

CKAD^{Q&As}

Certified Kubernetes Application Developer (CKAD) Program

Pass Linux Foundation CKAD Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.certbus.com/ckad.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Linux Foundation Official Exam Center

- ⚙ **Instant Download** After Purchase
- ⚙ **100% Money Back** Guarantee
- ⚙ **365 Days** Free Update
- ⚙ **800,000+** Satisfied Customers



QUESTION 1

Exhibit:



Task

A deployment is falling on the cluster due to an incorrect image being specified. Locate the deployment, and fix the problem.

- A. Please check explanations
- B. Place Holder

Correct Answer: AB

Pending

QUESTION 2

Exhibit:



Context

A pod is running on the cluster but it is not responding.

Task

The desired behavior is to have Kubernetes restart the pod when an endpoint returns an HTTP 500 on the /healthz endpoint. The service, probe-pod, should never send traffic to the pod while it is failing.

Please complete the following:

The application has an endpoint, /started, that will indicate if it can accept traffic by returning an HTTP 200.

If the endpoint returns an HTTP 500, the application has not yet finished initialization.

The application has another endpoint /healthz that will indicate if the application is still working as expected by returning an HTTP 200. If the endpoint returns an HTTP 500 the application is no longer responsive.

Configure the probe-pod pod provided to use these endpoints .

The probes should use port 8080.

A. Please check explanations

B. Place Holder

Correct Answer: AB

Solution: In the configuration file, you can see that the Pod has a single Container. The periodSeconds field specifies that the kubelet should perform a liveness probe every 5 seconds. The initialDelaySeconds field tells the kubelet that it should wait 5 seconds before performing the first probe. To perform a probe, the kubelet executes the command cat /tmp/healthy in the target container. If the command succeeds, it returns 0, and the kubelet considers the container to be alive and healthy. If the command returns a nonzero value, the kubelet kills the container and restarts it. When the container starts, it executes this command: /bin/sh -c "touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600" For the first 30 seconds of the container's life, there is a /tmp/healthy file. So during the first 30 seconds, the command cat /tmp/healthy returns a success code. After 30 seconds, cat /tmp/healthy returns a failure code. Create the Pod: kubectl apply -f https://k8s.io/examples/pods/probe/exec-liveness.yaml Within 30 seconds, view the Pod events: kubectl describe pod liveness-exec The output indicates that no liveness probes have failed yet: FirstSeen LastSeen Count From SubobjectPath Type Reason Message -----

scheduler } Normal Scheduled Successfully assigned liveness- exec to worker0 23s 23s 1 {kubelet worker0}
spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox" 23s 23s 1 {kubelet worker0}
spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox" 23s 23s 1 {kubelet worker0}
spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined] 23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container
with docker id 86849c15382e After 35 seconds, view the Pod events again: kubectl describe pod liveness-exec At the
bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been
killed and recreated. FirstSeen LastSeen Count From SubobjectPath Type Reason Message -----

37s 37s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness- exec to
worker0 36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox" 36s
36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "k8s.gcr.io/busybox" 36s
36s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e;
Security:[seccomp=unconfined] 36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container
with docker id 86849c15382e 2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe
failed: cat: can't open \"/tmp/healthy\": No such file or directory Wait another 30 seconds, and verify that the container
has been restarted: kubectl get pod liveness-exec The output shows that RESTARTS has been incremented: NAME
READY STATUS RESTARTS AGE liveness-exec 1/1 Running 1 1m

