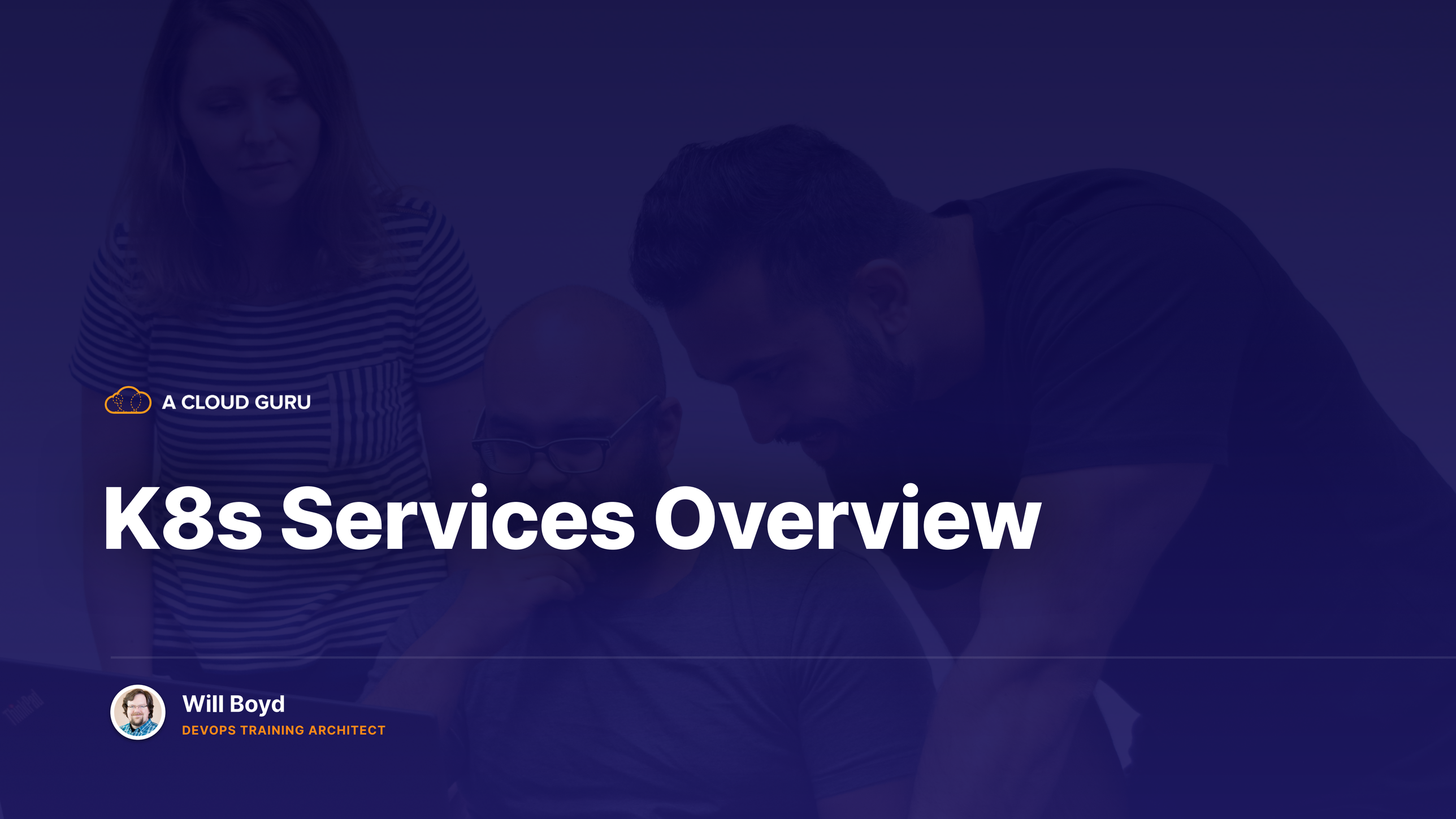


A CLOUD GURU

K8s Services Overview

Will Boyd

DEVOPS TRAINING ARCHITECT

LESSON BREAKDOWN

K8s Services Overview

What Is a Service?

