KPLABS Course

Certified Kubernetes Administrator

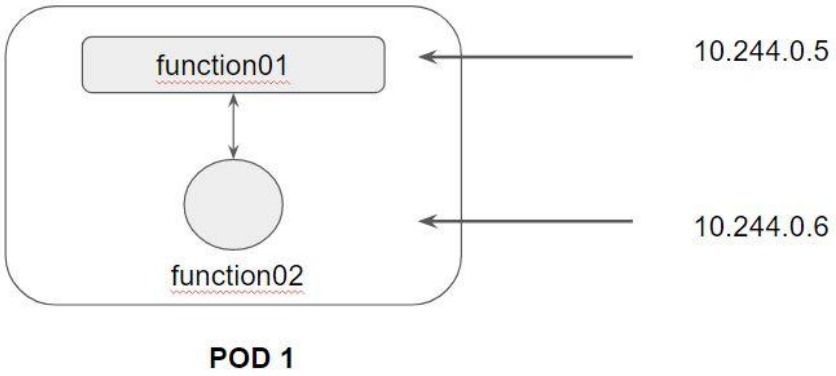
Services and Networking

ISSUED BY
Zeal Vora

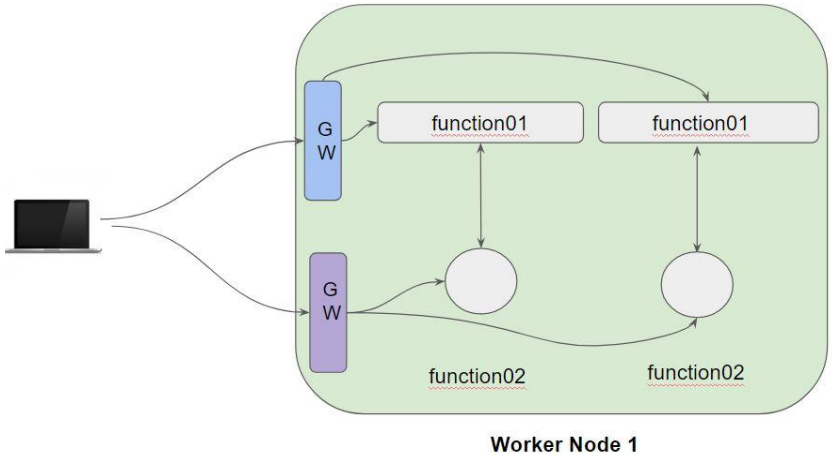
REPRESENTATIVE
instructors@kplabs.in

Module 1: Overview of Service

Whenever you create a Pod, the containers created will have Private IP addresses.



Following is a high-level diagram on the functionality of Service:



In a Kubernetes cluster, each Pod has an internal IP address.

Pods are generally ephemeral, they can come and go anytime.

We can make use of service which acts as a gateway and can get us connected with right set of pods.

Service is an abstract way of exposing application running in the pods as a network service.

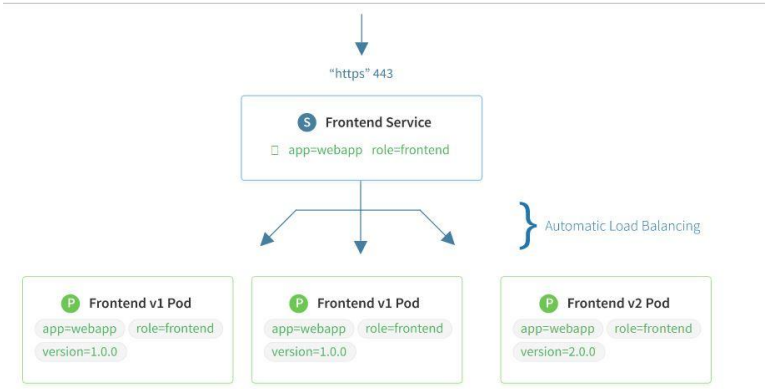
There are several types of Kubernetes Services which are available:

