30 Days of Kubernetes challenge :-

#Day5 :-

- Namespaces
- Ingress
- Jobs & cronjobs
- Daemonsets
- statefulsets.
- ~/.kube/config - file

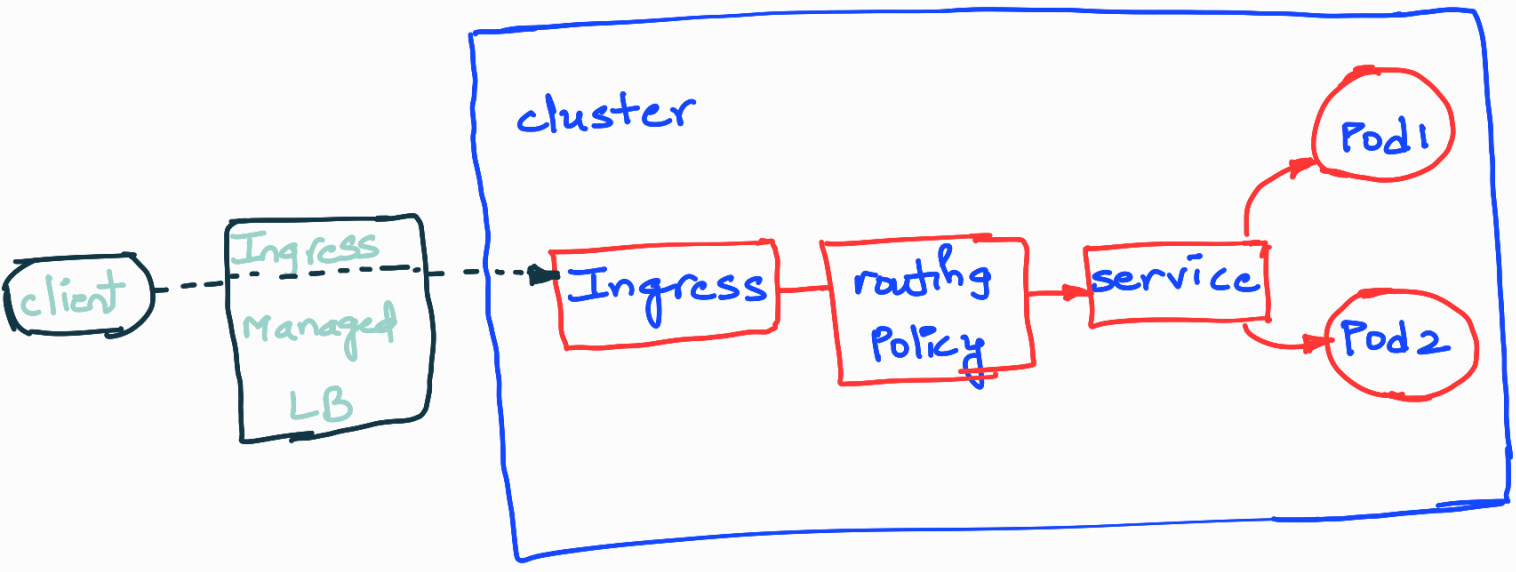
1) Namespaces :-

- Namespaces provides mechanism for isolating group of resources within a cluster.
- Name of resource in NS should be unique.
- Deployments, services etc are Namespaced scoping objects.
  1. default
  2. Kube-node-lease
  3. Kube-public
  4. Kube-system

2) Ingress :-

- An API Object that manages external access to the services in a cluster typically HTTP.

- Ingress may provide load balancing, ssl termination and name based virtual hosting.



3) Jobs & Cron jobs :-

- Job creates one/more pods & will continue to retry execution of the pods until a specified no. of them successfully terminate

- A cronjob creates Jobs on a repeating schedule. Just like crontab in UNIX/Linux.

4) Daemonset :-

- A Daemonset ensures that all/some Nodes runs a copy of Pod. As Nodes added to cluster, Pods are added to them. As Nodes are removed from cluster, those pods are garbage collected. Deleting Daemonset will clean up the pods its created.

- Uses of daemonset are... running

① cluster storage daemon on every Node

② logs collection daemon on every Node

③ Node monitoring daemon on every Node

5) statefulsets :-

- Statefulset is the workload API used to manage stateful applications.

- Manages the deployment and scaling set of Pods and provides guarantees about ordering and uniqueness of these pods.

6) ~/.kube/config file :-

- Kubernetes configuration file called as kubeconfig

- It is a YAML file that specifies various settings and parameters for interacting with Kubernetes cluster.

- This file is used by kubectl cmd to authenticate with cluster, specifies cluster and configure other options.

- key Elements in Kubeconfig file are

① cluster configwation

② User configwation

③ Context configwation

④ current context

checkout Example "kubeconfig" below

yaml

```yaml
apiVersion: v1
kind: Config
clusters:
- name: my-cluster
  cluster:
    server: https://cluster-api-server
    certificate-authority-data: <certificate-authority-data>
users:
- name: my-user
  user:
    client-certificate-data: <client-certificate-data>
    client-key-data: <client-key-data>
contexts:
- name: my-context
  context:
    cluster: my-cluster
    user: my-user
current-context: my-context
```