Introducing the Ingress API Object and Controller

Nigel Brown

@n_brownuk www.windsock.io

Module Outline

Coming up:

- Introduce the Ingress concept and API
- Discuss the nature of ingress controllers
- Differentiate between host-based and path-based routing
- Learn how to define ingress objects to route requests to backends

Ingress

A Kubernetes API object that manages the routing of external HTTP/S traffic to services running in a cluster

Ingress Objects and Controllers

Ingress Object Definition

+

Ingress Controller Deployment

Why Third-party Controllers?

Part of the ingress controller's function, is to act as a reverse proxy for cluster workloads

Best of breed is subjective, instead we have the ability to choose a solution that works for us

Infrastructure providers can create ingress controllers optimized for their environments

Proxy-based Ingress Controllers

Nginx

Maintained by the k8s community

https://git.io/fh4UC

Traefik

Based on a cloud native edge router

https://git.io/Jfl6E

Contour

Uses the popular Envoy service proxy

https://git.io/fh4Ua

Characteristics of the Ingress API

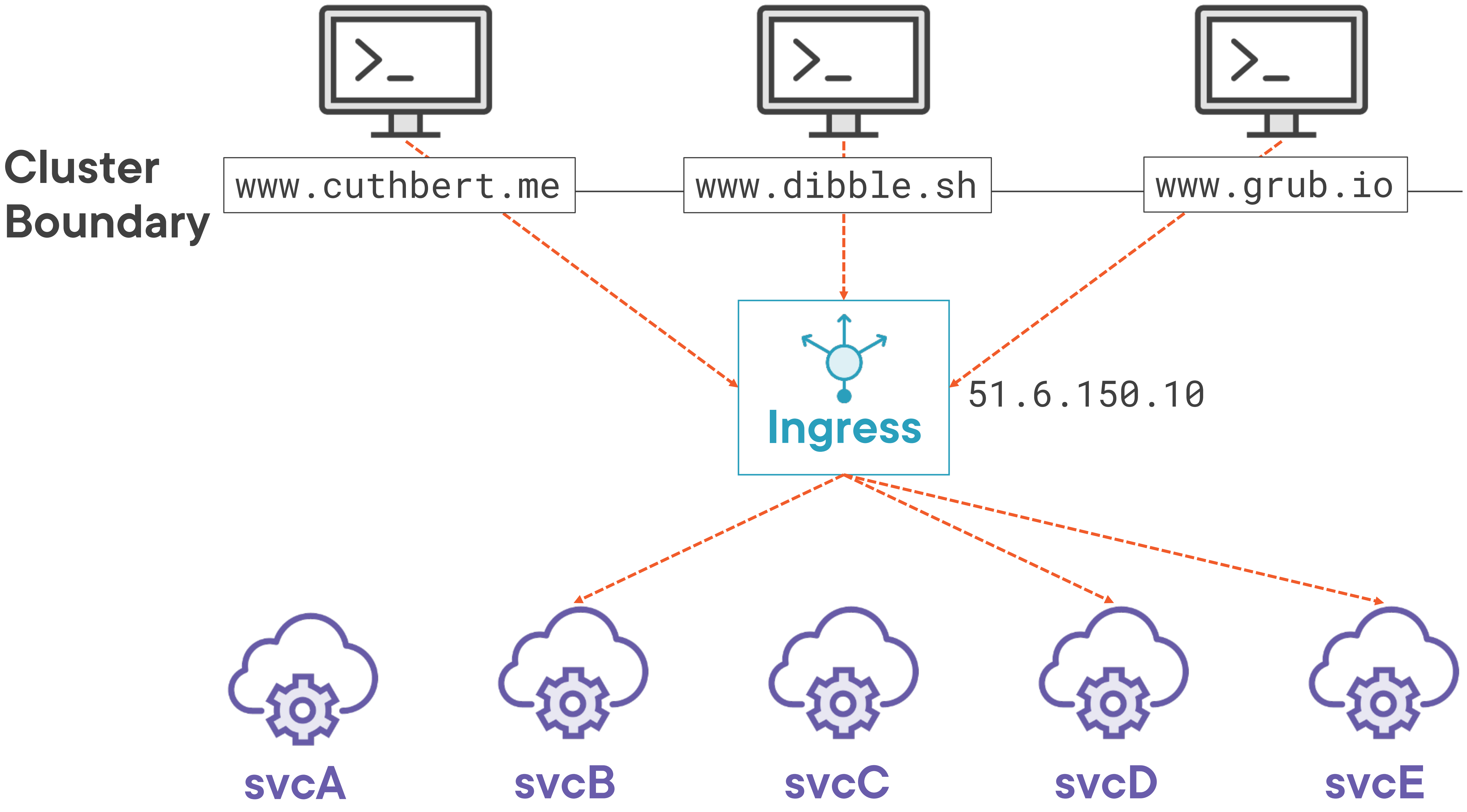
Defines traffic routes between external clients and services

