Kubernetes Pod creation and termination process

A Pod is a Kubernetes abstraction that represents a group of one or more application containers (such as Docker), and some shared resources for those containers like volumes,

The magic of Kubernetes starts with creating these pods after applying manifests.

Pod creation

The pod creation process requires multiple components to work together to bring the pods to ready state. Following are the different stages while creating the pod

1. **Scheduling** – This is the first step where K8s scheduler looks on all the nodes and picks the best node to run the pod. Eliminates irrelevant nodes like not enough capacity, Taint nodes, master nodes…
   From the relevant nodes, picks the best node with enough requested limits.
   When the scheduler is unable to place the pod in a suitable node, it goes to **pending** state while a failed scheduling event is issued to the pod. Reasons can be Insufficient CPU or memory, Node selector or taints, pod affinity, persistent volume unavailable, nodes are cordoned for updates or restart etc. This is when the auto scaler spins a new node.
2. **Infra creation** – After finding and scheduling the pod on the suitable node, then the kubelet in the node runs the pod by creating infra needed for the pod. Ex: reserving CPU and memory resources needed, attaching volumes, config maps etc
3. **Container creation –** Node to run the pod and Infra setup is done on the node. Kubelet can start to run the containers by pulling the images for the init containers and application containers.
   In this stage we can see few errors like –
   Create config error – referencing config map or secret or persistent volume that does not exist
   Image pull error – container unable to pull the image form a container registry
4. **Container start** – Containers are running but not ready to take requests
5. **Ready** – Traffic will start flow into the pod

Pod Termination

Pods are ephemeral units of work that designed to be terminated. Common triggers for pod termination can be scaling, deployment rollout, forced termination by node drain. In Kubernetes, pods termination process can be Graceful or Ungraceful.

In graceful shutdown, use kubectl to manually delete a pod. Default graceful termination time is 30sec for pod to shutdown itself after that it will be ungraceful termination. We have 2 ways to handle termination to save any work or not to lose any data.

1. Linux signal SIGTERM – write code to catch the SIGTERM signal
2. Prestop hook – This hook is called immediately before a container is terminated. Execute a http request or script before pod termination using prestop hooks.