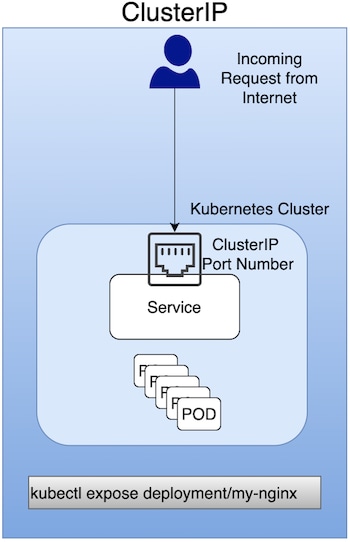
Ingress vs. ClusterIP vs. NodePort vs. LoadBalancer

Ingress, ClusterIP, NodePort, and LoadBalancer are all ways to get external traffic into your cluster, and they each do it differently. Let’s take a look at how each works and where you would use them.

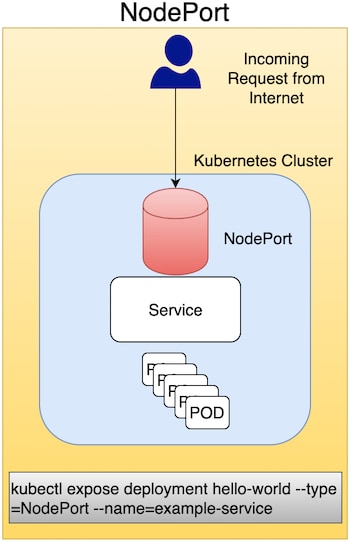
ClusterIP

ClusterIP is the preferred option for internal service access and uses an internal IP address to access the service. Some examples of where ClusterIP might be the best option include service debugging during development and testing, internal traffic, and dashboards.



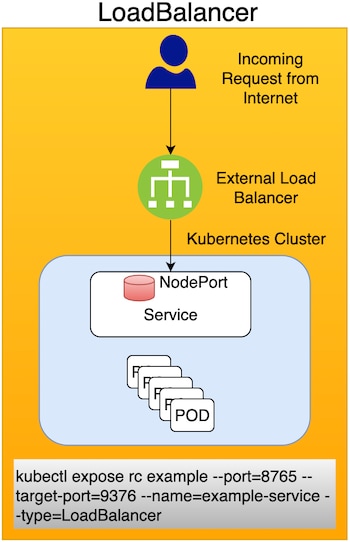
NodePort

A NodePort is a [virtual machine (VM](https://www.ibm.com/cloud/learn/virtual-machines)) used to expose a service on a Static Port number. It's primarily used for exposing services in a non-production environment (in fact, production use is not recommended). As an example, a NodePort would be used to expose a single service (with no load-balancing requirements for multiple services).



LoadBalancer

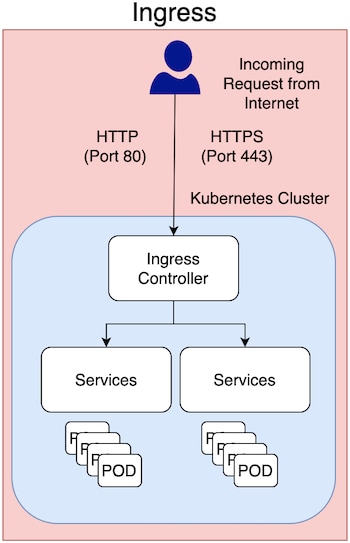
This method uses an external LoadBalancer to expose services to the Internet. You can use LoadBalancer in a production environment, but Ingress is often preferred.



Ingress

Ingress enables you to consolidate the traffic-routing rules into a single resource and runs as part of a Kubernetes cluster. Some reasons Kubernetes Ingress is the preferred option for exposing a service in a production environment include the following:

* Traffic routing is controlled by rules defined on the Ingress Resource.
* Ingress is part of the Kubernetes cluster and runs as pods.
* An external Load Balancer is expensive, and you need to manage this outside the Kubernetes cluster. Kubernetes Ingress is managed from inside the cluster.



In production environments, you typically use Ingress to expose applications to the Internet. An application is accessed from the Internet via Port 80 (HTTP) or Port 443 (HTTPS), and Ingress is an object that allows access to your Kubernetes services from outside the Kubernetes cluster.

Summary

The Kubernetes Ingress API lets you expose your applications deployed in a Kubernetes cluster to the Internet with routing rules into a single source. To implement Ingress, you need to configure an Ingress Controller in your cluster—it is responsible for processing Ingress Resource information and allowing traffic based on the Ingress Rules. It's important to choose the right service with appropriate configuration to expose your application to the Internet based on the guidelines listed above.