Allows encrypted communication using Transport Layer Security

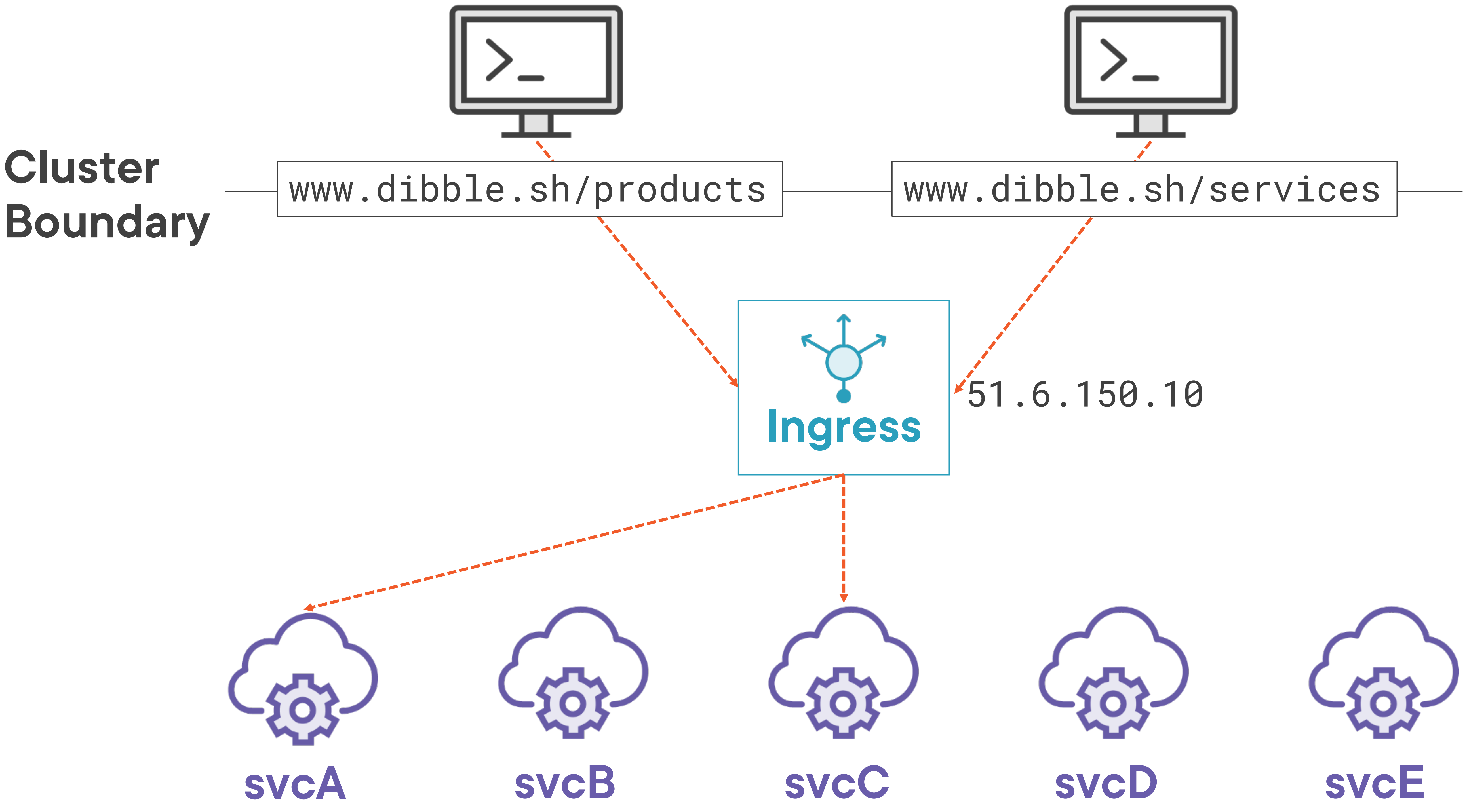
Load balances client traffic across a service's endpoints (pods)