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: liveness-exec
spec:
  containers:
  - name: liveness
    image: k8s.gcr.io/busybox
    args:
    - /bin/sh
    - -c
    - touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600
    livenessProbe:
      exec:
        command:
        - cat
        - /tmp/healthy
      initialDelaySeconds: 5
      periodSeconds: 5
```

QUESTION 3

Context Anytime a team needs to run a container on Kubernetes they will need to define a pod within which to run the container. Task Please complete the following: Create a YAML formatted pod manifest /opt/KDPD00101/pod1.yml to create a pod named app1 that runs a container named app1cont using image lfcncf/arg-output with these command line arguments: -lines 56 -F Create the pod with the kubectl command using the YAML file created in the previous step When the pod is running display summary data about the pod in JSON format using the kubectl command and redirect the output to a file named /opt/KDPD00101/out1.json All of the files you need to work with have been created, empty, for your convenience

When creating your pod, you do not need to specify a container command, only args.



A. Please check explanations

B. Place Holder

Correct Answer: AB

Solution:

```
student@node-1:~$ kubectl run appl --image=lfcencf/arg-output --dry-run=client -o yaml > /opt/KD  
PD00101/pod1.yml  
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: appl
    name: appl
spec:
  containers:
  - image: lfccncf/arg-output
    name: appl
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}

```

"/opt/KDPD00101/pod1.yml" 15L, 242C 3,1 All

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: appl
    name: appl
spec:
  containers:
  - image: lfccncf/arg-output
    name: appl
    args: ["--image", "arg", "--s"]

```

11,30 All


```
pod/app1 created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS             RESTARTS   AGE
app1          0/1     ContainerCreating   0           5s
counter       1/1     Running             0           4m44s
liveness-http 1/1     Running             0           6h50m
nginx-101     1/1     Running             0           6h51m
nginx-configmap 1/1     Running             0           6m21s
nginx-secret   1/1     Running             0           11m
poller        1/1     Running             0           6h51m
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
app1          1/1     Running   0           26s
counter       1/1     Running   0           5m5s
liveness-http 1/1     Running   0           6h50m
nginx-101     1/1     Running   0           6h51m
nginx-configmap 1/1     Running   0           6m42s
nginx-secret   1/1     Running   0           12m
poller        1/1     Running   0           6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
```

Readme Web Terminal

```
nginx-configmap 1/1 Running 0 6m2
nginx-secret 1/1 Running 0 11m
poller 1/1 Running 0 6h5
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
app1          1/1     Running   0           26s
counter       1/1     Running   0           5m5s
liveness-http 1/1     Running   0           6h50m
nginx-101     1/1     Running   0           6h51m
nginx-configmap 1/1     Running   0           6m42s
nginx-secret   1/1     Running   0           12m
poller        1/1     Running   0           6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
student@node-1:~$ kubectl create -f /opt/KDPD00101/pod1.yml
pod/app1 created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
app1          1/1     Running   0           20s
counter       1/1     Running   0           6m57s
liveness-http 1/1     Running   0           6h52m
nginx-101     1/1     Running   0           6h53m
nginx-configmap 1/1     Running   0           8m34s
nginx-secret   1/1     Running   0           14m
poller        1/1     Running   0           6h53m
student@node-1:~$ kubectl get pod app1 -o json >
```

```

poller          1/1      Running      0          6h51m
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
app1          1/1     Running   0          26s
counter       1/1     Running   0          5m5s
liveness-http 1/1     Running   0          6h50m
nginx-101     1/1     Running   0          6h51m
nginx-configmap 1/1     Running   0          6m42s
nginx-secret   1/1     Running   0          12m
poller        1/1     Running   0          6h51m
student@node-1:~$ kubectl delete pod app1
pod "app1" deleted
student@node-1:~$ vim /opt/KDPD00101/pod1.yml
student@node-1:~$ kubectl create -f /opt/KDPD00101/pod1.yml
pod/app1 created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
app1          1/1     Running   0          20s
counter       1/1     Running   0          6m57s
liveness-http 1/1     Running   0          6h52m
nginx-101     1/1     Running   0          6h53m
nginx-configmap 1/1     Running   0          8m34s
nginx-secret   1/1     Running   0          14m
poller        1/1     Running   0          6h53m
student@node-1:~$ kubectl get pod app1 -o json > /opt/KDPD00101/out1.json
student@node-1:~$
student@node-1:~$

