K21Academy
Learn Cloud from Experts

ORACLE Gold Partner | Silver Microsoft Partner | DevOps INSTITUTE REGISTERED EDUCATION PARTNER

Docker & Certified Kubernetes Administrator (CKA)

Concepts & Architecture | Storage | Networking | HA & Clusters | Scheduling
Administration | Docker Compose | Security | Troubleshooting

16 Module | 60+ Lessons | 33 Hands-On Labs | Exam Preparation | On-Job Support

docker + kubernetes

docker

+

kubernetes

K21Academy
Learn Cloud from Experts

Kubernetes Service

Module Agenda

- Need of a Service in Kubernetes
- Service Resources
- Service Definition
- Service types

Need of a Service in Kubernetes

K21Academy
Learn Cloud from Experts

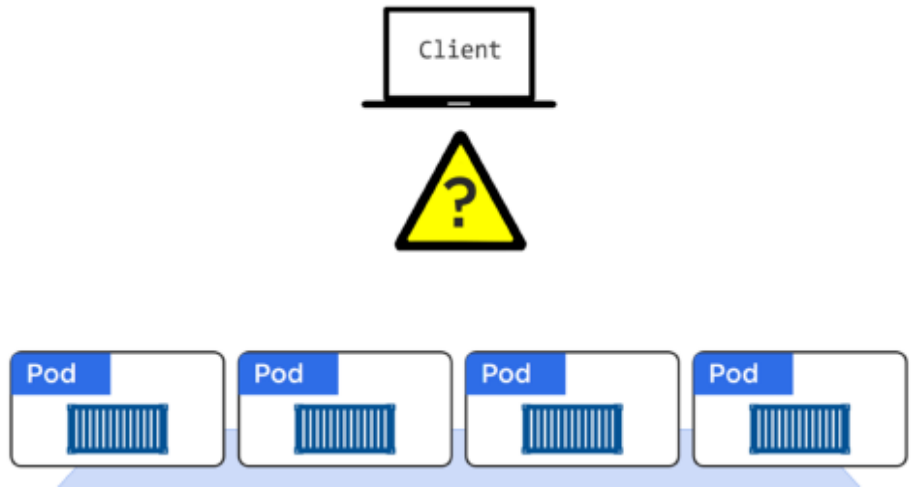
- Every Pod in a Kubernetes gets its own IP address.
- Pods are ephemeral resources.When Pods fail, they get replaced with new Pods that have new IPs.
- Scaling-up a Deployment introduces new Pods with new IP addresses. Scaling-down a Deployment removes Pods.
- This creates a large amount of IP churn, and creates the situation where Pod Ips cannot be relied on.
- Kubernetes Pods are created and destroyed to match the state of the cluster.

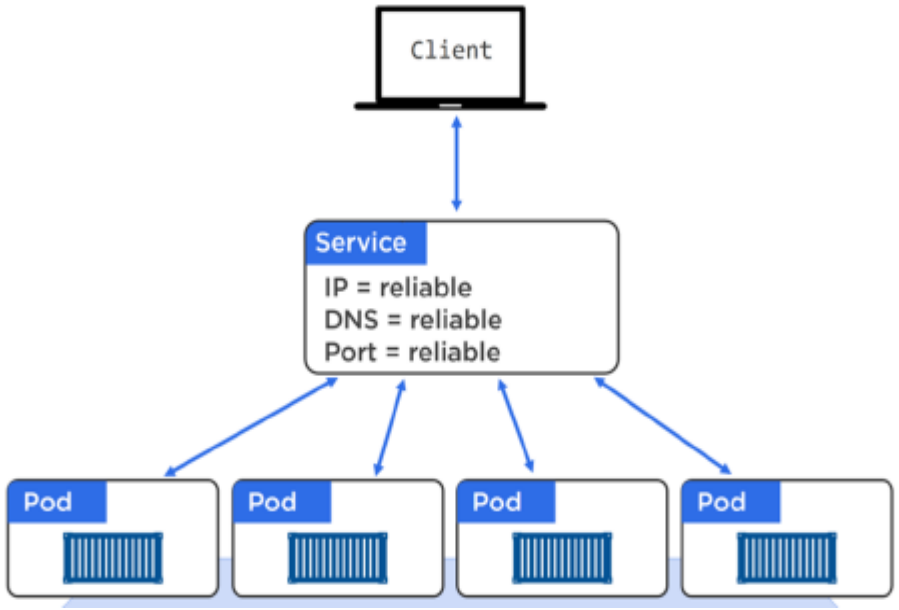
Service Resources

K21Academy
Learn Cloud from Experts

- Service provides a single IP address and port for a set of Pods and can load-balance across them.
- A Service is an abstraction which defines a logical set of Pods and a policy by which to access them.
- The set of Pods targeted by a Service is usually determined by a selector.

