LFS458

Kubernetes Administration

Version 1.30.1

THE LINUX FOUNDATION | Training & Certification

Version 1.30.1

Published by:

the **Linux Foundation**
https://www.linuxfoundation.org

Nondisclosure of Confidential Information

"Confidential Information" shall not include any of the following, even if marked confidential or proprietary: (a) information that relates to the code base of any open source or open standards project (collectively, "Open Project"), including any existing or future contribution thereto; (b) information generally relating or pertaining to the formation or operation of any Open Project; or (c) information relating to general business matters involving any Open Project.

This course does not include confidential information, nor should any confidential information be divulged in class.

Contents

List of Figures

Chapter 1

Closing and Evaluation Survey

1.1 Evaluation Survey

Evaluation Survey

Thank you for taking this course brought to you by **The Linux Foundation**.

Your comments are important to us and we take them seriously, both to measure how well we fulfilled your needs and to help us improve future sessions.

- Please Evaluate your training using the link your instructor will provide.
- Please be sure to check the spelling of your name and use correct capitalization as this is how your name will appear on the certificate.
- This information will be used to generate your **Certificate of Completion**, which will be sent to the email address you have supplied, so make sure it is correct.

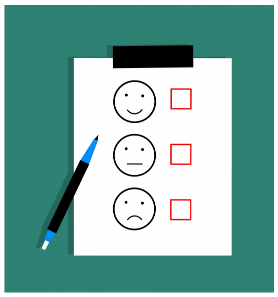
Figure 1.1: **Course Survey**

Appendices

Appendix A

Domain Review

A.1 CKA Exam

Domain Review

- Review the Candidate Handbook
- Find the proper version of the Curriculum Overview
- Write out each bullet point and steps
- Recognize good YAML examples in documentation
- Practice at speed

Navigate to the https://www.cncf.io/certification/cka/ page and scroll down to look at the `Exam Resources` section. Read through each of the resources as you prepare for the CKA exam. Begin by looking through the **Candidate Handbook**. You can either view it as several pages, or download the entire handbook as a pdf file. Read through each section. Take note of the `Resources allowed during exam` section as this will list the URLs and browser you can use for the exam. Read the rest of the document carefully.

Also navigate to the **Exam Curriculum** https://github.com/cncf/curriculum page. You should find PDF files for the CKA, CKAD, and the CKS exam. There may be more than one PDF. Ensure the PDF you use matches the version of the exam you are going to take. These update on a regular basis. It is your responsibility to revisit and ensure you are looking at the correct version of the file.

Write out each of the knowledge, skills and abilities listed in the curriculum. Ensure you know the command line steps to complete each of the items. **Warning:** The word `understand` means more than you have a general idea. It means you are able to create, integrate, troubleshoot, and properly remove that object.

The exam now uses a secure browser, so you will not be able to use your own bookmarks. It is important you are familiar with the allowed documentation, so that you can use known, working YAML with minimal searching. Test each for the version of Kubernetes the exam is using. If the version changes you must re-check each YAML file to ensure it still works. It is better to know it works than discover it during the exam and have to troubleshoot the issue.

Many people simply run out of time during the exam. Use a timer and practice the curriculum at the pace necessary to complete each item during the exam, and have enough time to verify and troubleshoot your own work.

A.2 Exam Domain Review

Exercise A.1: Are you Ready?

1. If you have not searched for working and tested YAML examples, and can easily search for them again for all of the subjects the domain review mentions for the exam, you may:

   a. Run out of time while searching for good YAML examples.

   b. Make more mistakes.

   c. Use YAML that isn't proper for the Kubernetes version of the exam and waste time trying to troubleshoot the issue.

   d. All of the above.

2. Answer all that apply. In the context of the `Curriculum Overview` the term **Understand** when stating what a candidate should be able to do means:

   a. You only have a general idea what the object does.

   b. You can create the object.

   c. You can configure and integrate the object with other objects.

   d. You can properly update and test the object.

   e. You can troubleshoot the object.

3. Have you practiced creating, integrating, and troubleshooting all of the domain review items **at speed**?

   a. No. I kind of did the exercises. So I'm good.

   b. No. I did the labs twice over a two week period.

   c. Yes. I know the exam is intense and practiced with a clock running to make sure I can get everything done and also check my work.

Solution A.1

Are You Ready?

1. d.
2. b, c, d, e.
3. Hopefully c.