Service Routing

Endpoints

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

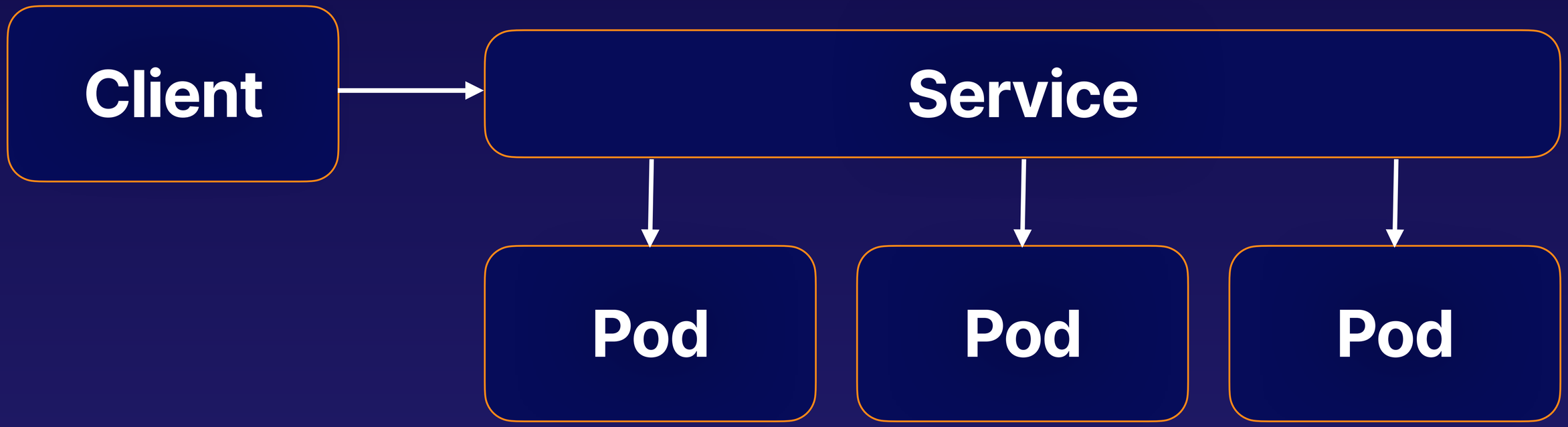
What Is a Service?

Kubernetes **Services** provide a way to expose an application running as a set of Pods.

They provide an abstract way for clients to access applications without needing to be aware of the application's Pods.

Service Routing

Clients make requests to a Service, which **routes** traffic to its Pods in a load-balanced fashion.



A CLOUD GURU

Endpoints

Endpoints are the backend entities to which Services route traffic. For a Service that routes traffic to multiple Pods, each Pod will have an endpoint associated with the Service.

Tip: One way to determine which Pod(s) a Service is routing traffic to is to look at that service's Endpoints.

LESSON BREAKDOWN

K8s Services Overview

What Is a Service?

Service Routing

Endpoints

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

Using K8s Services

Will Boyd

DEVOPS TRAINING ARCHITECT

LESSON BREAKDOWN

Using K8s Services

- Service Types
- ClusterIP Services
- NodePort Services
- LoadBalancer Services
- Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