Service





Defining a Service

- A Service in Kubernetes is a REST object, similar to a Pod.
- The name of a Service object must be a valid.
- The default protocol for Services is TCP; you can also use any other supported protocol.
- Example:
  - Suppose you have a set of Pods that each listen on TCP port 9376 and carry a label app=MyApp:

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

Services and Endpoint objects

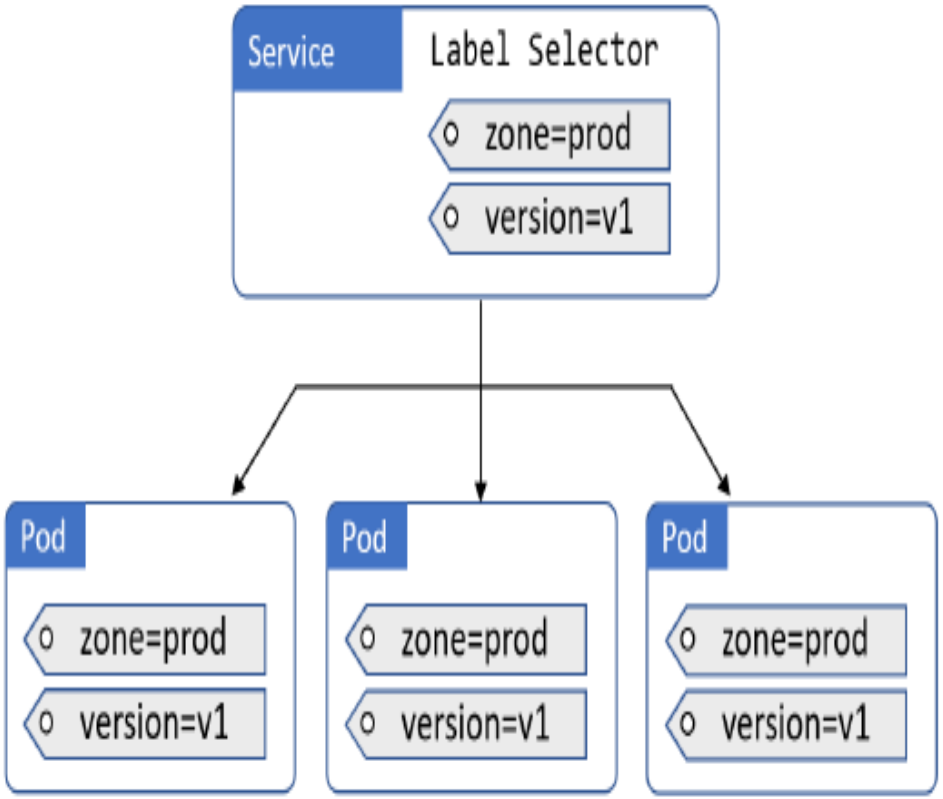
K21Academy
Learn Cloud from Experts

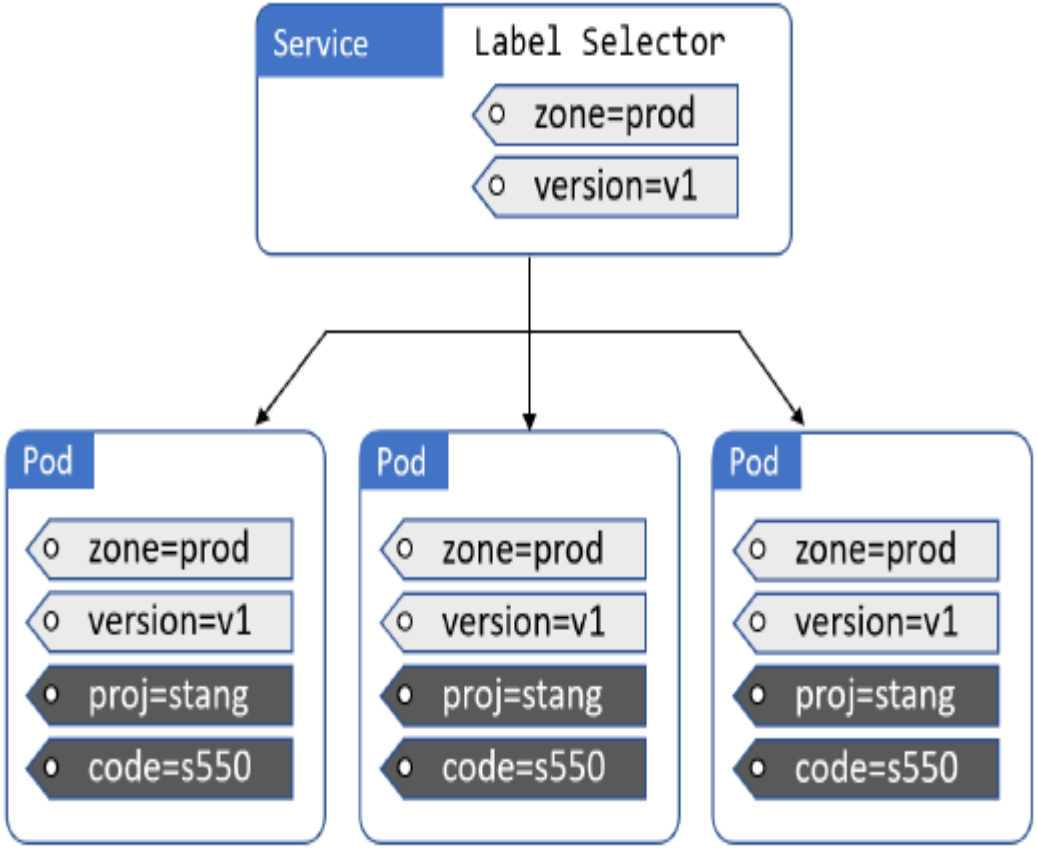
- As Pods come-and-go (scaling up and down, failures, rolling updates etc.), the Service dynamically updates its list of healthy matching Pods.