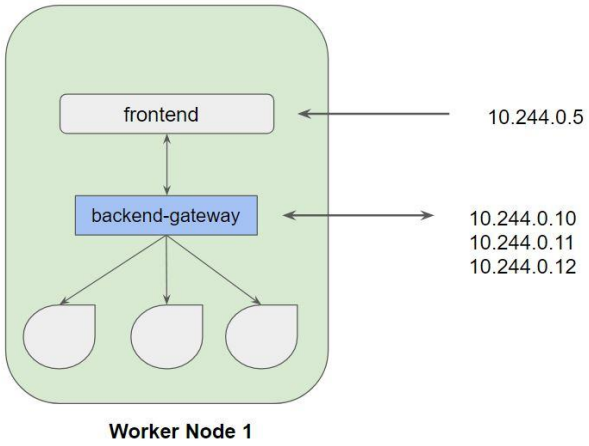
- NodePort
- ClusterIP
- LoadBalancer
- ExternalName

Module 2: Creating our first Service and Endpoint

Kubernetes Service can act as an abstraction which can provide a single IP address and DNS through which pods can be accessed.

Endpoints track the IP address of the objects that service can send traffic to.





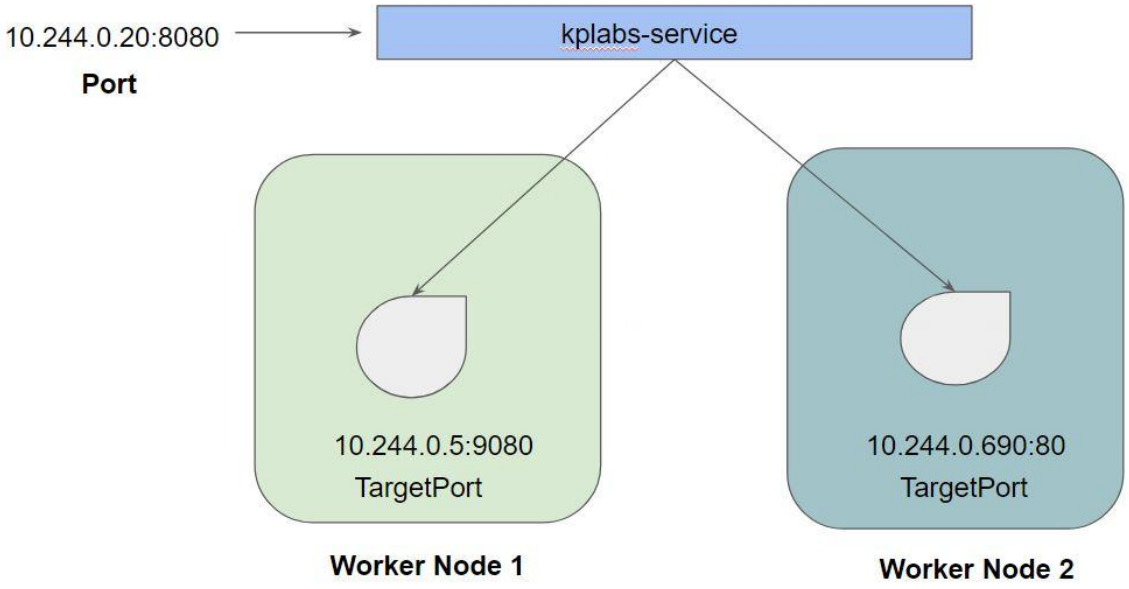
Module 3: Service Type - ClusterIP

Whenever the service type is ClusterIP, an internal cluster IP address is assigned to the service.

Since an internal cluster IP is assigned, it can only be reachable from within the cluster.

This is a default ServiceType.