The Ingress API has remained largely static, and is likely to evolve

Name-based Virtual Hosts



Path-based Routing

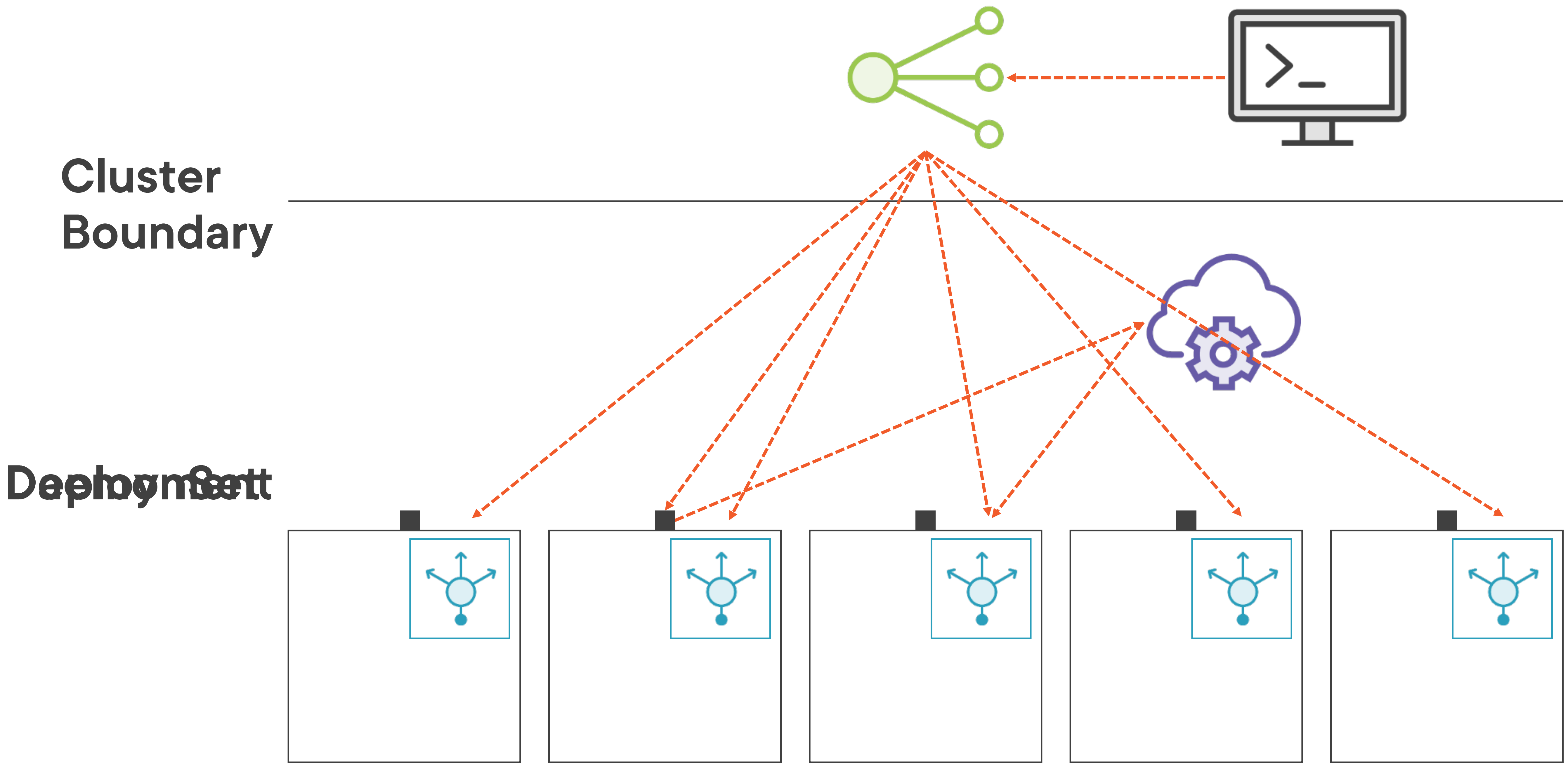


Demo

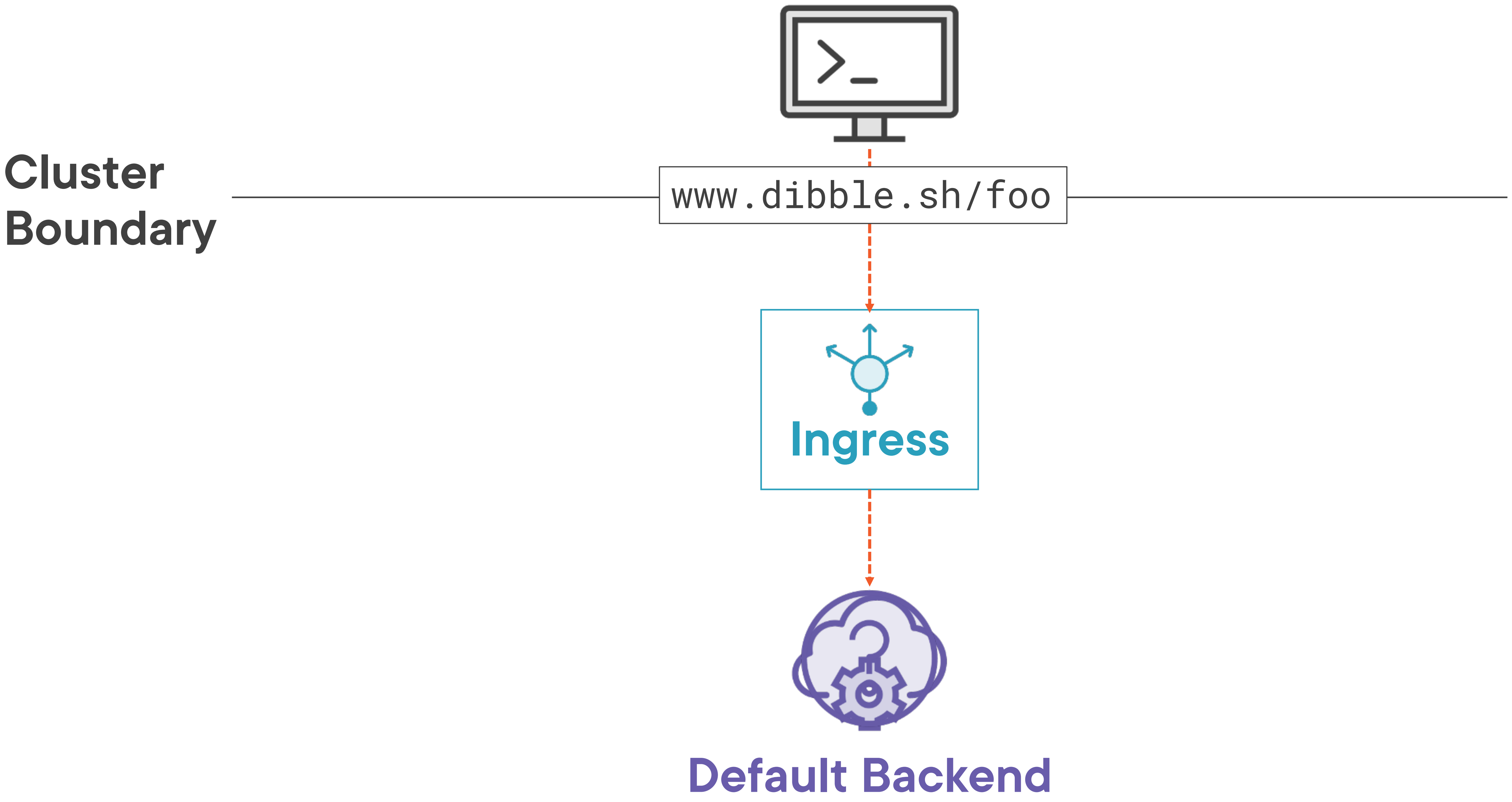
Deploying the Nginx Ingress Controller

- Discuss the available deployment options for ingress controllers
- Deploy the open source Nginx ingress controller
- Inspect the API objects created in the cluster

