accept 3rd party cookies for the duration of the exam ONLY.		Please ensure that your browser accepts 3rd party cookies for the duration of the exam ONLY. To change your settings visit: chrome://settings/content and unblock 3rd party cookies.
Webcam/Microphone	• Minimum VGA 640 x 480 resolution • Enabled built in or external microphone	✓	
Google Chrome Extension	Install Innovative Exams Google Chrome Extension	✓	
Ports	**TCP:** port 80 and 443		
Bandwidth	**Minimum** 500Kbps download and 256Kbps upload **Recommended** 5Mbps or higher download and upload	✓	Minimum and Recommended bandwidth may differ depending on your exam. Please refer to your exam confirmation e-mail for any such differences.
Hardware Requirements	• 1GB RAM & 2GHz dual core processor • Minimum 1280 x 800 resolution		Please ensure that you meet the minimum hardware requirements, listed to the left, before launching your exam at scheduled time.
Testing Environment	• Room must be quiet, private and well lit. Public spaces such as coffee shops, stores, etc. are not allowed. • Desk must clear of all notes and electronics. • Examinee should reside in the center of the camera frame • No bright lights or windows behind the examinee		Please ensure that you meet the testing environment requirements, listed to the left, before launching your exam at scheduled time.



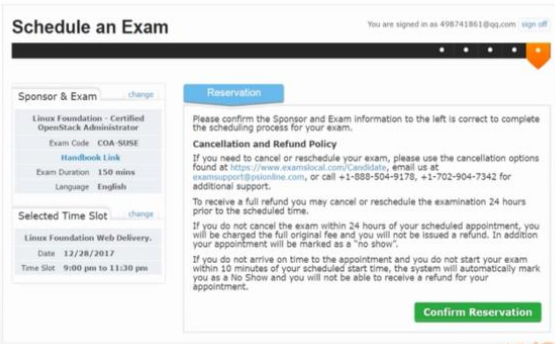
在此处可以看到已经注册了的考试，点击"Schedule Exam"：



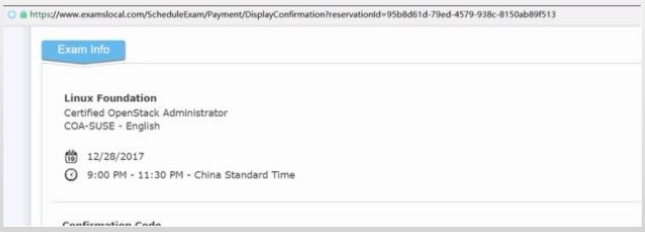
依次选择要选择的日期、时间和时区，选择Next:



再次更精确的选择，比如是晚上8:00开始，还是8:15开始等等，选好之后跳转到：



点击"Confirm Reservation",然后跳转到：



2.考题讲解

1.Set configuration context $ kubectl config use-context k8s
Monitor the logs of Pod foobar and Extract log lines corresponding to error *unable-to-access-website* Write them to /opt/KULM00201/foobar

答案
kubectl logs pod | grep '*unable-to-access-website*' > /opt/KULM00201/foobar

2.Set configuration context $ kubectl config use-context k8s
List all PVs sorted by **name**, saving the full kubectl output to /opt/KUCC0010/my_volumes . Use kubectl s own functionally for sorting the output, and do not manipulate it any further.

答案:

kubect get pv --all-namespaces --sort-by={.metadata.name} > /opt/KUCC0010/my_volumes

3. Set configuration context $ kubectl config use-context k8s
Ensure a single instance of Pod nginx is running on each node of the kubernetes cluster where nginx also represents the image name which has to be used. Do no override any taints currently in place.
Use **Daemon sets** to complete this task and use ds.kusc00201 as Daemonset name.

答案：

```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
 name: busybox
spec:
 template:
  metadata:
   labels:
    app: busybox
  spec:
   containers:
   - name: nginx
    image: nginx
```

4. Set configuration context $ kubectl config use-context k8s
Perform the following tasks
Add an init container to lumpy--koala (Which has been defined in spec file /opt/kucc00100/pod-spec-KUCC00100.yaml)
The init container should create an empty file named /workdir/calm.txt
If /workdir/calm.txt is not detected, the Pod should exit
Once the spec file has been updated with the init container definition, the Pod should be created.