Module 4: Port vs TargePort

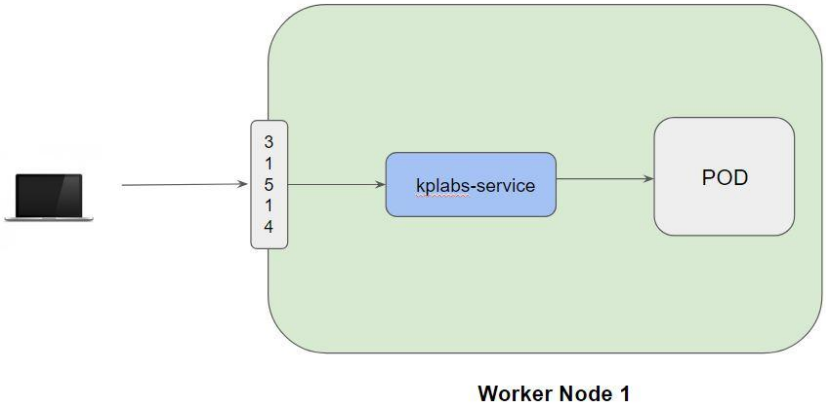


Module 5: Service Type - NodePort

From the name, we can identify that it has to do with opening a port on the nodes.

If the service type is NodePort, then Kubernetes will allocate a port (default: 30000-32767) on every worker node.

Each node will proxy that port into your service.



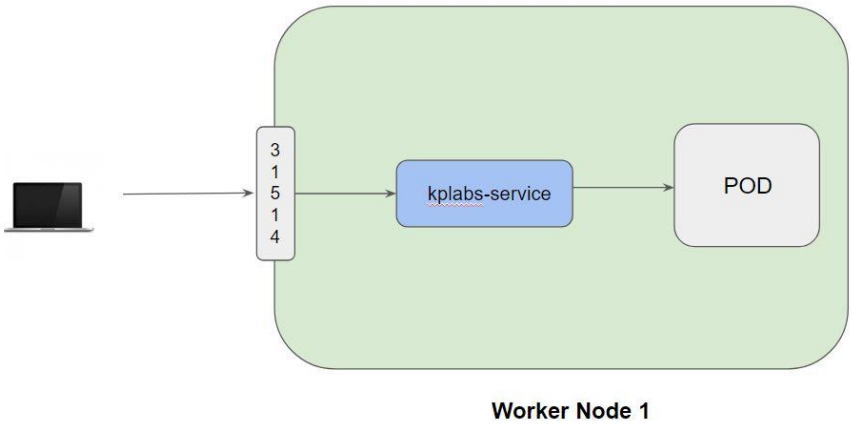
Module 6: Service Type - LoadBalancer

6.1 Challenges with NodePort

We know that NodePort ServiceType will assign a node in all the worker node which can forward the traffic to the underlying service.

Challenge in NodePort: We need to access it via IP/DNS:Port

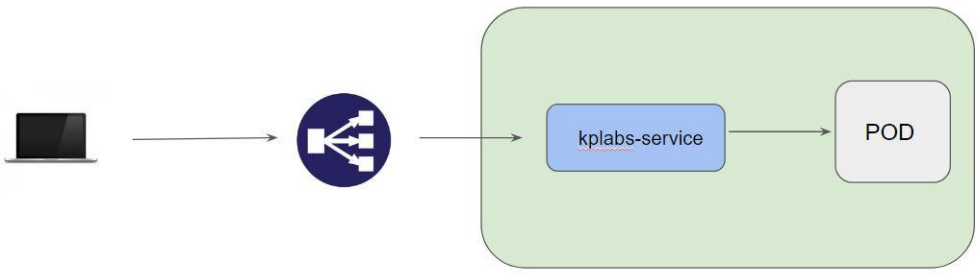
Example: google.com:31514



6.2 Understanding LoadBalancer Service Type

LoadBalancer Service Type will automatically deploy an external load balancer.

This load balancer takes care of routing requests to the underlying service.



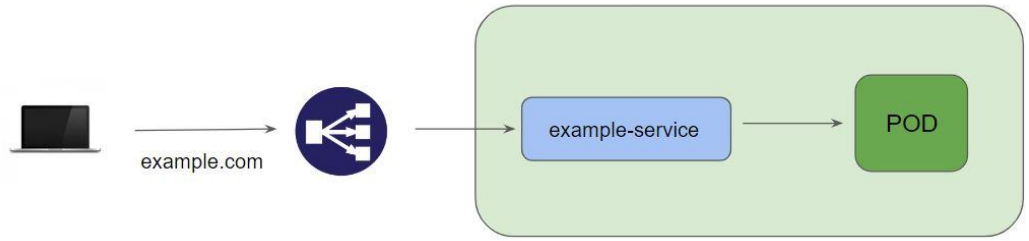
6.3 Important Pointers

The overall implementation of LoadBalancer depends on your Cloud Provider.