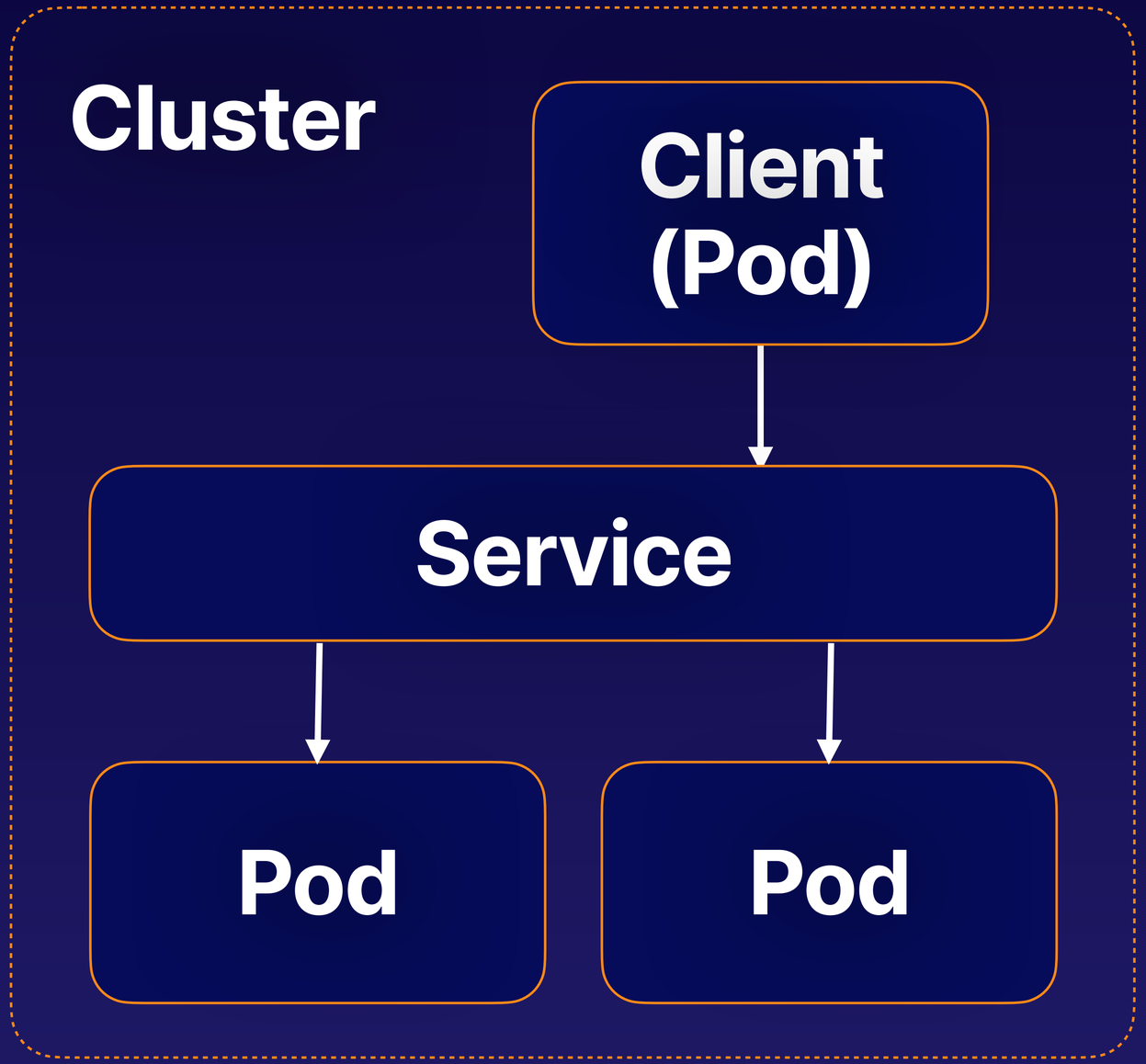
Service Types

Each Service has a type. The **Service type** determines how and where the Service will expose your application. There are four service types:

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName (outside the scope of CKA)