老段工作室 www.rhce.cc

答案：

```
initContainers:
 - name: init-poda
  image: busybox
  command: ['sh', '-c', 'touch /workdir/calm.txt']
  volumeMounts:
  - name: workdir
   mountPath: "/workdir"
```

5. Set configuration context $ kubectl config use-context k8s
Create a pod named kucc4 with a single container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached + consul

答案

```
apiVersion: v1
kind: Pod
metadata:
 name: kucc4
 labels:
  role: myrole
spec:
 containers:
  - name: web1
   image: nginx
 containers:
  - name: web2
   image: redis
....
```

6. Set configuration context $ kubectl config use-context k8s
Schedule a Pod as follows:
Name: nginx-kusc00101
Image: nginx
Node selector: disk=ssd

答案:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc00101
  labels:
    role: myrole
spec:
  nodeSelector:
    disk: ssd
  containers:
    - name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
          protocol: TCP
```

7.Set configuration context $ kubectl config use-context k8s
Create a deployment as follows
Name: nginx-app
Using container nginx with version 1.11.9-alpine
The deployment should contain 3 replicas
Next, deploy the app with new version 1.12.0-alpine by performing a rolling update and record that update.
Finally, rollback that update to the previous version 1.11.9-alpine

答案: kubectl run nginx-app --image='nginx:1.11.9-alpine' --replicas=3

kubectl set image deployment/nginx-app nginx-app='nginx:1.12.0-alpine' --record=true

kubectl rollout undo deployment/nginx-app

8.Set configuration context $ kubectl config use-context k8s
Create and configure the service front-end-service so it's accessible through NodePort / ClusterIP and routes to the existing pod named front-end

答案:

kubectl expose pod front-end --name=front-end-service --type="NodePort" --port=80

9.Set configuration context $ kubectl config use-context k8s
Create a Pod as follows:
Name: jenkins
Using image: jenkins
In a new Kubenetes namespace named *website-frontend*

答案

```
apiVersion: v1
kind: Pod
metadata:
 name: jenkins
 labels:
  role: myrole1
spec:
 containers:
  - name: jenkins
    image: jenkins
```

kubectl create ns *website-frontend*
kubectl apply -f aa.yaml -n website-frontend

10. Set configuration context $ kubectl config use-context k8s
Create a deployment spec file that will:
Launch 7 replicas of the redis image with the label: app_env_stage=dev
Deployment name: kual00201
Save a copy of this spec file to /opt/KUAL00201/deploy_spec.yaml (or .json)
When you are done, clean up (delete) any new k8s API objects that you produced during this task

答案

```
kubectl run kual00201 --image=redis --replicas=7 --dry-run -o yaml > /opt/KUAL00201/deploy_spec.yaml
```

11. Set configuration context $ kubectl config use-context k8s
Create a file /opt/KUCC00302/kucc00302.txt that lists all pods that implement Service **foo** in Namespace **production**.
The format of the file should be one pod name per line.

答案

```
kubectl get svc --show-labels -n production

kubectl get pods -l name=haha -n production | grep -v NAME | awk '{print $1}' >> /opt/KUCC00302/kucc00302.txt
```

如果foo这个服务有多个标签的话，依次查找

12. Set configuration context $ kubectl config use-context k8s
Create a Kubernetes Secret as follows:
Name: super-secret
Credential: alice **or** username: bob
Create a Pod named ***pod-secrets-via-file*** using the **redis** image which mounts a secret named super-secret at /secrets
Create a second Pod named ***pod-secrets-via-env*** using the **redis** image, which exports **credential**/**username** as TOPSECRET / CREDENTIALS

```
echo -n 'bob'|base64
apiVersion: v1
kind: Secret
metadata:
 name: super-secret
type: Opaque
data:
 username: Ym9i
```

```
apiVersion: v1
kind: Pod
metadata:
 labels:
  run: nginx
 name: pod-secrets-via-file
spec:
 volumes:
 - name: xx
  secret:
   secretName: super-secret
 containers:
 - image: redis
  name: redis
  volumeMounts:
  - name: xx
   mountPath: "/secrets"
   readOnly: true
```

```
apiVersion: v1
kind: Pod
metadata:
 name: redis
 labels:
  name: redis
spec:
 containers:
 - image: redis
  name: redis
  env:
   - name: CREDENTIALS
    valueFrom:
     secretKeyRef:
      name: user
```

13. Set configuration context $ kubectl config use-context k8s
Create a pod as follows:
Name: non-persistent-redis
Container image: redis
Named-volume with name: cache-control
Mount path: /data/redis
It should launch in the pre-prod namespace and the volume MUST NOT be persistent.

答案:

```
apiVersion: v1
kind: Pod
metadata:
 name: non-persistent-redis
 labels:
  purpose: non-persistent-redis
spec:
 volumes:
 - name: cache-control
  emptyDir: {}
 containers:
 - name: redis
  image: redis
  volumeMounts:
  - mountPath: /data/redis
   name: cache-control
```

14. Set configuration context $ kubectl config use-context k8s
Scale the deployment **webserver** to **6** pods

答案

```
kubectl scale deployment/webserver --replicas=6
```

15. Set configuration context $ kubectl config use-context k8s
Check to see how many nodes are ready (not including nodes tainted NoSchedule) and write the number to /opt/nodenum

答案