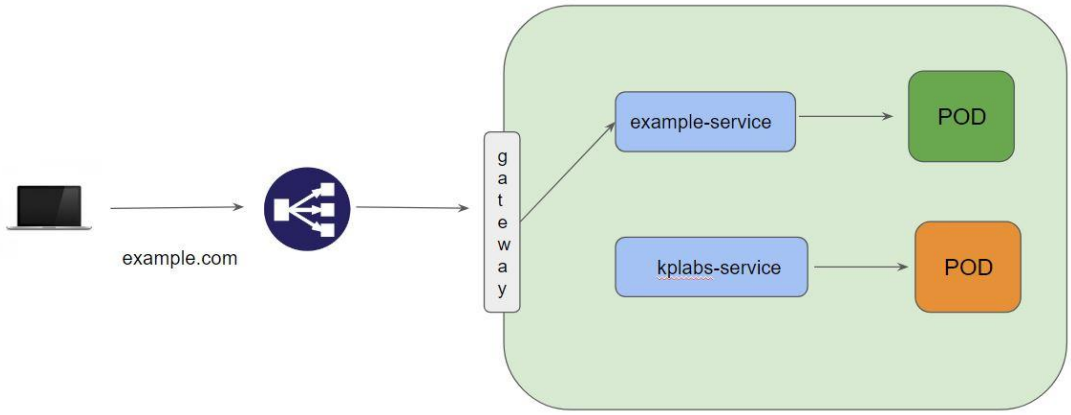
If you plan to use it in bare-metal, then you will have to provide your own load balancer implementation.

Module 7: Ingress

Whenever making use of the LoadBalancer Service Type, out of the box, you can make use of a single website.



With ingress, we can set up multiple rules through which traffic can be routed to an appropriate service depending on the URL of the website that is requested.



Kubernetes Ingress is a collection of routing rules which governs how external users access the services running within the Kubernetes cluster.

Ingress can provides various features which includes:

- Load Balancing
- SSL Termination
- Named-based virtual hosting

Module 8: Ingress Resource and Ingress Controllers

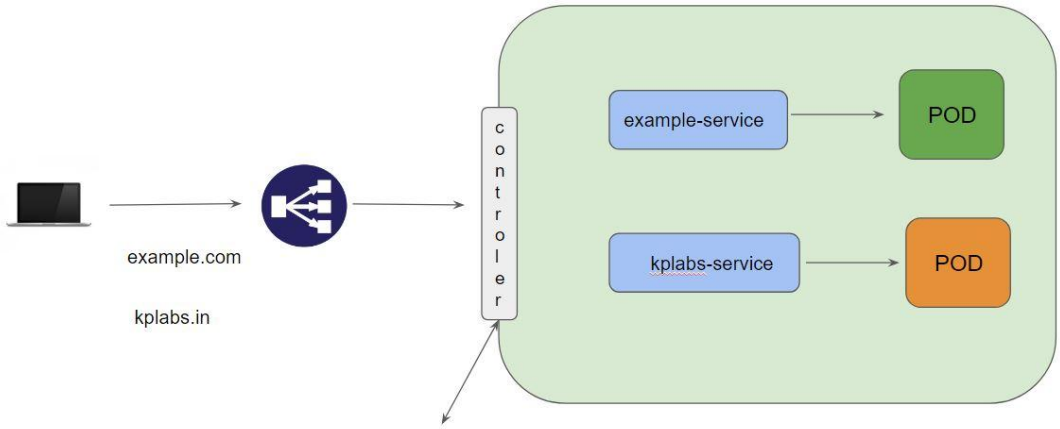
There are two sub-components when we discuss about Ingress:

- Ingress Resource
- Ingress Controllers

Ingress Resource contains set of routing rules based on which traffic is routed to a service.

Ingress Controller takes care of the Layer 7 proxy to implement ingress rules.

