

# CKA and CKAD exam preparation notes

v1 February 19, 2021 – initial version

Reid Peryam – CKA certified 1/21, CKAD certified 2/21

I am not a Kubernetes expert, I started learning k8s September 2020 from K21 Academy by enrolling in CKA and CKAD courses. Hopefully, my perspectives and knowledge for preparing for CKA and CKAD exams will be useful to other students of K21 because I took the same training courses as they did and went on to prepare for and eventually pass the CKA and CKAD certification exams.

## Thoughts for both exams:

One important thing that many people may not know – it is not enough to be familiar with topics or answers. You really must have them memorized. Exactly how to solve each question, and very quickly. My advice is not to schedule the exam until you are very confident that you can answer every practice question you encounter quickly and with no research or uncertainty. Probably still there will be unexpected exam questions or topics but they will not prohibit you from passing if you are confident and prepared for everything else. K21 may not communicate effectively (in my opinion) how much effort it takes to go from complete k8s noob (like me) to passing both CKA and CKAD exams. In my case it was a lot of work, self-study and learning.

## CKA exam:

The first time I took the CKA exam I failed with a score of 49/100. After that I realized that the K21 portal had practice questions and exams to study (whoops!) – I drilled them multiple times and the second time I took the CKA exam I scored a 62/100 – the passing score is 66; I was extremely disappointed because I was sure that I did everything perfectly. After I failed the second time, I was more careful with details of questions and answers to avoid unintentional mistakes while studying more practice questions.

For instance, `kubect exec -it` into a created pod to verify command output was as expected, looking at pod logs, verifying service endpoints were created, remembering to create a service for a network policy etc. I found subtle mistakes that I was doing and not realizing – each time I figured out I was missing something it was really the best way to learn and unfortunately it requires some mistakes to happen to learn properly. The third time I took the exam I passed.

## CKA questions and resources

The practice questions and exams that were added to the CKA course are quite good and thorough. I did have a couple of questions that I was not prepared for including using service accounts and also secrets.

Here are some additional outside resources that you may find beneficial in your CKA exam preparation.

### Kubernetes - Google Sheets

*Very good for associating k8s docs to concepts and learning to use bookmarks. Also a tab for CKAD exam topics.*

[google.com](https://www.google.com)



**How I passed the CKA (Certified Kubernetes Administrator) Exam | by Nilesh Jayanandana | Platfor**



[medium.com](https://medium.com)

**Killer Shell - CKS CKA CKAD Simulator**

*costs \$30 USD for two practice exams.  
The questions are very difficult.*

[killer.sh](https://killer.sh)

## CKAD exam

I felt that the CKAD exam very difficult – after taking the exam I was sure I had failed it. I was surprised to learn that I actually had passed. I believe the CKAD exam questions are more difficult and complex than the CKA questions. There is more configuration of existing cluster resources across namespaces and troubleshooting failure. You are writing a lot of command line code quickly and constantly just to get the work done. Also, instead of creating single k8s objects you are creating sometimes four objects (configmaps, volumes, env vars, pod and service) and then configuring them all to work together as directed or troubleshooting why they do not. So instead of understanding one concept or yaml file, you need to understand the concepts and configurations of how to allow them to operate with others – that takes deeper understanding. I only had 3 minutes left at the end of the exam to answer the final three questions – there is less time to complete all of the tasks compared to the CKA exam and not once did I have time to verify or double-check my answer.

## Exam Preparation Resouces

### CKAD questions

Some topics that were covered in the exam that I was not really prepared for include Ambassador pattern, the streaming side car log adapter pattern (hard!!) and configuring a pod and service to use an existing network policy deployed on the cluster. Yes there are topics and practice questions surrounding these from K21 as well as outside resources but the questions in the exam were more difficult and complex than the examples that I studied.

Here are some additional outside resources that you may find beneficial in your CKA exam preparation

**Practice Enough With These 150 Questions for the CKAD Exam | by Bhargav Bachina | Bachina Labs |**



[medium.com](https://medium.com)

## dgkanatsios/CKAD-exercises: A set of exercises to prepare for Certified Kubernetes Application D

(in my opinion the single best resource)

[github.com](https://github.com/dgkanatsios/CKAD-exercises)



## Chrome Bookmarks

Something that I feel is an absolute requirement to do that is not mentioned much is to create a very thorough list of bookmarks in your Chrome browser to target each exam (the exam only allows you to use Chrome browser). Have a folder created on your Chrome bookmarks bar like this:

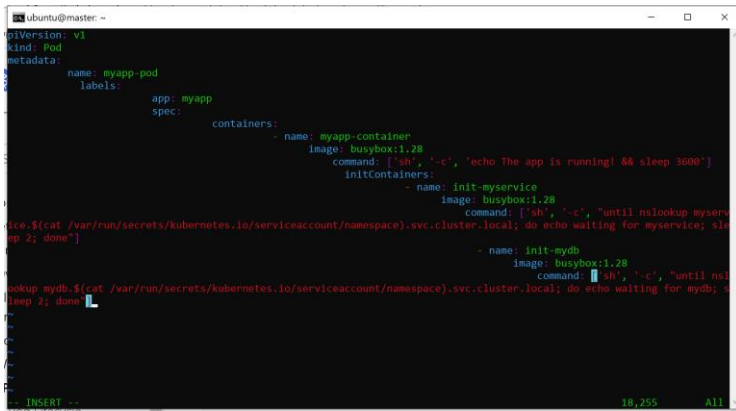
```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
    - image: k8s.gcr.io/test-webserver
      name: test-container
      volumeMounts:
        - mountPath: /cache
          name: cache-volume
  volumes:
    - name: cache-volume
      emptyDir: {}
```

This way, when an exam question mentions a specific object or configuration you can browse to the required yaml very quickly to copy and paste into your K8s object. Usually, you will copy it into the exam-provided notepad application first to modify it before pasting into your email console's vim editor. Note that this is not cheating, it is legal and accepted. Though it isn't readily advertised or acknowledged. Note that you can only use bookmarks from the official K8s documentation! I have no idea how anyone would pass these exams without first configuring proper bookmarks.

Because of this I have included the bookmarks I used to pass both exams here for you. Feel free to import them into Chrome and edit, delete or re-order them as you need. You will first need to learn what they are and what questions they are useful for to utilize them effectively – so practice! Here is how to import them into chrome:

## Azure AKS

Even though we setup and configured our own VM k8s clusters during the K21 CKA and CKAD courses, I found that these weren't very good for my exam study and preparation because of the following issue: when pasting yaml into a Windows cmd console window it gets mutated (this also happens in PowerShell):



```
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app: myapp
spec:
  containers:
    - name: myapp-container
      image: busybox:1.28
      command: ['sh', '-c', 'echo The app is running! && sleep 3000']
      initContainers:
        - name: init-myservice
          image: busybox:1.28
          command: ['sh', '-c', 'until nslookup myserv
          sleep 2; done']
        - name: init-mysdb
          image: busybox:1.28
          command: ['sh', '-c', 'until ns
          sleep 2; done']
```

Because of this I used an Azure K8s Cluster (AKS). There are videos on K21 portal for setting it up. But pay attention to how they are configured because the default settings will allocate very expensive VMs! Instead switch to the least expensive that will support K8s (B2s):

### Primary node pool

The number and size of nodes in the primary node pool in your cluster. For production workloads, at least 3 nodes are recommended for resiliency. For development or test workloads, only one node is required. If you would like to add additional node pools or to see additional configuration options for this node pool, go to the 'Node pools' tab above. You will be able to add additional node pools after creating your cluster. [Learn more about node pools in Azure Kubernetes Service](#)

Node size \* ⓘ

Standard B2s

2 vcpus, 4 GiB memory

[Change size](#)

Node count \* ⓘ

1

For CKA study you will want two nodes because many tasks involve sshing onto non-master nodes, but for CKAD all you need is 1 node because you won't need to switch between nodes.

Note that for CKA tasks involving upgrading kubeadm and backing up etcd, I use my own vm k8s cluster that we configured in K21 courses because AKS does not seem to allow for this.

## Kubectl Commands

Here are some kubectl commands that I find very useful for both CKA and CKAD exam preparation. Be able to do these with your eyes closed, from memory for different variables. [This resource that I linked above](#) is very good to use for practicing kubectl commands and instilling them into your memory. Also, for reference checkout the [Kubectl cheat sheet](#) of the official k8s documentation. This will give you ideas for what is possible and help you learn new ways of doing things using the command line.

Kubectl create [K8s object like deployment] [args] `--dry-run=client -o yaml > my-thing.xml`

Kubectl run pod [args] `--dry-run=client -o yaml > my-thing.xml`

This pattern is useful for generating k8s object “boiler plate” yaml code that can then be extended from official k8s documents for a specific task.

For instance, when a question is “create a deployment named deploy with an nginx image that exposes port 80 and consumes a persistent volume mounted at “/var/log”. You could:

Kubectl create deployment deploy --image=nginx --port=80 `--dry-run=client -o yaml > deploy.yml`

And then:

Vim deploy.yml

[To add the persistent volume yaml from the k8s docs.](#)

Note: sometimes it is faster just to copy and paste the code from the docs, modify it, before pasting into vim – especially for CKA exam. However, for CKAD exam they really have tasks to test your ability to generate complex objects from the command line – so it is a good habit to get very good with this pattern.

