Q.) You have deployed a microservers appin on K8s and one of you pod is frequently crashing. How do you troubleshoot this?

→ (1) check pod log :- using this you have to check what is causeof crash using commond " kubectl Logs <pod-name> "

(2) describe the pod :- In this you have to check events like crashes

" kubectl describe pod <pod-name> "

(3) check resource limit :- ~~To check~~ some time pod

(i) crashes due to insufficient memory (RAM) or CPU

kubectl logs <pod-name>
kubectl describe pod <podname>
kubectl top pod <podname>

(ii) use to check by kubectl top pod <podname>

(a) Kubernet deployment & How is it work?

** → (i) Deployment is an object that provides decelerative updates to appln running into the pods through ~~rip~~ replicaset (also object)

(ii) It ensure that desired no. of pods are running or not and also manages process of scaling, rolling updates & rollback.

a) Your appln needs to scaled up to handled higher traffic. How would you scale a deployment in K8S?

→ To scale deployment =>

(i) use kubectl scale deployment <deployment-name> --replicas = <number-of-replica>
**

to increase no. of pod.

: [kubectl scale deployment <deployment-name> . --replica = <number-of-replica>

(ii) You can also increase no. of pods inside node to distribute load by using "Horizontal pod autoscaling"

(iii) To handled traffic we can also incres memory or cpu by increse resources in pod by performing "vertical pod autoscaling"

Scanned with ACE Scanner

(iv) Also incres nodes in cluster by performing cluster Autoscaling.

Q) What would you do if pod is stuck in "crashloop Backoff" state?

→ * Senario = pod repetadly crash & restart

("Crashloop Backoff")

(i) 1st we have to check logs of pod i.e 1st check what happens inside pod.

By, "kubectl logs <pod-name>"

(ii) Then check what cause ~~crash~~ of pod crashing by kubectl describe pod <pod-name>

(iii) ~~Then~~ Then you have to increas the memory or cpu becoz this is main cause to crash pod & increas memory by increases resources in pod using vertical pod autoscaling.

(iv) Fix the issue By providing correct environment variables, mounting volm or updating container image

(Q) How do you ensure high availability for an appln in kubernetes?

→ senario => you need to avoid downtime.

Scanned with ACE Scanner

(i) use multiple replicas by deployment in YAML file.

(ii) using node-affinity ~~for~~ k8s feature deploy or pod Anti-affinity pods in multiple node.

(iii) ~~user~~ use readiness probe to check healthy pod receive traffic / handeld traffic or not.

→ using this method you can check your app in pod is ready to handler user requests

a) what should we do if ~~no~~ one node in cluster goes down?

→ (*) ~~no~~ senario = node hosting app in pods crashes.

~~(i) 1st k8s~~

(i) 1st k8s detect the failure & cause of failure and reshedule pods on other healthy nodes (if enough resources or CPU or memory available)

(ii) To handel this problem I will do,

(a) monitoring ~~to~~ of cluster using tools like Grafana and ~~prometheus~~ prometheus

(b) By using pod anti-affinity / node-affinity rule you can deploy pods in node

* command use deploy pod => (i) kubectl apply -f <pod.yaml>
then verify pod status => (ii) kubectl get pods

Scanned with ACE Scanner

I.M.P

* deploy pod ~~us~~ without YAML file

- Scenario ⇒ How can you deploy pod without YAML file

~~kubectl run my-pod~~

kubectl run my-pod --image=nginx

*

NGINX :- (i) It is service of linux if any request comes from browser by HTTP/HTTPS then it shows their webpage to user.

(ii) It is used in many cases like load balancer, reverse proxy or mail proxy.

(iii) Users - Netflix.

Q) How would you deploy specific version of an appln?

→ ~~Sce~~ Scenario :- deploy version 1.2.0 of an appln.

*

(i) 1st update image in .yaml file

[image option present in container
name : my-app-container
image : my-app : 1.2.0 (version name)]

save this change file in pod.yaml

then apply in k8s cluster using

* kubectl apply -f <pod.yaml>

Then check our update by

kubectl get pods

Scanned with ACE Scanner

* Debugging :-

(1) check pod status :- kubectl get pod

check status column :- like pending, running, crashLoopBackOff or error

(2) Then describe pod to check reason of crash By kubectl describe <pod.name> pod

Scanned with ACE Scanner

(3) Then ~~cto~~ check logs - to check what ~~internal~~ happend actually inside pod..

⇒ "kubectl logs < pod-name>"

(4) for multicontainer to check logs. by

* ⇒ kubectl Logs <pod-name> -c <container-name>

(4) Access pod's shell in container :- Used for deeper investisation of pod ~~ea~~ you can execute ~~pods~~ shell inside container for deeper investisation

⇒ kubectl exec -it <pod-name> -- /bin/sh

& for multicontainer :-

⇒ kubectl exec -it <pod-name> -c <container-name> -- /bin/sh

(5) check event in name space :-

⇒ kubectl get events --sort-by = '.metadata.creationtimestamp'

(5) ~~ctd~~ Debug using ephemeral / tempory debug container :-

⇒ Pod is running but not accessible then you can used temporary debug container:

⇒ kubectl debug pod/ <pod-name> -it --image = busybox

Scanned with ACE Scanner

kubectl rollout undo deployment/object.name --to-version = version no.

(7) Inspect pod yaml :-

kubectl ~~pod~~ get pod <pod-name> -o yaml

(8) Delete and restart pod :-

kubectl delete pod <pod-name>

Q) How do you troubleshoot a k8s cluster when app is running slow?

→ (1) 1st check resource usage :-

By, kubectl top pod <pod-name>

(2) check Logs of pod :-

kubectl Logs <pod-name>

(3) check events :-

kubectl get events

(4) Find check status of pod & check ~~connectivity~~ connectivity betn pod & ~~container~~ Services.

~~kubectl~~

kubectl get pod.

Q) How do you perform rollback in k8s? ⇒ if you want to get updated from privious pod

→ ~~using cmd~~ you can go to privious version by go back to old version

⇒ kubectl rollout undo deployment/object name --to-version = version no.

⇒ Also check history kubctl ~~history~~ rollout history deployment/ <deployment-name>