- It does this through a combination of the label selector and a construct called an Endpoints object.
- Each Service that is created, automatically gets an associated Endpoints object.
- All this Endpoints object is, is a dynamic list of all of the healthy Pods on the cluster that match the Service's label selector.

Labels

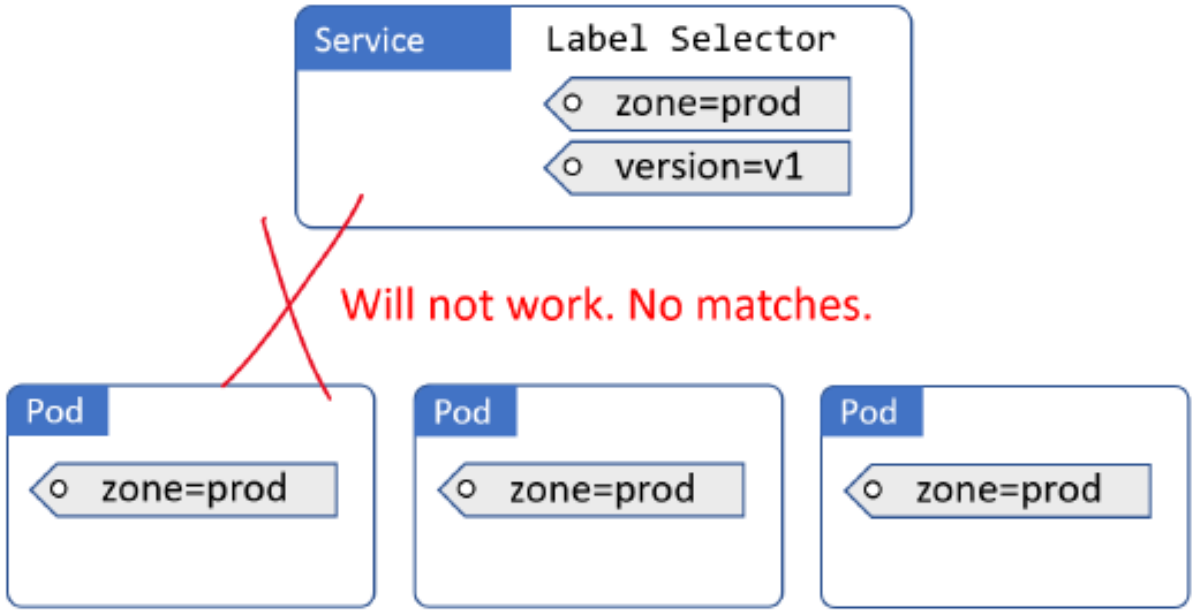
- Labels are simple key/value pairs
- Can be attached to objects (like pods or nodes)
- Via a label selector a client can select a set of objects
- Let you categorize objects

Service





Service..