Ingress Controller Deployment



Invalid Requests



Default Backend

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: single-default-backend-ingress
spec:
  defaultBackend:
    service:
      name: default-backend-svc
      port:
        number: 8080
```

The default backend has been interpreted to mean different things to different people

Ambiguity of the Default Backend Service

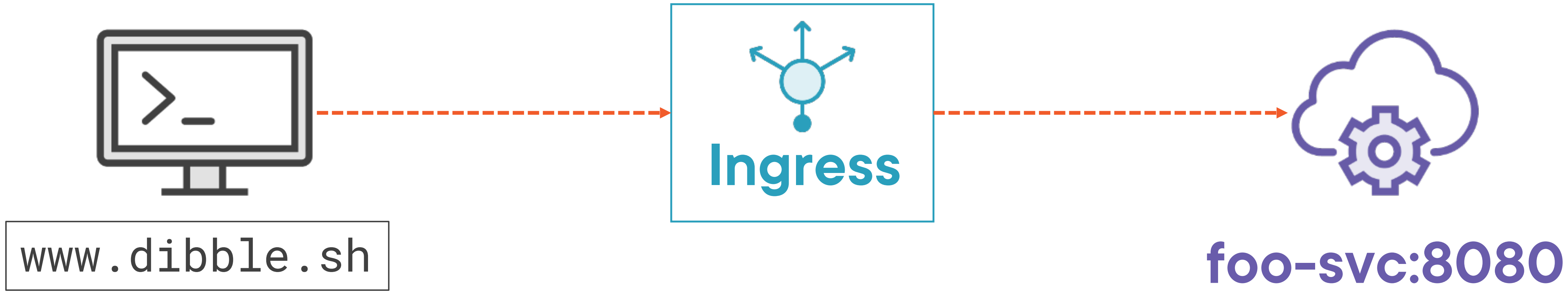
Perfect design for routing ingress traffic to a single backend service

Works with ingress definitions containing rules for HTTP requests

Some ingress controllers deploy a default backend during installation

Documentation implies default backend is controller’s responsibility

Host Rule Routing



www.dibble.sh

Ingress

foo-svc:8080

Ingress Host Rule

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: host-rule-ingress
spec:
  rules:
  - host: www.dibble.sh
    http:
      paths:
      - backend:
          service:
            name: foo-svc
            port:
              number: 8080
        pathType: ImplementationSpecific
```