Exercise A.2: Preparing for the CKA Exam

Very Important

The source pages and content in this review could change at any time. **IT IS YOUR RESPONSIBILITY TO CHECK THE CURRENT INFORMATION.**

Before Taking Exam

Use this exercise as a resource after you complete the course but before you take the exam. Review the resources, know what good YAML looks like, and practice creating and working with objects at exam speed to assist with review and preparation.

1. Using a browser go to https://www.cncf.io/certification/cka/ and read through the program description.

2. In the **Exam Resources** section open the Curriculum Overview and Candidate-handbook in new tabs. Both of these should be read and understood prior to sitting for the exam.
3. Navigate to the Curriculum Overview tab. You should see links for domain information for various versions of the exam. Select the latest version, such as **CKA_Curriculum_V1.25.pdf**. The versions you see may be different. You should see a new page showing a PDF.
4. Read through the document. Be aware that the term `Understand`, such as `Understand Services`, is more than just knowing they exist. In this case expect it to also mean create, configure, update, and troubleshoot.
5. Using only the exam-allowed URLs and sub-domains search for YAML examples for each domain or skill item. Ensure it works for the version of the exam you are taking, as the YAML may not have been re-tested after a new release. Become familiar with out to find each good example again, so you can find the page again during the exam.
6. Using a timer see how long it takes you to create and verify the objects listed below. Write down the time. Try it again and see how much faster you can complete and test each step.

”Practice until you get it right. Then practice until you can’t get it wrong” -Unknown

Domain Review Items

This list is copied from competency domains found on the PDF. Again, **it remains your responsibility to check the web page for any changes to this list**.

- **Cluster Architecture, Installation & Configuration**
  - Manage role based access control (RBAC)
  - Use Kubeadm to install a basic cluster
  - Manage a highly-available Kubernetes cluster
  - Provision underlying infrastructure to deploy a Kubernetes cluster
  - Preform a version upgrade on a Kubernetes cluster using Kubeadm
  - Implement etcd backup and restore
- **Workloads & Scheduling**
  - Understand deployments and how to preform rolling updates and rollbacks
  - Use ConfigMaps and Secrets to configure applications
  - Know how to scale applications
  - Understand the primitives used to create robust, self-healing, application deployments
  - Understand how resource limits can affect Pod scheduling
  - Awareness of manifest management and common templating tools
- **Services & Networking**
  - Understand host networking configuration on the cluster nodes
  - Understand connectivity between Pods
  - Understand ClusterIP, NodePort, LoadBalancer service types and endpoints
  - Know how to use Ingress controllers and Ingress resources
  - Know how to configure and use CoreDNS
  - Choose an appropriate container network interface plugin
- **Storage**
  - Understand storage classes, persistent volumes

  - Understand volume mode, access modes and reclaim policies for volumes
  - Understand persistent volume claims primitive
  - Know how to configure applications with persistent storage

- **Troubleshooting**
  - Evaluate cluster and node logging
  - Understand how to monitor applications
  - Manage container stdout & stderr logs
  - Troubleshoot application failure
  - Troubleshoot cluster component failure
  - Troubleshoot networking

Exercise A.3: Practicing Skills

This exercise is to help you practice your skills. It does not cover all the items listed in the domain review guide. You should develop your own steps to build a full list of skill tests and steps.

Also note that all the detailed steps are not included. You should be able to complete these steps without being told what to type.

In a work or exam environment you may not be told exactly what to do or how to do it. The following steps are meant to get you used to thinking about solutions when the exact need isn't clear.

1. Find and use the `review1.yaml` file included in the course tarball. Use the **find** output and copy the YAML file to your home directory. Use **kubectl create** to create the object. Determine if the pod is running. Fix any errors you may encounter. The use of **kubectl describe** may be helpful.

```
student@cp:~$ find ~ -name review1.yaml

student@cp:~$ cp <copy-paste-from-above>  .

student@cp:~$ kubectl create -f review1.yaml
```

2. After you get the pod running remove any pods or services you may have created as part of the review before moving on to the next section. For example:

```
student@cp:~$ kubectl delete -f review1.yaml
```

3. Use the `review2.yaml` file to create a non-working deployment. Fix the deployment such that both containers are running and in a `READY` state. The web server listens on port 80, and the proxy listens on port 8080.