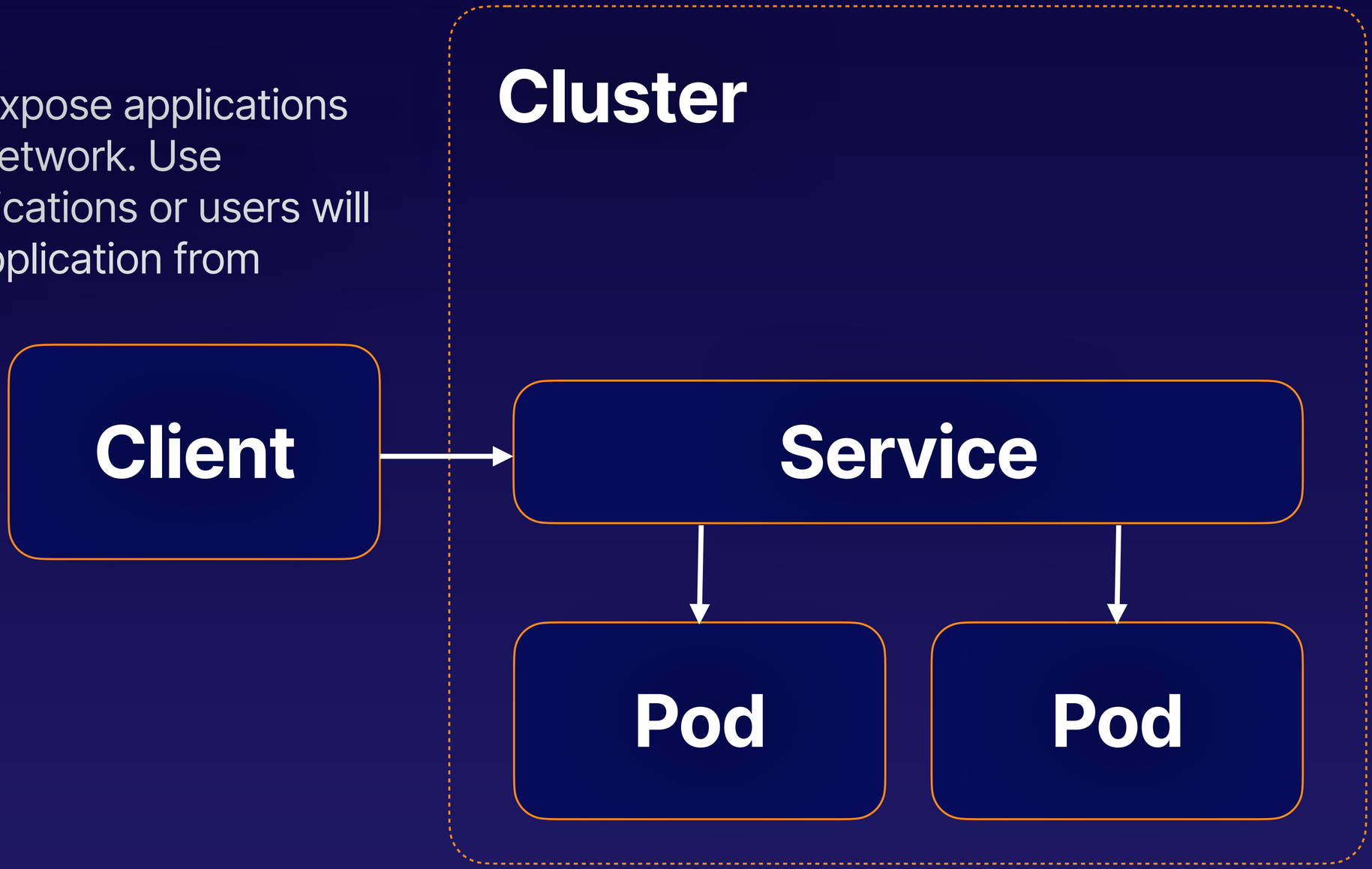
ClusterIP Services

ClusterIP Services expose applications **inside** the cluster network. Use them when your clients will be other Pods within the cluster.