Types of Services

K21Academy
Learn Cloud from Experts

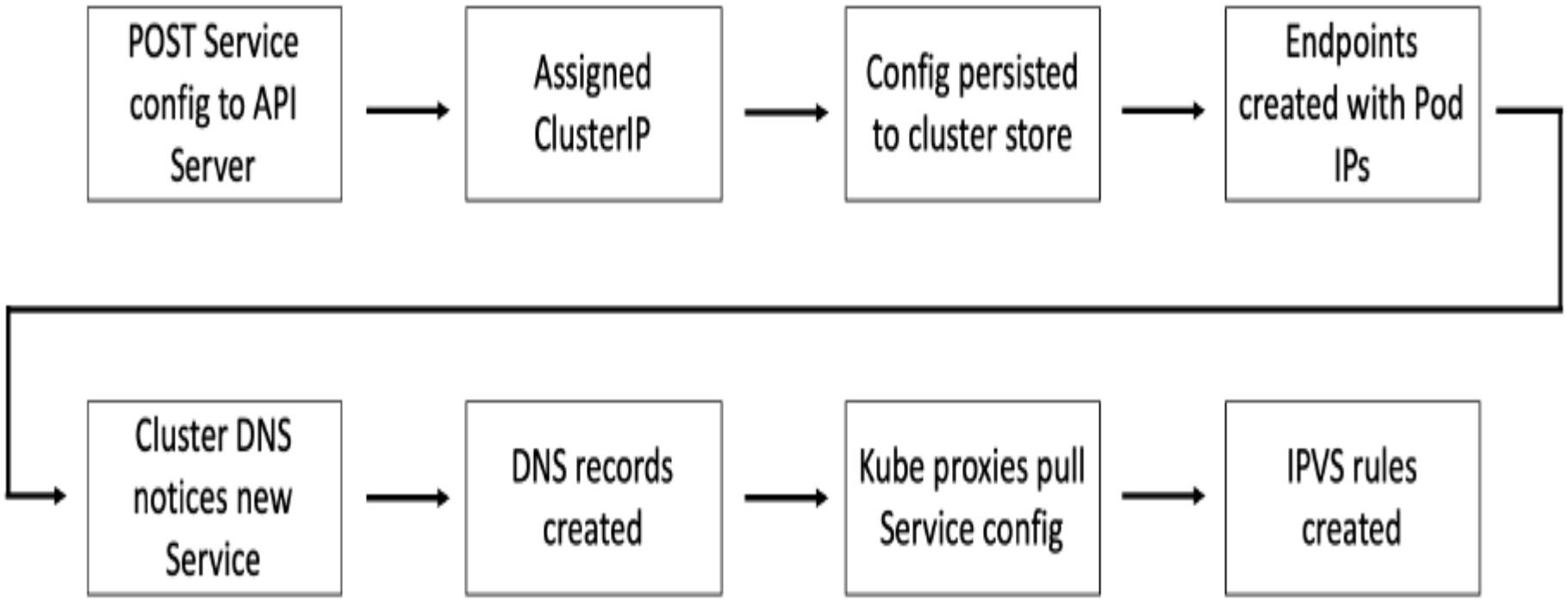
- The three common ServiceTypes are:
  - **ClusterIP:** This is the default option and gives the Service a stable IP address internally within the cluster. It will not make the Service available outside of the cluster.
  - **NodePort:** This builds on top of ClusterIP and adds a cluster-wide TCP or UDP port. It makes the Service available outside of the cluster on a stable port.
  - **LoadBalancer:** This builds on top of NodePort and integrates with cloud-based load-balancers.

Clusterip

K21Academy
Learn Cloud from Experts

- A ClusterIP Service has a stable IP address and port that is only accessible from inside the cluster.
- The ClusterIP gets registered against the name of the Service on the cluster's internal DNS service.
- All Pods in the cluster are pre-programmed to know about the cluster's DNS service, meaning all Pods are able to resolve Service names.