```

QUESTION 4

Exhibit: Context Developers occasionally need to submit pods that run periodically. Task Follow the steps below to create a pod that will start at a predetermined time and which runs to completion only once each time it is started: Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: date in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be hello Create the resource in the above manifest and verify that the job executes successfully at least once

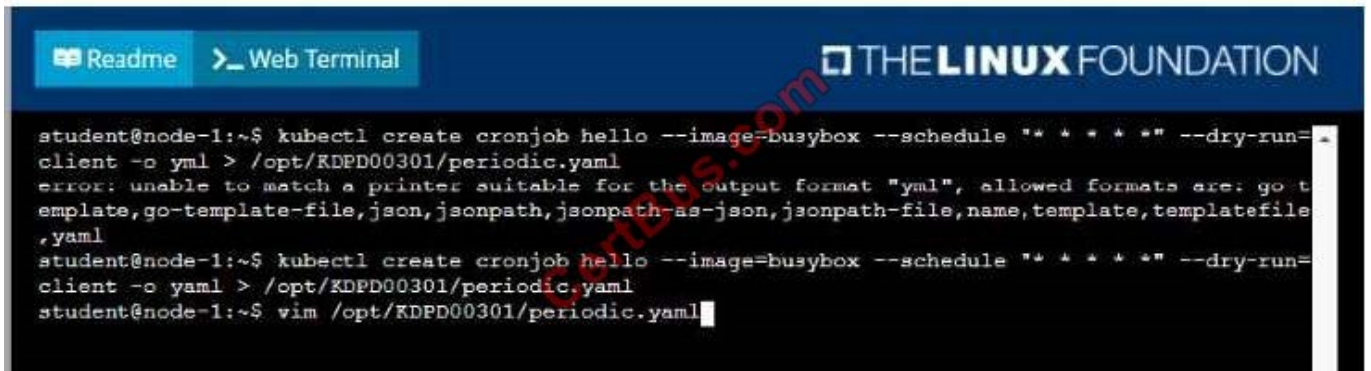


A. Please check explanations

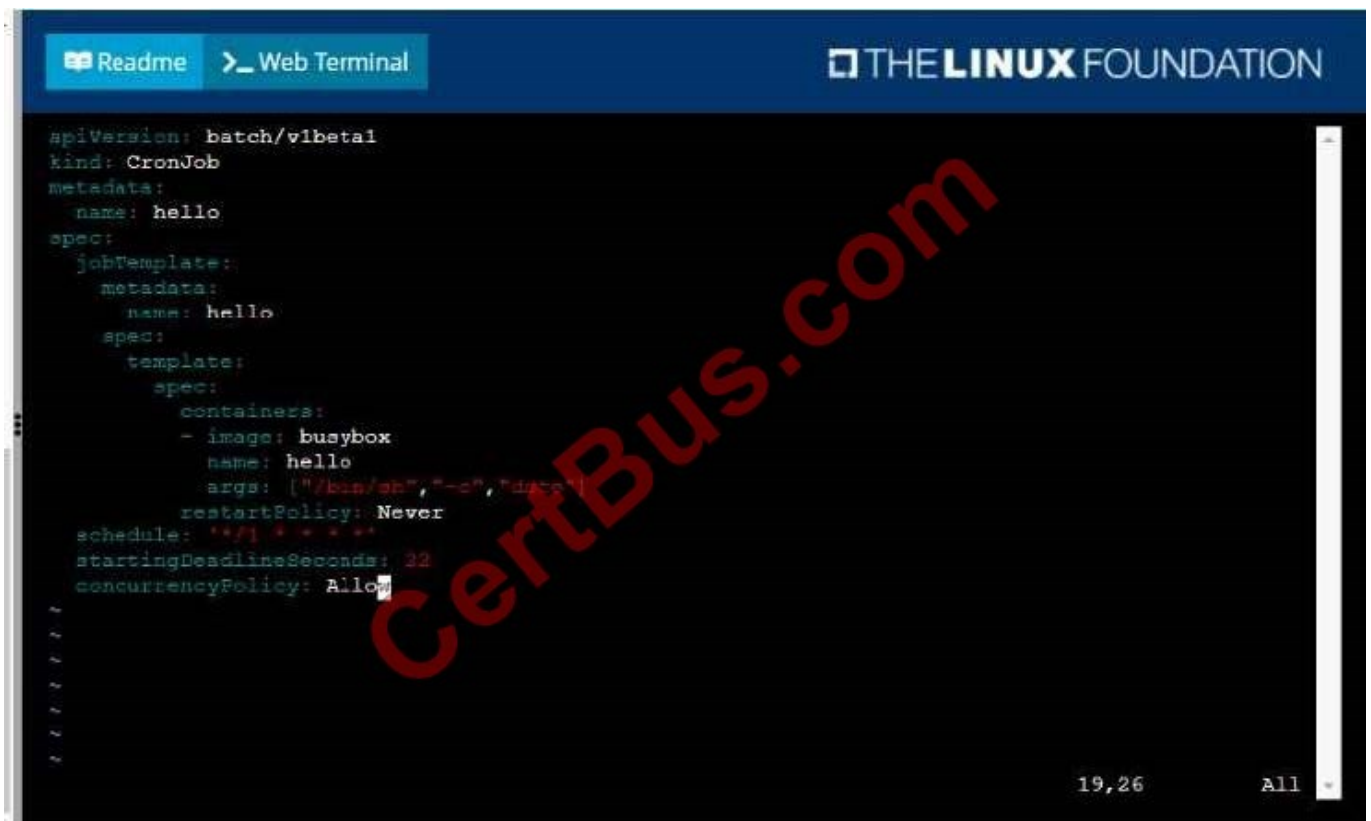
B. Place Holder

Correct Answer: AB

Solution:



The screenshot shows a terminal window with a dark background. At the top, there are two buttons: 'Readme' and 'Web Terminal'. The title bar says 'THE LINUX FOUNDATION'. The terminal text shows a user attempting to create a cronjob named 'hello' using the 'kubectl' command. The command is: `kubectl create cronjob hello --image=busybox --schedule '* * * * *' --dry-run=client -o yaml > /opt/KDPD00301/periodic.yaml`. The output is an error message: `error: unable to match a printer suitable for the output format "yaml", allowed formats are: go template, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile, yaml`. The user then repeats the command, and the terminal shows the command being executed.



The screenshot shows a terminal window with a dark background. At the top, there are two buttons: 'Readme' and 'Web Terminal'. The title bar says 'THE LINUX FOUNDATION'. The terminal text shows the YAML configuration for a CronJob. The configuration is as follows:

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
          restartPolicy: Never
  schedule: '* * * * *'
  startingDeadlineSeconds: 32
  concurrencyPolicy: Allow
```