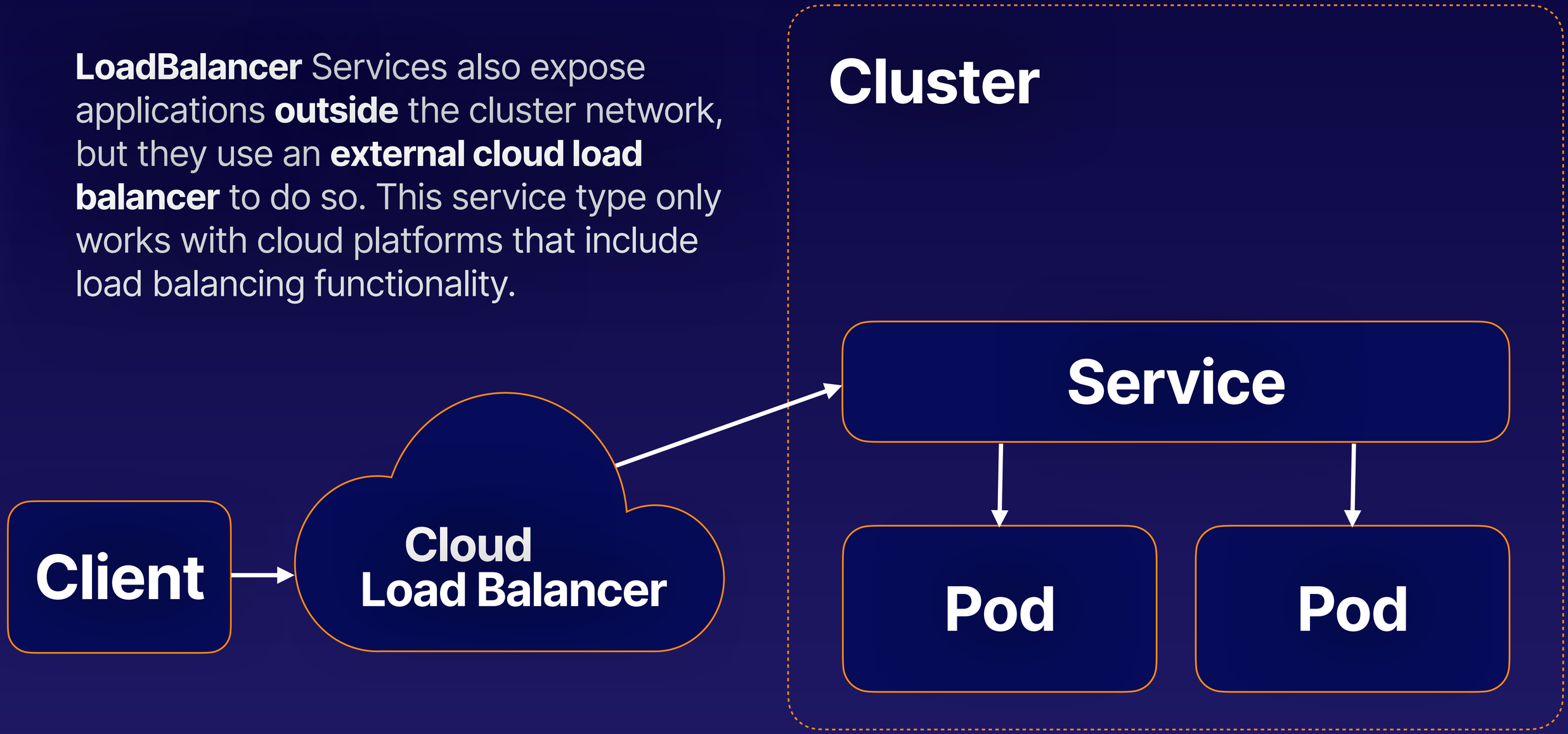
NodePort Services

NodePort Services expose applications **outside** the cluster network. Use NodePort when applications or users will be accessing your application from outside the cluster.



LoadBalancer Services

LoadBalancer Services also expose applications **outside** the cluster network, but they use an **external cloud load balancer** to do so. This service type only works with cloud platforms that include load balancing functionality.



Hands-On Demonstration

LESSON BREAKDOWN

Using K8s Services

- Service Types
- ClusterIP Services
- NodePort Services
- LoadBalancer Services
- Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

Discovering K8s Services with DNS

Will Boyd

DEVOPS TRAINING ARCHITECT

LESSON BREAKDOWN

Discovering K8s Services with DNS

Service DNS Names

Service DNS and Namespaces

Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

A CLOUD GURU

Service DNS Names

The Kubernetes DNS (Domain Name System) assigns **DNS names** to Services, allowing applications within the cluster to easily locate them.