Kubectl create `-f my-thing.yml --dry-run=server`

This will check your yaml’s validity without actually creating the resource on the cluster. Better to find syntax issues in the yaml before you create it so that you do not need to troubleshoot the running instance. Most useful for complex objects and configurations for CKAD.

Kubectl create [x] `--help | grep [y] -i`

This will help you locate optional k8s command-line arguments from the K8s help. It retrieves the help documentation for a particular kubectl command and then filters (searches) on it. The -i flag tells it to ignore case in the filter.

For instance if you are creating a ConfigMap from a file but are uncertain to use the command-line argument --from-file= or --from-env-file= you can do this:

Kubectl create cm --help | grep from- -i

```
--from-env-file='': Specify the path to a file to read lines of key=val pairs to create a configmap (i.e. a Docker .env file).
--from-file=[]: Key file can be specified using its file path, in which case file basename will be used as configmap key, or optionally with a key and file path, in which case the given key will be used. Specifying a directory will iterate each named file in the directory whose basename is a valid configmap key.
```

You can now see that for config files containing data representing key value pairs, you should use --from-env-file

Kubectl create [x] `--help | grep Examples -i -A 15`

This is a really useful command for learning how to use the kubectl command line fast and effectively. What it does is retrieve the examples for creating a k8s object (and nothing else) by specifying -A and a number we can look for Examples and the next 15 lines. For instance:

```

# Start the nginx pod using a different command and custom arguments.
ubuntu@master:~$ kubectl run pod --help | grep Examples -A 26
Examples:
# Start a nginx pod.
kubectl run nginx --image=nginx

# Start a hazelcast pod and let the container expose port 5701.
kubectl run hazelcast --image=hazelcast/hazelcast --port=5701

# Start a hazelcast pod and set environment variables "DNS_DOMAIN=cluster" and "POD_NAMESPACE=default" in the container.
kubectl run hazelcast --image=hazelcast/hazelcast --env="DNS_DOMAIN=cluster" --env="POD_NAMESPACE=default"

# Start a hazelcast pod and set labels "app=hazelcast" and "env=prod" in the container.
kubectl run hazelcast --image=hazelcast/hazelcast --labels="app=hazelcast,env=prod"

# Dry run. Print the corresponding API objects without creating them.
kubectl run nginx --image=nginx --dry-run=client

# Start a nginx pod, but overload the spec with a partial set of values parsed from JSON.
kubectl run nginx --image=nginx --overrides='{ "apiVersion": "v1", "spec": { ... } }'

# Start a busybox pod and keep it in the foreground, don't restart it if it exits.
kubectl run -i -t busybox --image=busybox --restart=Never

# Start the nginx pod using the default command, but use custom arguments (arg1 .. argN) for that command.
kubectl run nginx --image=nginx -- <arg1> <arg2> ... <argN>

# Start the nginx pod using a different command and custom arguments.
kubectl run nginx --image=nginx --command -- <cmd> <arg1> ... <argN>

```

Kubectl describe [pod/deployment/etc] | grep image -i

Search against all the metadata from the describe command using this pattern

Kubectl describe [pod/deployment/etc] | grep events -i -A 20

Find events for an object – very useful for troubleshooting

```

ubuntu@master:~$ kubectl describe pod -n kdpd00201 frontend-5bfc4c9c7-57sz5 | grep Events -A 15
Events:
  Type     Reason             Age          From          Message
  ----     -
Normal    SandboxChanged     24h (x5 over 24h)    kubelet       Pod sandbox changed, it will be killed and re-created.
Normal    Pulled              24h          kubelet       Container image "nginx:1.13.7" already present on machine
Normal    Created             24h          kubelet       Created container nginx
Normal    Started             24h          kubelet       Started container nginx
Normal    SandboxChanged     44m (x7 over 45m)    kubelet       Pod sandbox changed, it will be killed and re-created.
Normal    Pulled              44m          kubelet       Container image "nginx:1.13.7" already present on machine
Normal    Created             44m          kubelet       Created container nginx
Normal    Started             44m          kubelet       Started container nginx

```

Kubectl describe [x] | grep deployment/environment/label -i -A 10

Get describe metadata you are looking for

Kubectl describe node | grep taint -i

Check which nodes have taints

Kubectl get pod SOME\_POD\_NAME -o yaml > some-pod-copy.yaml

This command copies an existing resource into a new yaml file. In CKAD exam you need to modify existing resources on the cluster. It is better to first copy them, edit the yaml and then verify that it can be created as expected (see the --dry-run=server command above) before deleting the existing resource.

## Conclusion

I hope this is a useful resource for your exam preparations. I will be starting AZ 303 and AZ 304 courses through K21 next. If you would like to connect on LinkedIn that would be great as I am always looking to expand my network and connect with learners like you and me. You can find me on LinkedIn [here](#).