```
kubectl get node | grep -i  ready | wc -l
```

16. Set configuration context $ kubectl config use-context k8s
From the Pod label **name=cpu-utilizer**, find pods running high CPU workloads and write the name of the Pod consuming most CPU to the file /opt/cpu.txt (which already exists)

答案:

```
kubectl top pods -l name=cpu-utilizer
```

```
echo ' kusc00201-5tzfk ' >> /opt/cpu.txt
```

如果要找出消耗CPU最高的node

```
kubectl top nodes
```

17. Set configuration context $ kubectl config use-context k8s
Create a deployment as follows
Name: nginx-dns
Exposed via a service: nginx-dns
Ensure that the service & pod are accessible via their respective DNS records
The container(s) within any Pod(s) running as a part of this deployment should use the **nginx** image
Next, use the utility nslookup to look up the DNS records of the service & pod and write the output to /opt/service.dns and /opt/pod.dns respectively.
Ensure you use the busybox:1.28 image (or earlier) for any testing, an the latest release has an unpstream bug which impacts the use of nslookup.

答案

```
kubectl run nginx-dns --image=nginx
kubectl expose  deployment  nginx-dns --port=80
kubectl get pods  -o wide 获取pod的IP
kubectl run busybox -it --rm --image=busybox:1.28 sh
    nslookup nginx-dns
    nslookup 100.92.90.6.default.pod.cluster.local
```

18.No configuration context change required for this item
Create a snapshot of the etcd instance running at https://127.0.0.1:2379 saving the snapshot to the file path /data/backup/etcd-snapshot.db
The etcd instance is running etcd version 3.2.18
The following TLS certificates/key are supplied for connecting to the server with etcdctl
CA certificate: /opt/KUCM00302/ca.crt
Client certificate: /opt/KUCM00302/etcd-client.crt
Client key: /opt/KUCM00302/etcd-client.key

答案

```
export ETCDCTL_API=3
etcdctl help   在etdctl snapshoht save --help
etcdctl --endpoints=https://127.0.0.1:2379 \
--ca-file=/opt/KUCM00302/ca.crt \
--certfile=/opt/KUCM00302/etcd-client.crt \
--key=/opt/KUCM00302/etcd-client.key snapshot save /data/backup/etcd-snapshot.db
```

19.Set configuration context $ kubectl config use-context *ek8s*
Set the node labelled with name=ek8s-node-1 as unavailable and reschedule all the pods running on it.

```
kubectl get node --show-labels |grep name=ek8s-node-1
#找出 node

kubectl drain ek8s-node-1
```

然后删除运行此node上的pod

20.Set configuration context $ kubectl config use-context *wk8s*
A Kubernetes worker node, labelled with **name=wk8s-node-0** is in state NotReady . Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.
Hints:
You can ssh to the failed node using $ ssh wk8s-node-0
You can assume elevated privileges on the node with the following command $ sudo -i

答案: 是kubelet 没有启动

```
kubectl get node
查看 一个 node 是 notReady ssh 上去
systemctl status kubelet 发 现 没 有 启 动
systemctl start kubelet;systemctl enable kubelet
```

21. Set configuration context $ kubectl config use-context *wk8s*
Configure the kubelet systemd managed service, on the node labelled with **name=wk8s-node-1**, to launch a Pod containing a single container of image nginx named myservice automatically. Any spec files required should be placed in the /etc/kubernetes/manifests directory on the node.
Hints:
You can ssh to the failed node using $ ssh wk8s-node-1
You can assume elevated privileges on the node with the following command $ sudo -i

答案：

在node上

systemctl status kubelet

找到
--pod-manifest-path
所对应的目录

在里面创建一个文件

```
apiVersion: v1
kind: Pod
metadata:
 name: static-web
 labels:
  role: myrole
spec:
 containers:
  - name: web
   image: nginx
   ports:
    - name: web
     containerPort: 80
     protocol: TCP
```

22. 题目很长忽略，建议不做

23.Set configuration context $ kubectl config use-context *bk8s*
Given a partially-functioning Kubenetes cluster, identify symptoms of failure on the cluster.
Determine the node, the failing service and take actions to bring up the failed service and restore the health of the cluster. Ensure that any changes are made permanently.
The worker node in this cluster is labelled with **name=bk8s-node-0**
Hints:
You can ssh to the relevant nodes using $ ssh $(NODE) where $(NODE) is one of **bk8s-master-0** or **bk8s-node-0**
You can assume elevated privileges on any node in the cluster with the following command
$ sudo -i

答案:　　是kube-manager-controller 没有启动 启动就做完了
kubectl get cs
能看到 controller manager 没有启动登陆到master 上

systemctl start kube-manager-controller.service

24.Set configuration context $ kubectl config use-context *hk8s*
Creae a persistent volume with name app-config of capacity 1Gi and access mode ReadWriteOnce. The type of volume is hostPath and its location is /srv/app-config

答案:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: app-config
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /srv/app-config
```