You must have an ingress controller to satisfy an Ingress. Only creating an Ingress resource has no effect



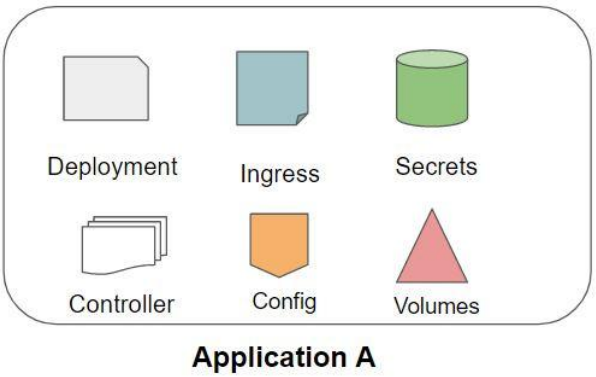
Rule	Host	Service
1	example.com	example-service
2	kplabs.in	kplabs-service

Module 9: Helm

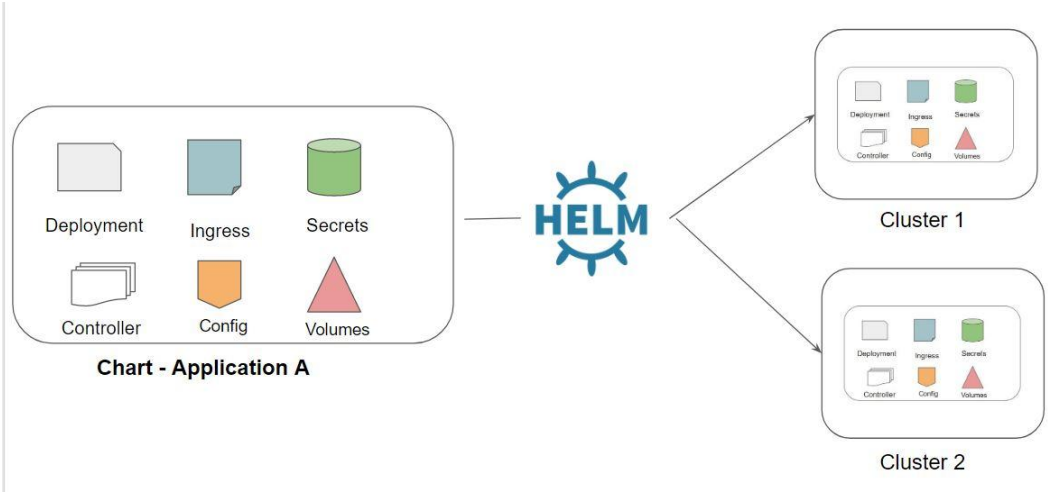
Helm is one of the package manager for Kubernetes.

Kubernetes application can contain lot of lot of objects like:

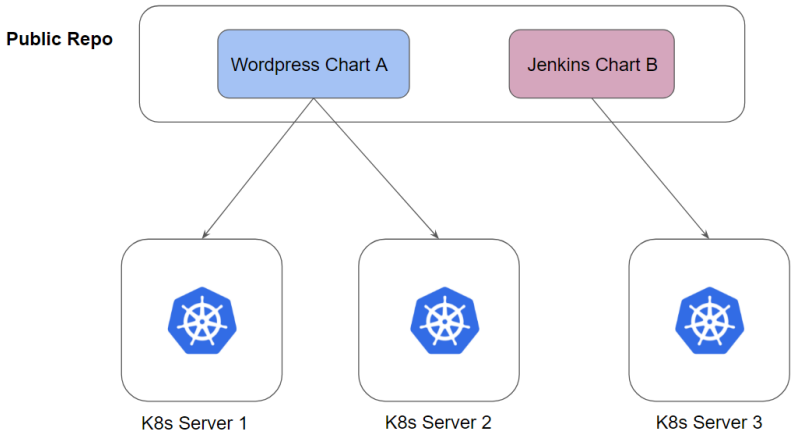
Deployments, Secrets, LoadBalancers, Volumes, services, ingress and others.