Service Flow



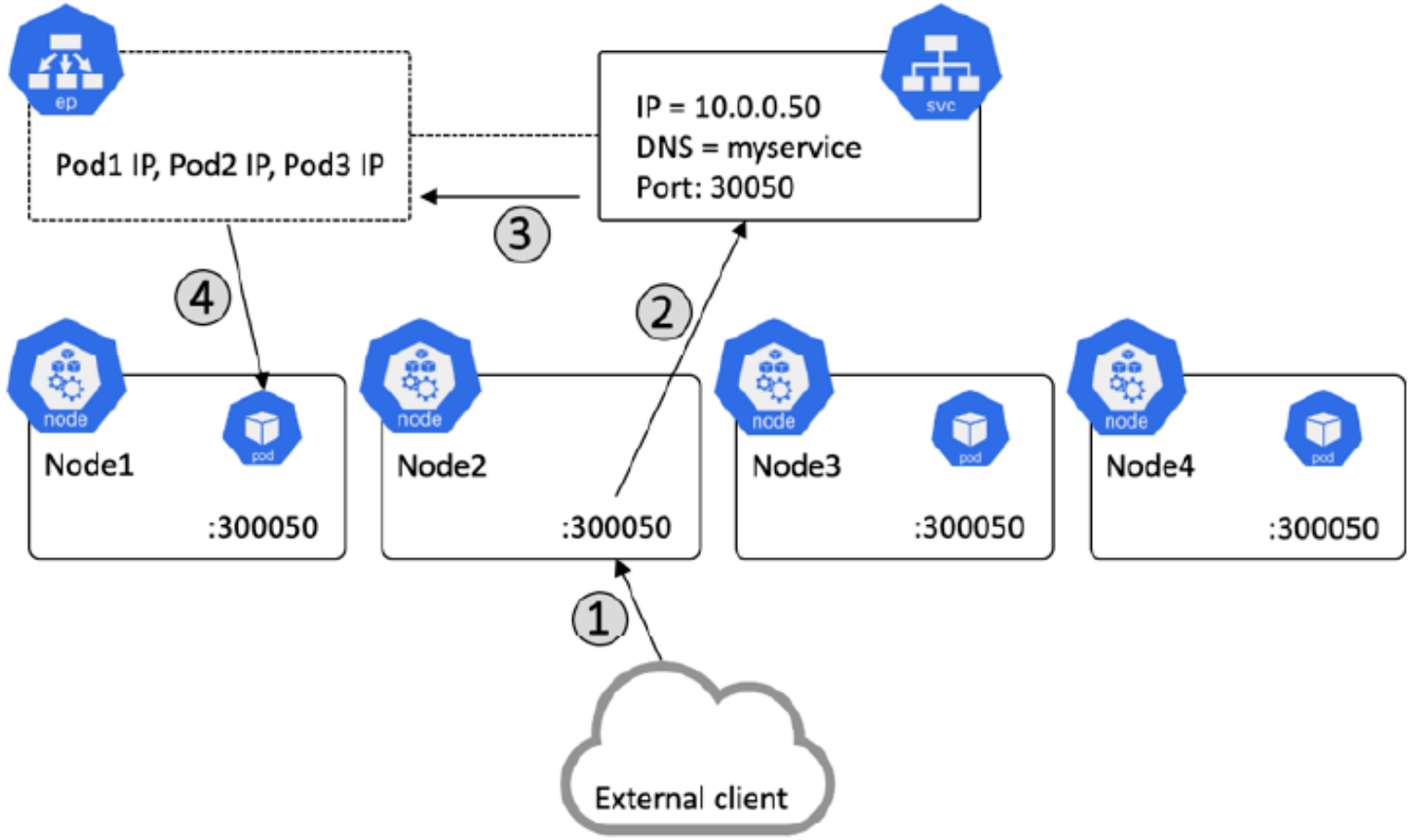
NodePort

- NodePort Service is built on top of ClusterIP and enables access from outside of the cluster.
- The service can be accessed from outside of the cluster by sending a request to the IP address of any cluster node on NodePort
- The Service object has a reliable NodePort mapped to every node in the cluster the NodePort value is the same on every node.
- This means that traffic from outside of the cluster can hit any node in the cluster on the NodePort and get through to the application (Pods).
- If you set the type field to NodePort, the Kubernetes control plane allocates a port from a range specified by --service-node-port-range flag (default: 30000-32767).