Host Field Permutations

```
Host: www.dibble.sh
```

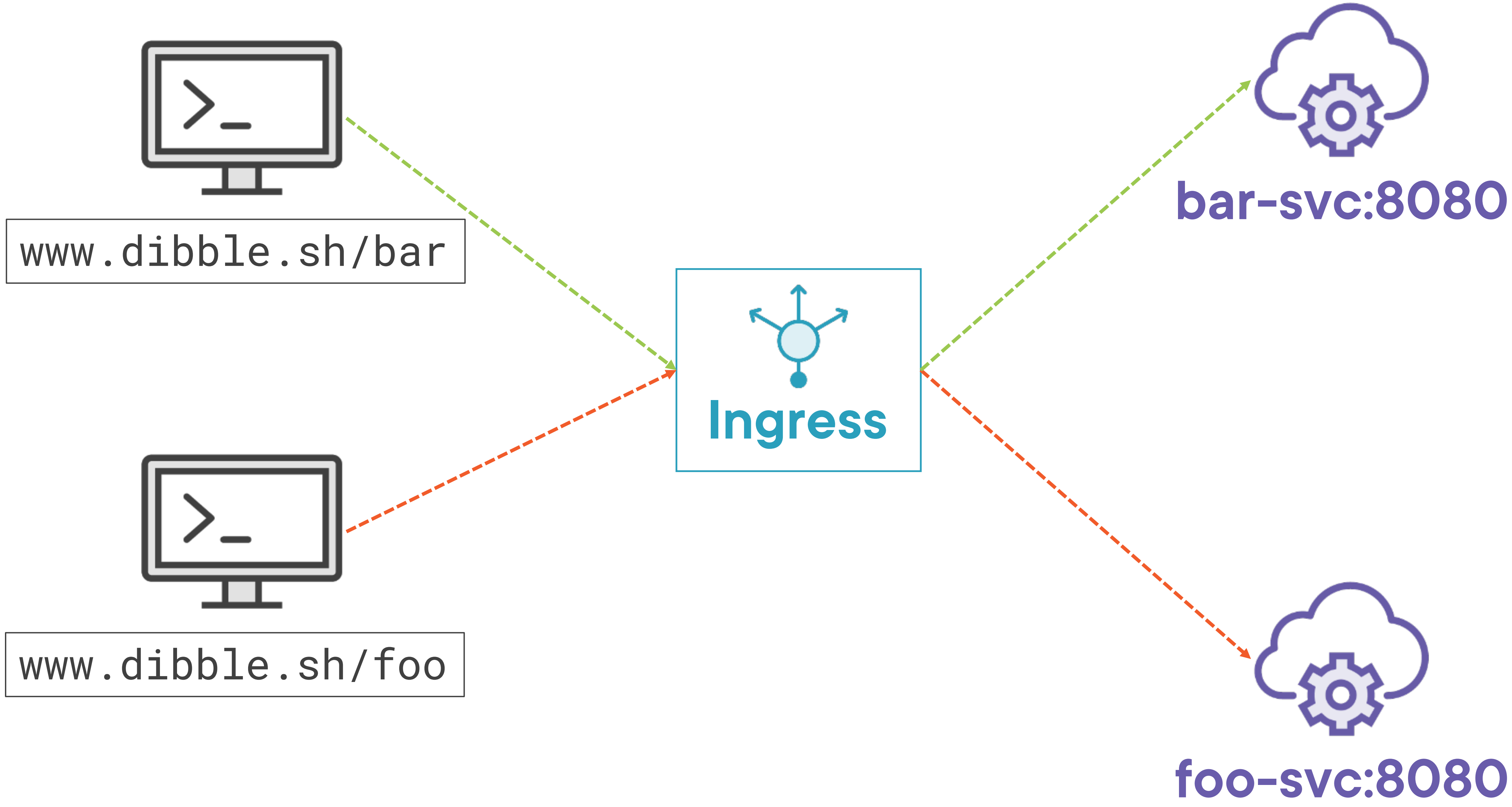
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: host-rule-ingress
spec:
  rules:
  - http:
      ...

  - host: www.dibble.sh
    http:
      ...
```

```
Host: blog.dibble.sh
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: host-rule-ingress
spec:
  rules:
  - host: *.dibble.sh
    http:
      ...
```

Path Rule Routing



Ingress Path Rule

```
<snip>
spec:
  rules:
  - http:
      paths:
      - path: /foo
        backend:
          service:
            name: foo-svc
            port:
              number: 8080
        pathType: Prefix
      - path: /bar
        backend:
          service:
            name: bar-svc
            port:
              number: 8080
        pathType: Prefix
```

Path Types

Specifies how a path in an incoming request is matched against a path definition in the ingress object

Exact match

Request path must match ingress path exactly

Prefix match

Request path matches a URL path prefix defined in ingress

Controller specific

Match is determined by the ingress controller implementation

Demo

Using Ingress Rules to Route to Backends

- Define and deploy an ingress for default backend service
- Demonstrate host-based routing rules
- Routing to backend services based on URL paths

Module Summary

What we covered:

- The Ingress API allows us to define routes for ingress traffic
- Ingress controllers fulfil the routing defined in Ingress objects
- Default backends help to manage the response to invalid requests

Securing Ingress Traffic

The Ingress API facilitates secure communication with services using the Transport Layer Security