A service's fully qualified domain name has the following format:

```
service-name.namespace-name.svc.cluster-domain.example
```

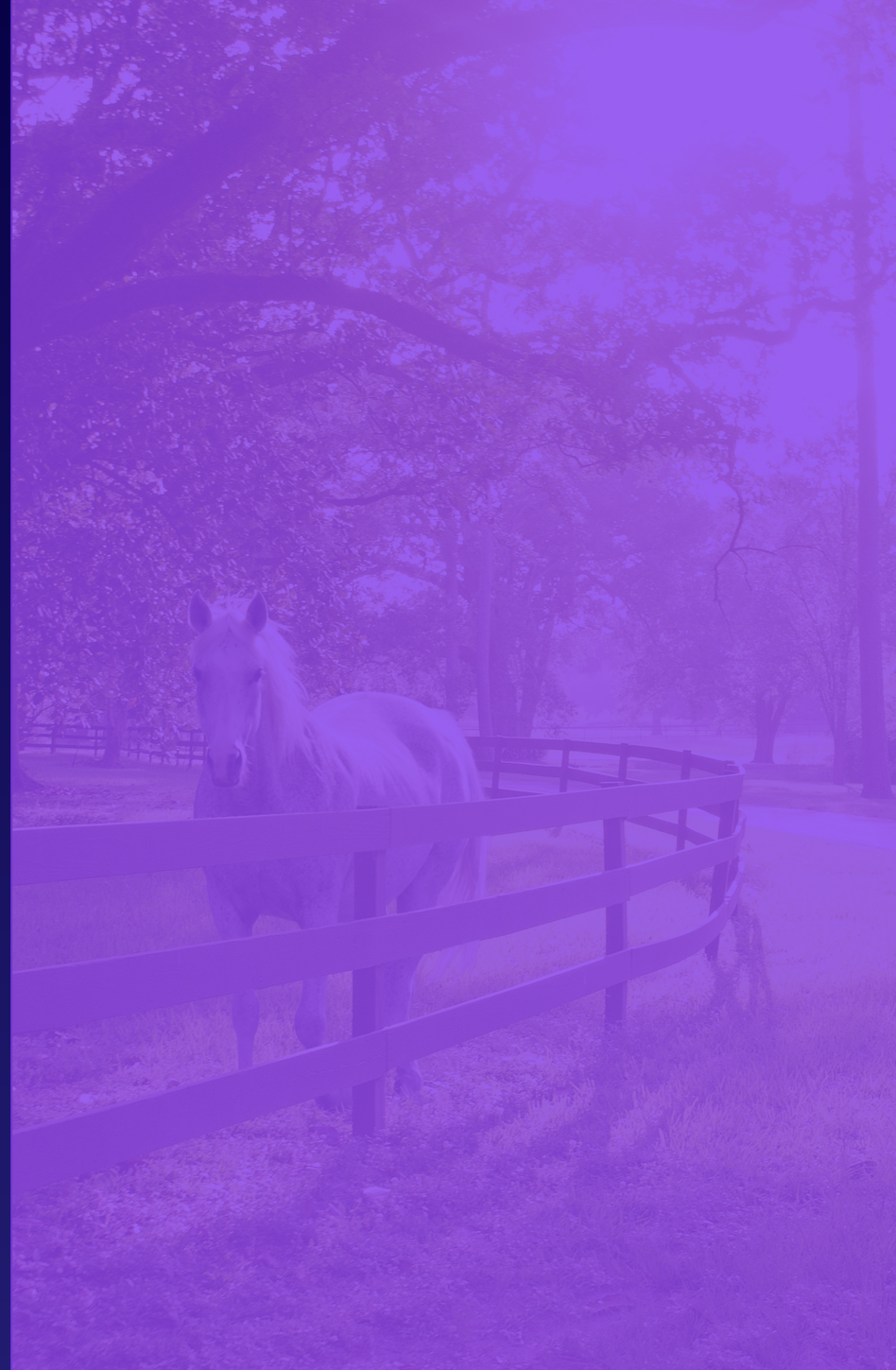
The default cluster domain is `cluster.local`.