Example: NodePort

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
      # By default and for convenience, the `targetPort` is set to the same value as the `port` field.
    - port: 80
      targetPort: 80
      # Optional field
      # By default and for convenience, the Kubernetes control plane will allocate a port from a range (default: 30000-32767)
      nodePort: 30007
```

NodePort Service flow

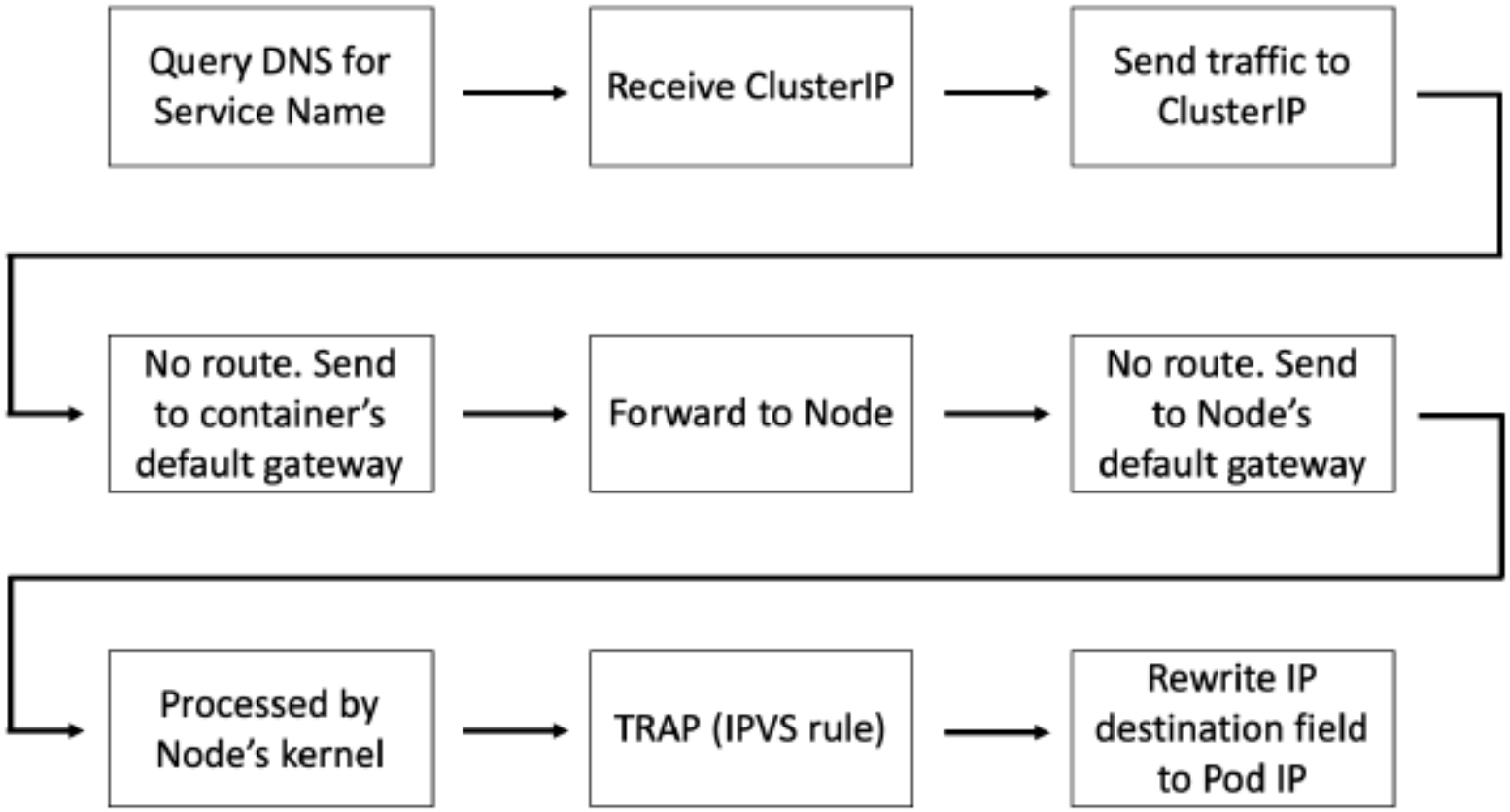


Multi-Port Services

- Kubernetes lets you configure multiple port definitions on a Service object.
- When using multiple ports for a Service, you must give all of your ports names so that these are unambiguous.

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - name: http
      protocol: TCP
      port: 80
      targetPort: 9376
    - name: https
      protocol: TCP
      port: 443
      targetPort: 9377
```

Service Flow



Find Us


https://www.facebook.com/K21Academy


http://twitter.com/k21Academy


https://www.linkedin.com/company/k21academy

https://www.youtube.com/dockerkubernetes


https://www.instagram.com/k21academy/