Following diagram depicts Helm Charts



All the Helm charts are stored in a central repository through which charts can be pulled and deployd.

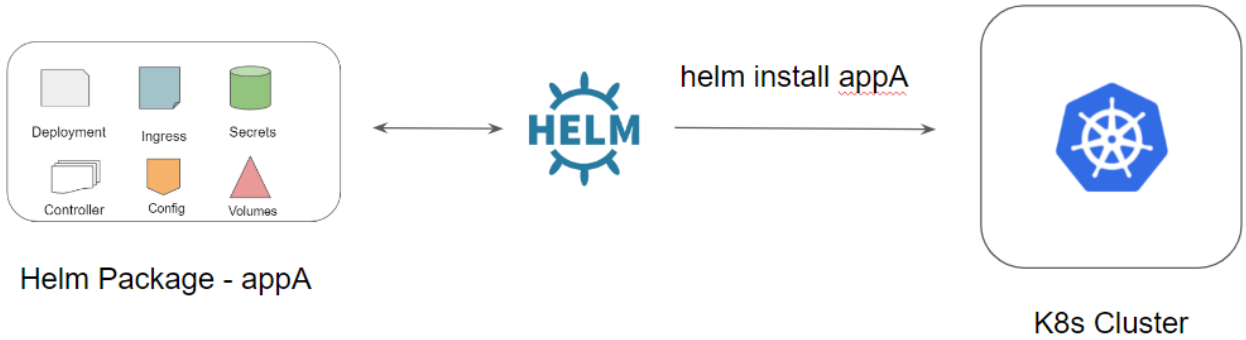


Before we conclude, let us revise the concepts:

- A Chart is a Helm package.
- It contains all of the resource definitions necessary to run an application, tool, or service inside of a Kubernetes cluster.
- A Repository is the place where charts can be collected and shared

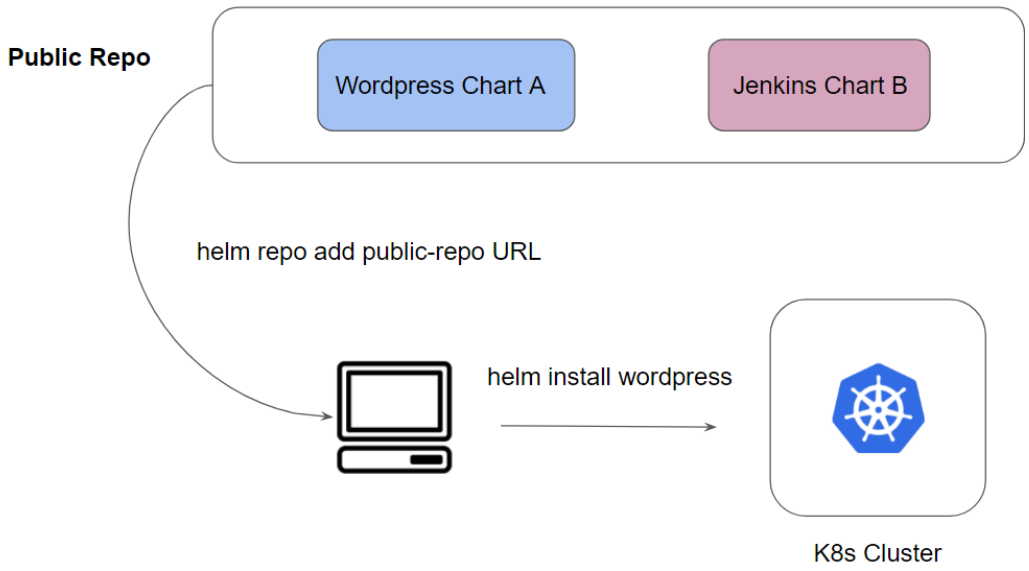
Module 10: Deploying Helm Charts

To install a new package, use the **helm install** command



Helm packages can be placed in local disk or can also be stored centrally in public or private repositories.

Depending on the location of the package, there is a slight change in the installation step.



Important Note:

- Every application packaged in Helm has its own set of requirements. Make sure to read the instructions carefully.
- Install Helm Package only from trusted repositories.

