Kubernetes Check-List

(Do not edit or comment, make a copy of this doc and perform your work there)

- ☐ Overview
- ☐ Kubernetes Architecture
- ☐ Kubernetes API-Resources
    - ☐ Pods
        - ☐ Pod Lifecycle
    - ☐ Deployments
    - ☐ Daemonsets
    - ☐ Jobs and CronJobs
    - ☐ Replicasets
    - ☐ Stateful Sets
    - ☐ Config Maps
    - ☐ Service Accounts
    - ☐ Secrets
    - ☐ Service
        - ☐ ExternalName
        - ☐ ClusterIP
        - ☐ NodePort
        - ☐ LoadBalancer
    - ☐ Ingress
    - ☐ RBAC(Role, ClusterRole, RoleBinding, ClusterRoleBinding)
    - ☐ PV, PVCs and Storage Class
    - ☐ Scheduling, Preemption and Eviction

Tasks

1. Create a new deployment deploy-1 using image `busybox with 3 replicas and command sleep 3600.
2. Create a pod called `httpd using the image `httpd:alpine` in the secops namespace. Next, create a service of type ClusterIP by the same name (httpd). The target port for the service should be 80.

3. Create a pod `box0 with image `busybox and command ‘sleep 30; touch /tmp/debug; sleep 30; rm /tmp/debug’. The pod should not be ready until the file /tmp/debug is available in the pod. If the file doesn't exist, the pod must restart.

4. Create a pod nginx03 with image nginx, make sure the server is up and running every 10 seconds. The nginx server ideally takes 30 seconds to warm up so ensure that there are no false negatives when reporting the health of the pod.

5. Create a job named “hello-job” with the image “busybox” which echo’s “Hello I’m running job”.

6. Create a job that calculates pi to 2000 decimal points using the container with the image named perl and the following commands issued to the container: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)”]. Once the job has completed, check the logs to and export the result to pi-result.txt.

7. This is a multistage problem:

    a. Create a deployment deploy01 with image nginx with port 80 exposed.

    b. Update the deployment with image nginx:dev and make sure you record the execution of this action.

    c. If the deployment in step b is unsuccessful, rollback the deployment to the previous state by setting any fields explicitly.

    d. If the nginx deployment is in a healthy state, scale the deployment to run 3 replicas, also record this execution for audit-keeping.

    e. Display the history of deployment deploy-01 and store it in file /opt/answers/31.txt.

8. Create a pod that uses secrets

    a. Pull secrets from environment variables

    b. Pull secrets from a volume

    c. Dump the secrets out via kubectl to show it worked

9. Create a Pod with two containers, both with image busybox and command "echo hello; sleep 3600". Connect to the second container and run 'ls'

10. Create a pod with an nginx container exposed on port 80. Add a busybox init container which downloads a page using "wget -O /work-dir/index.html http://neverssl.com/online". Make a volume of type emptyDir and mount it in both containers. For the nginx container, mount it on "/usr/share/nginx/html" and for the init container, mount it on "/work-dir". When done, get the IP of the created pod and create a busybox pod and run "wget -O- IP"

11. Create a role the will allow users to get, watch, and list pods and container logs

12. Create a Pod

    a. name red using nginx image

    b. Update the pod red to use an initContainer that uses the busybox image and sleeps for 20 seconds

13. Create a cron job with an image busybox that runs every minute and writes 'date; echo Hello from the Kubernetes cluster' to standard output. The cron job should be terminated if it takes more than 17 seconds to start execution after its scheduled time (i.e. the job missed its scheduled time).

14. Create a temp busybox pod and connect via wget to foo service. Verify that each time there's a different hostname returned. Delete deployment and services to cleanup the cluster.

15. Liveness, Readiness and Startup Probes

    (i) Create an nginx pod with a liveness probe that just runs the command 'ls'. Save its YAML in pod.yaml. Run it, check its probe status, delete it.

    (ii) Modify the pod.yaml file so that the liveness probe starts kicking in after 5 seconds whereas the interval between probes would be 5 seconds. Run it, check the probe, delete it.

    (iii) Create an nginx pod (that includes port 80) with an HTTP readinessProbe on path '/' on port 80. Again, run it, check the readinessProbe, delete it.

16. Create a pod that has 3 containers in it? (Multi-Container)

17. Create a new Deployment named wordpress.

    I. Use container image wordpress:4.8-apache.
    II. Use deployment strategy Recreate.
    III. There should be 3 replicas created.
    IV. The pods should request 10m cpu and 64Mi memory.
    V. The livenessProbe should perform an HTTP GET request to the path /readme.html and port 80 every 5 seconds.

18. Configure PodAntiAffinity to ensure that the scheduler does not co-locate replicas on a single node.

(a) Pods of this deployment should be able to run on master nodes as well, create the proper toleration.

(b) Create a pod ubuntu03 with image ubuntu and add capabilities for SYS_TIME to each container. Run the command date -s '01JAN 2020 10:00:00 to verify if it is successful.

(c) Create a job that will roll a dice using image kodekloud/throw-dice. The image generates a random number between 1 and 6 and exits. It is a success only if it is a 6 else it is a failure. Check how many times it takes the job to trigger to get a successful outcome.

References for Preparation

- https://github.com/walidshaari/Kubernetes-Certified-Administrator
- https://medium.com/dzerolabs/just-in-time-kubernetes-a-beginners-guide-to-kubernetes-core-concepts-19ee7acbafa1
- https://medium.com/the-programmer/kubernetes-fundamentals-for-absolute-beginners-architecture-components-1f7cda8ea536
- https://medium.com/devops-mojo/kubernetes-objects-resources-overview-introduction-understanding-kubernetes-objects-24d7b47bb018
- https://medium.com/edureka/kubernetes-networking-a46d9f994bab
- https://medium.com/swlh/kubernetes-storage-explained-558e85596d0c
- https://medium.com/geekculture/storage-kubernetes-92eb3d027282
- https://loft.sh/blog/kubernetes-probes-startup-liveness-readiness/
- https://sysdig.com/blog/kubernetes-limits-requests/
- https://bluexp.netapp.com/blog/cvo-blg-kubernetes-storageclass-concepts-and-common-operations