INTRODUCTION TO KUBERNETES INGRESS

In this blog post, we'll cover Kubernetes Ingress, including what it is and some of the concepts you'll need to learn. This articles assumes you have a solid understanding of Kubernetes.

What is Kubernetes ingress?

To start simply, a Kubernetes ingress exposes HTTP and HTTPS routes from outside of a cluster to services created inside the cluster. In general, you'll find that:

- Rules defined on the ingress resource control traffic routes.
- You can set up ingress to do several things:
    - Provide services externally-reachable URLs
    - Load balance traffic
    - Offer name-based virtual hosting
    - Terminate SSL (secure sockets layer) or TLS (transport layer security)
- Ingress controllers handle the ingress with the help of a load balancer. We can also achieve this outcome with an edge router, which is a virtual or physical router that enforces the firewall policy for the cluster.
- Ingress does not expose any random port; only HTTP and HTTPS.

Understanding ingress resource

Ingress is typically defined with an ingress resource. Like most K8S resources, the ingress resource needs apiVersion, kind, and metadata.

- Annotation can be used to configure ingress controller with certain options, e.g., **rewrite-target** which targets URI where the traffic must be redirected for nginx ingress.
- The spec section is where you'll configure the load balancer or the proxy server.

An example of a resource looks like this:

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: demo-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /demopath
        backend:
          serviceName: demo
          servicePort: 80
```

Ingress rules

The ingress rules are a set of rules used to process incoming traffic to services in the cluster. Using the example above, let's go over the rules section of the spec.

1. **Specify how the rule applies.** We can use HTTP/S or host-based rule (foo.bar.com). In the example above, we are using HTTP as opposed to host-based.
2. **Define the path and the backend service/port to route to.** In our example above, the service name is demo with port 80. (Note: Both the host and the path must match the content of an incoming request *before* the load balancer will direct traffic to the referenced service.)

If a rule is not defined, all traffic will be routed to a default backend that is specified in ingress controller.

Types of ingress

The three types of ingress are:

- **Single service** exposes a single service just like NodePort for example.
- **Simple fanout** routes traffic from a single IP to multiple services based on the URI.
- **Name-based virtual hosting** routes traffic to multiple hostnames on the same IP.

TLS (transport layer security)

Securing an ingress is simple. In the resource manifest, specify a secret with a private key and certificate.

Load balancing

An ingress controller has some load balancing policy settings apply to all ingress. Two common ones are the load balancing algorithm and the backend weight scheme.

Ingress controllers

For ingress resources to work, an ingress controller must be running. This type of controller is not like other controllers in k8s (i.e., part of the cluster). Instead, the ingress controller is a separate entity that must be deployed separately.

Examples of ingress controllers are:

- GCE
- nginx
- istio
- Kong

Refer to K8S documentation to a see full list. You can run any number of controllers in the cluster so long as they are annotated properly with **ingress.class**.