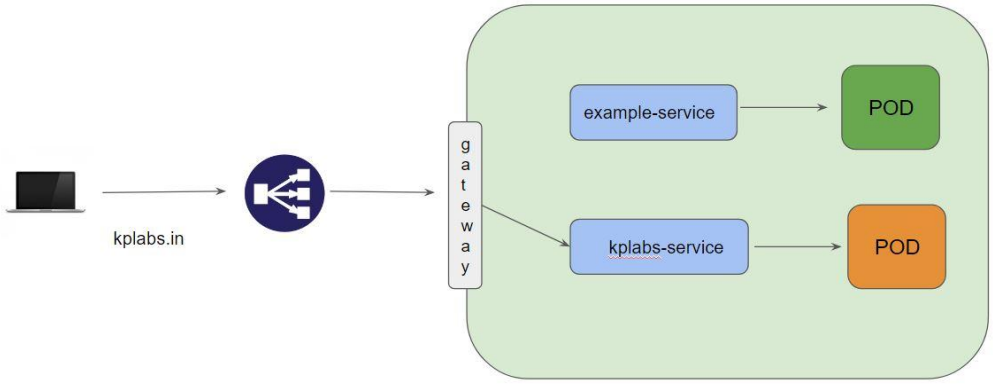
Following table describes all the basic Helm commands:

Commands	Description
helm repo add	Adds a chart repository.
helm install <chart>	Installs Helm Package
helm uninstall <release>	Uninstalls the release.
helm repo list	List repositories
helm list	List the releases

Module 11: Name Based Virtual Hosting

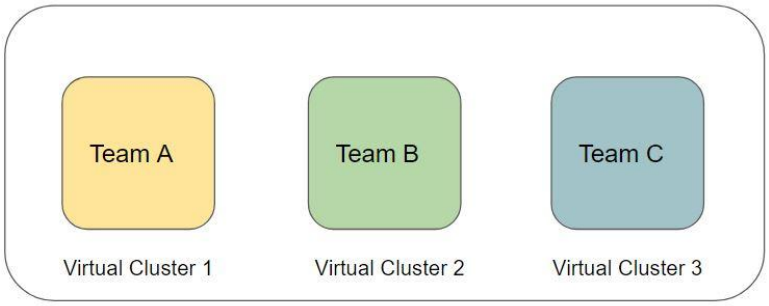
Name-based virtual hosts support routing HTTP traffic to multiple host names at the same IP address.

The requests are routed based on the HTTP host-header.



Module 12: Namespace

Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces.



Following is the list of namespaces that are available in Kubernetes:

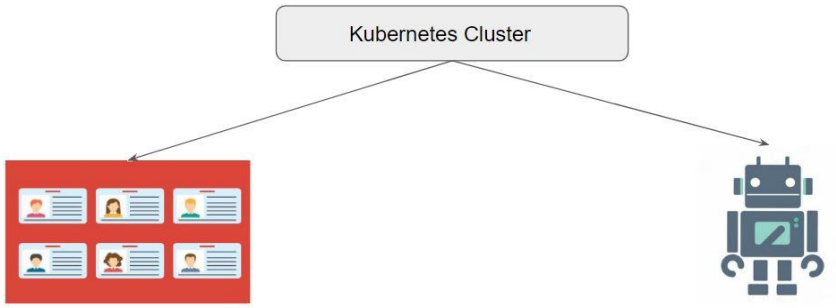
Namespace	Description
default	The default namespace for objects with no other namespace
kube-system	The namespace for objects created by the Kubernetes system
kube-public	This namespace is created automatically and is readable by all users. It contains information, like CA, that helps kubeadm join and authenticate worker nodes.
kube-node-release	The kube-node-lease namespace contains lease objects that are used by kubelet to determine node health.