```

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME          SCHEDULE          SUSPEND   ACTIVE   LAST SCHEDULE   AGE
hello         */1 * * * *       False     0        <none>           6s
student@node-1:~$

```

QUESTION 5

Exhibit:



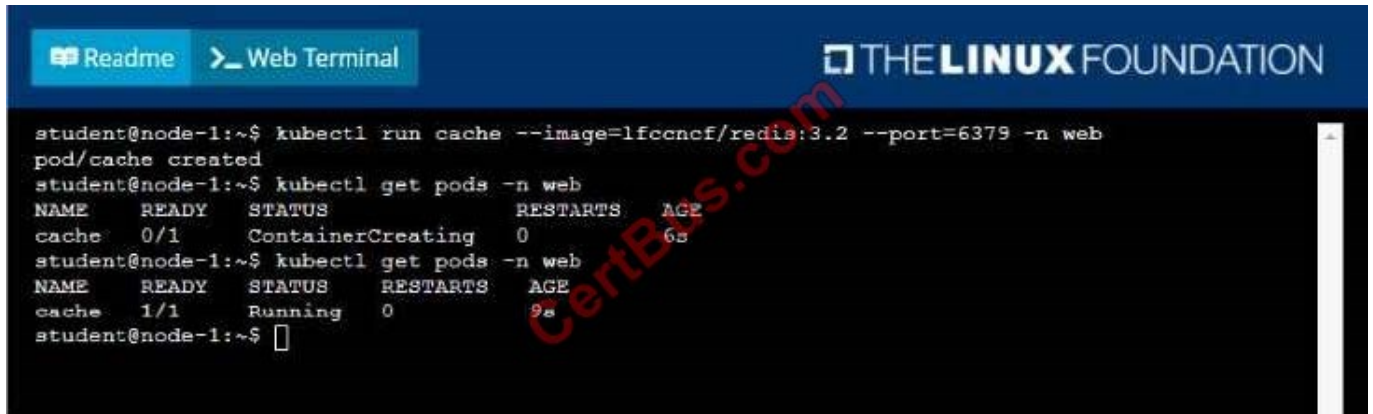
Context A web application requires a specific version of redis to be used as a cache. Task Create a pod with the following characteristics, and leave it running when complete: The pod must run in the web namespace. The namespace has already been created The name of the pod should be cache Use the lffcnf/redis image with the 3.2 tag Expose port 6379

A. Please check explanations

B. Place Holder

Correct Answer: AB

Solution:



```
student@node-1:~$ kubectl run cache --image=lfcncf/redis:3.2 --port=6379 -n web
pod/cache created
student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS             RESTARTS   AGE
cache     0/1     ContainerCreating   0           6s
student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS    RESTARTS   AGE
cache     1/1     Running   0           9s
student@node-1:~$
```

[CKAD VCE Dumps](#)

[CKAD Exam Questions](#)

[CKAD Braindumps](#)

To Read the [Whole Q&As](#), please purchase the [Complete Version](#) from [Our website](#).

Try our product !

100% Guaranteed Success

100% Money Back Guarantee

365 Days Free Update

Instant Download After Purchase

24x7 Customer Support

Average **99.9%** Success Rate

More than **800,000** Satisfied Customers Worldwide

Multi-Platform capabilities - **Windows, Mac, Android, iPhone, iPod, iPad, Kindle**

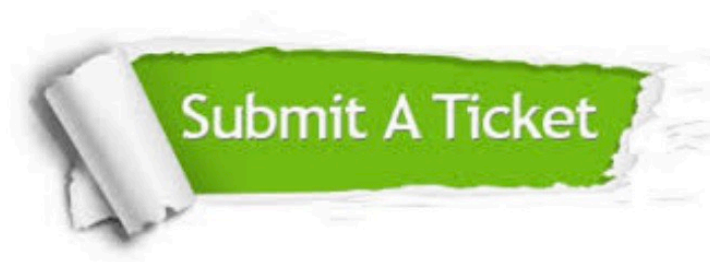
We provide exam PDF and VCE of Cisco, Microsoft, IBM, CompTIA, Oracle and other IT Certifications.
You can view Vendor list of All Certification Exams offered:

<https://www.certbus.com/allproducts>

Need Help

Please provide as much detail as possible so we can best assist you.

To update a previously submitted ticket:



 One Year Free Update Free update is available within One Year after your purchase. After One Year, you will get 50% discounts for updating. And we are proud to boast a 24/7 efficient Customer Support system via Email.	 Money Back Guarantee To ensure that you are spending on quality products, we provide 100% money back guarantee for 30 days from the date of purchase.	 Security & Privacy We respect customer privacy. We use McAfee's security service to provide you with utmost security for your personal information & peace of mind.
---	---	--

Any charges made through this site will appear as Global Simulators Limited.

All trademarks are the property of their respective owners.

Copyright © certbus, All Rights Reserved.