A CLOUD GURU

Service DNS and Namespaces

A Service's fully qualified domain name can be used to reach the service from within **any Namespace** in the cluster.

```
my-service.my-namespace.svc.cluster.local
```



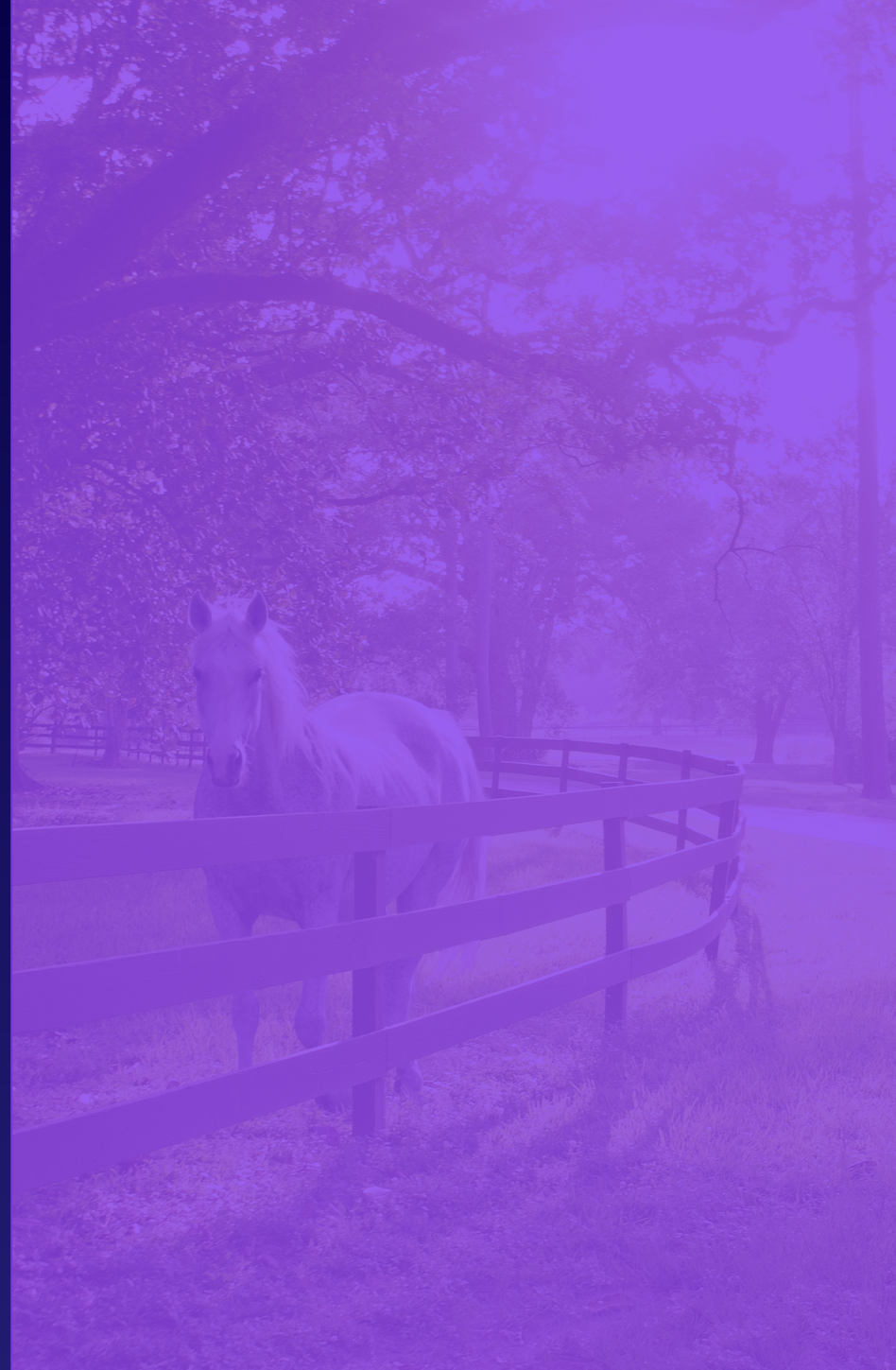
A CLOUD GURU

Service DNS and Namespaces

A Service's fully qualified domain name can be used to reach the service from within **any Namespace** in the cluster.