4. View the default page of the web server. When successful verify the `GET` activity logs in the container log. The message should look something like the following. Your time and IP may be different.

```
192.168.124.0 - - [3/Dec/2020:03:30:31 +0000] "GET / HTTP/1.1" 200 612 "-"
"curl/7.58.0" "-"
```

5. Find and use the `review4.yaml` file to create a pod, and verify it's running

6. Edit the pod such that it only runs on your worker node using the `nodeSelector` label.

7. Determine the CPU and memory resource requirements of `design-pod1`.

8. Edit the pod resource requirements such that the CPU limit is exactly twice the amount requested by the container. (Hint: subtract .22)

9. Increase the memory resource limit of the pod until the pod shows a `Running` status. This may require multiple edits and attempts. Determine the minimum amount necessary for the `Running` status to persist at least a minute.

10. Use the `review5.yaml` file to create several pods with various labels.

11. Using **only** the –selector value `tux` to delete only those pods. This should be half of the pods. Hint, you will need to view pod settings to determine the key value as well.

12. Create a new cronjob which runs `busybox` and the `sleep 30` command. Have the cronjob run every three minutes. View the job status to check your work. Change the settings so the pod runs 10 minutes from the current time, every week. For example, if the current time was 2:14PM, I would configure the job to run at 2:24PM, every Monday.

13. Delete any objects created during this review. You may want to delete all but the `cronjob` if you'd like to see if it runs in 10 minutes. Then delete that object as well.

14. Create a new secret called `specialofday` using the key `entree` and the value `meatloaf`.

15. Create a new deployment called `foodie` running the `nginx` image.

16. Add the `specialofday` secret to pod mounted as a volume under the `/food/` directory.

17. Execute a bash shell inside a `foodie` pod and verify the secret has been properly mounted.

18. Update the deployment to use the `nginx:1.12.1-alpine` image and verify the new image is in use.

19. Roll back the deployment and verify the typical, current stable version of nginx is in use again.

20. Create a new 200M NFS volume called `reviewvol` using the NFS server configured earlier in the lab.

21. Create a new PVC called `reviewpvc` which will uses the `reviewvol` volume.

22. Edit the deployment to use the PVC and mount the volume under `/newvol`

23. Execute a bash shell into the nginx container and verify the volume has been mounted.

24. Delete any resources created during this review.

25. Create a new deployment which uses the `nginx` image.

26. Create a new `LoadBalancer` service to expose the newly created deployment. Test that it works.

27. Create a new NetworkPolicy called `netblock` which blocks all traffic to pods in this deployment only. Test that all traffic is blocked to deployment.

28. Create a pod running nginx and ensure traffic can reach that deployment.

29. Update the `netblock` policy to allow traffic to the pod on port 80 only. Test that you can now access the default nginx web page.

30. Find and use the `review6.yaml` file to create a pod.

```
student@cp:~$ kubectl create -f review6.yaml
```

31. View the status of the pod.

32. Use the following commands to figure out why the pod has issues.

```
student@cp:~$ kubectl get pod securityreview

student@cp:~$ kubectl describe pod securityreview

student@cp:~$ kubectl logs securityreview
```

33. After finding the errors, log into the container and find the proper id of the nginx user.

34. Edit the pod such that the `securityContext` is in place and allows the web server to read the proper configuration files.

35. Create a new `serviceAccount` called `securityaccount`.

36. Create a ClusterRole named `secrole` which only allows create, delete, and list of pods in all apiGroups.

37. Bind the new clusterRole to the new serviceAccount.

38. Locate the token of the `securityaccount`. Create a file called `/tmp/securitytoken`. Put only the value of `token:` is equal to, a long string that may start with eyJh and be several lines long. Careful that only that string exists in the file.

39. Remove any resources you have added during this review

40. Create a new pod called `webone`, running the `nginx` service. Expose port 80.

41. Create a new service named `webone-svc`. The service should be accessible from outside the cluster.

42. Update both the pod and the service with selectors so that traffic for to the service IP shows the web server content.

43. Change the type of the service such that it is only accessible from within the cluster. Test that exterior access no longer works, but access from within the node works.

44. Deploy another pod, called `webtwo`, this time running the `wlniao/website` image. Create another service, called `webtwo-svc` such that only requests from within the cluster work. Note the default page for each server is distinct.

45. Test DNS names and verify CoreDNS is properly functioning.

46. Install and configure an ingress controller such that requests for webone.com see the `nginx` default page, and requests for webtwo.org see the `wlniao/website` default page. It does not matter which ingress controller you use.

47. Remove any resources created in this review.

48. Install a new cluster using an recent, previous version of Kubernetes. Backup etcd, then properly upgrade the entire cluster.

49. Create a pod running `busybox` without the scheduler being consulted.

50. Continue to create objects, integrate them with other objects and troubleshoot until each domain item has been covered.