Module 13: Service Accounts

13.1 Understanding Authentication

Kubernetes Clusters have two categories of users:

- Normal Users.
- Service Accounts

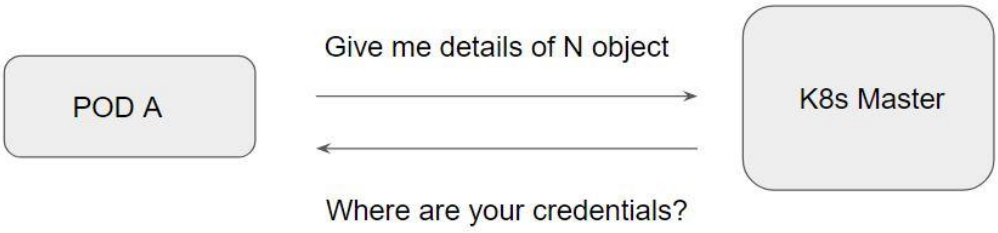


13.2 Understanding Service Accounts

Service Accounts allows the Pods to communicate with the API Server

Let's understand with a use-case:

Application within Pod A wants to retrieve an object within your K8S cluster.



Following diagram indicates the working of service account:

13.3 Important Pointer

Service Accounts are namespaced.

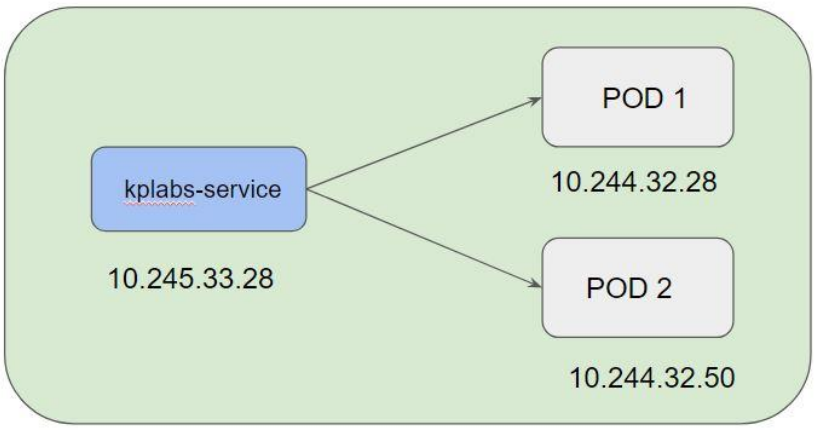
Default service account gets automatically created when you create a namespace

PODS are automatically mounted with the default service accounts.

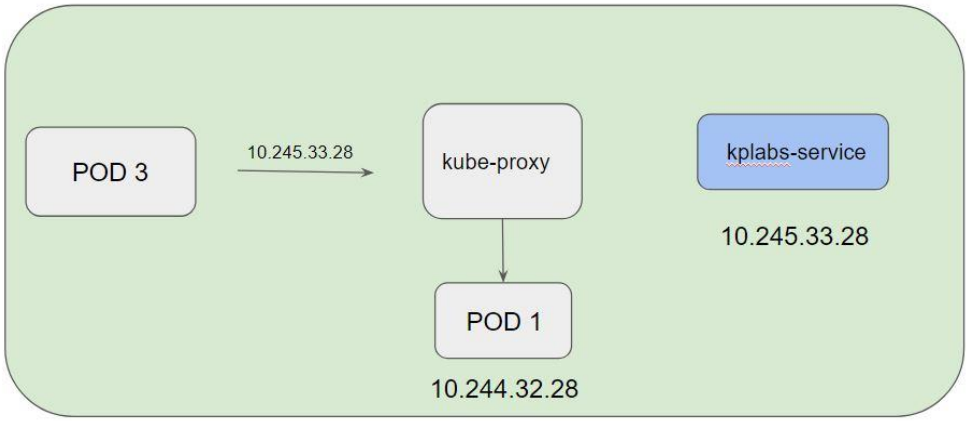
Module 14: kube-proxy

kube-proxy is primarily responsible for forwarding request from ClusterIP to Pod IP.

kube-proxy also takes care of implementing a form of virtual IP for Services



When there is a request for 10.245.33.28, this request is intercepted and redirected to one of the PODS associated with the service.


POD 3
10.245.33.28
kube-proxy
kplabs-service
10.245.33.28
POD 1
10.244.32.28