However, Pods within the same Namespace can also simply use the service name.

my-service



Hands-On Demonstration

LESSON BREAKDOWN

Discovering K8s Services with DNS

Service DNS Names

Service DNS and Namespaces

Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

Managing Access from Outside with K8s Ingress

Will Boyd

DEVOPS TRAINING ARCHITECT

LESSON BREAKDOWN

Managing Access from Outside with K8s Ingress

- What Is an Ingress?
- Ingress Controllers
- Routing to a Service
- Routing to a Service with a Named Port
- Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU

A CLOUD GURU

What Is an Ingress?

An **Ingress** is a Kubernetes object that manages external access to Services in the cluster.

An Ingress is capable of providing more functionality than a simple NodePort Service, such as SSL termination, advanced load balancing, or name-based virtual hosting.

A CLOUD GURU

Ingress Controllers

Ingress objects actually do nothing by themselves. In order for Ingresses to do anything, you must install one or more **Ingress controllers**.

There are a variety of Ingress Controllers available — all of which implement different methods for providing external access to your Services.

Routing to a Service

Ingresses define a set of **routing rules**. A routing rule's properties determine to which requests it applies.

Each rule has a set of **paths**, each with a **backend**. Requests matching a path will be routed to its associated backend.

In this example, a request to *http://<some-endpoint>/somepath* would be routed to port 80 on the `my-service` Service.

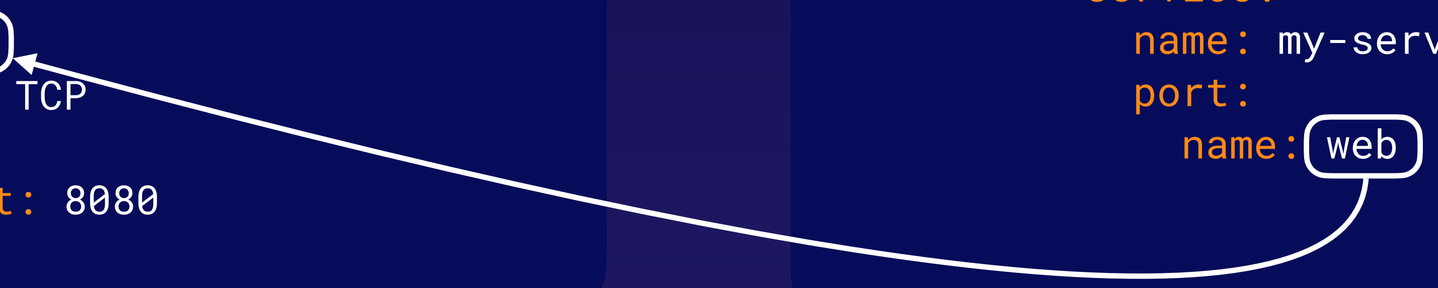
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
  - http:
      paths:
      - path: /somepath
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

Routing to a Service with a Named Port

If a Service uses a **named port**, an Ingress can also use the port's name to choose to which port it will route.

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - name: web
      protocol: TCP
      port: 80
      targetPort: 8080
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
  - http:
      paths:
      - path: /somepath
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              name: web
```



Hands-On Demonstration

LESSON BREAKDOWN

Managing Access from Outside with K8s Ingress

- What Is an Ingress?
- Ingress Controllers
- Routing to a Service
- Routing to a Service with a Named Port
- Hands-On Demonstration

Will Boyd
DEVOPS TRAINING ARCHITECT

A CLOUD GURU