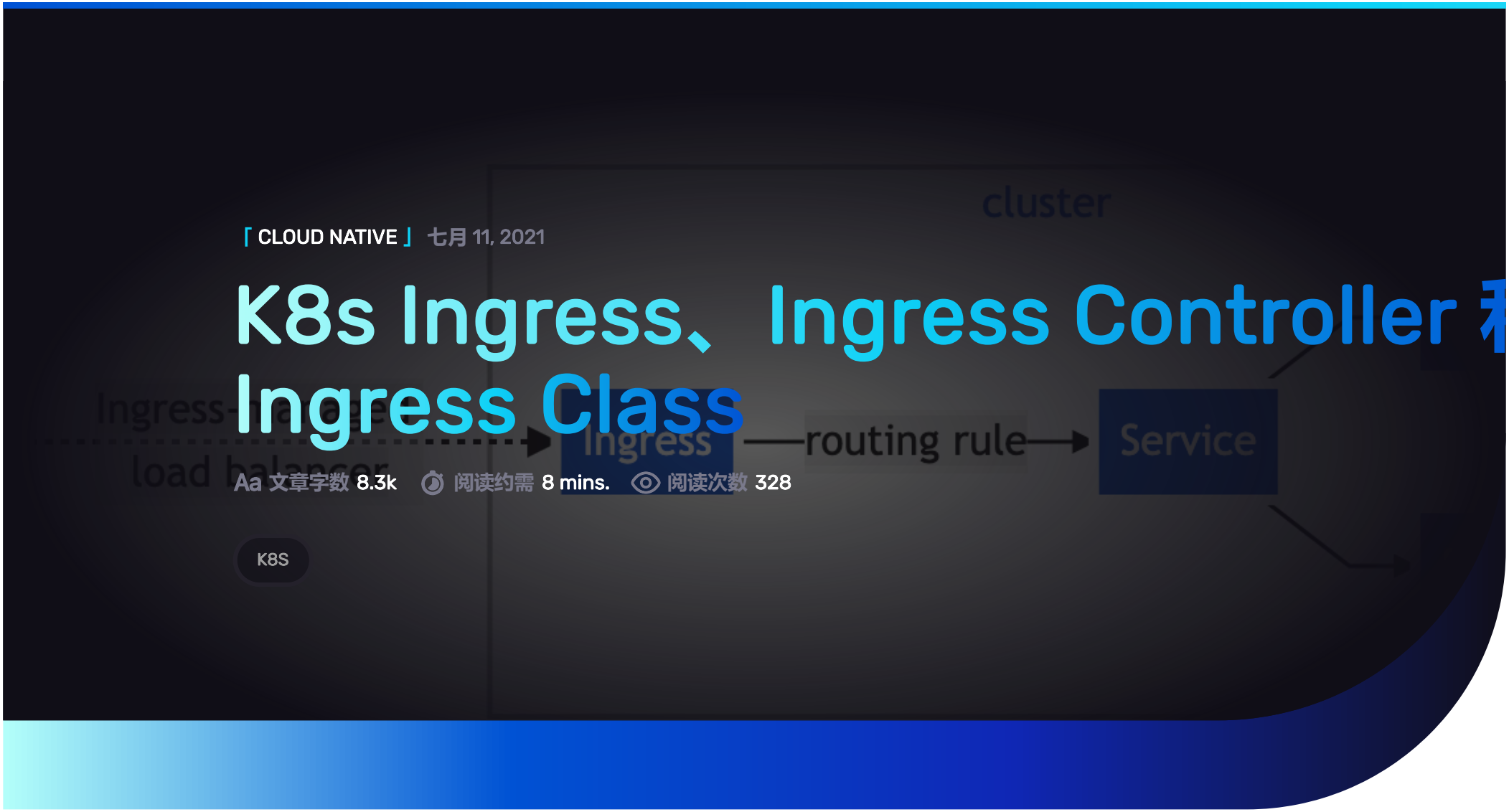

「CLOUD NATIVE」七月 11, 2021

K8s Ingress、Ingress Controller 和 Ingress Class

Aa 文章字数 8.3k 阅读约需 8 mins. 阅读次数 328

K8S

将 k8s 集群中服务暴露给集群外访问，最简单的方式莫过于使用 NodePort，好比在 docker 环境下为容器的服务端口绑定宿主机的端口，定义 NodePort 类型的 Service 后，即可通过集群中任意节点的 IP 加 nodePort 指定的端口访问到 k8s 集群中的服务。

但随着服务的增多，使用 NodePort 访问的问题也会逐渐显现出来：可用作 NodePort 的端口是一个有限的范围、不容易记忆、不好管理……

有没有更优雅的方式访问集群内的服务呢?

可以在集群内部署一个 Nginx 服务，NodePort 暴露 Nginx 的端口，再由 Nginx 代理访问集群内的服务。emm，没错，这种方式的确更好。在 k8s 中也有这样的一个资源，能够起到与这个 Nginx 类似的作用，即 Ingress。

Ingress

Ingress 在 k8s 集群中的作用，是定义外部对集群内服务的访问路由，例如：

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /testpath
        pathType: Prefix
        backend:
          service:
```

```
15          name: test
16          port:
17            number: 80
```

上面这个 Ingress 资源的定义，配置了一个路径为 `/testpath` 的路由，所有 `/testpath/**` 的入站请求，会被 Ingress 转发至名为 `test` 的服务的 80 端口的 `/` 路径下。

可以将 Ingress 狭义的理解为，Nginx 中的配置文件，如：`nginx.conf`。

Ingress Controller

很显然，只有 Nginx 的配置文件，是起不到转发请求的作用的，必须还要有 Nginx 程序。

同样，仅创建 Ingress 资源本身是没有任何作用的，还需要部署 **Ingress Controller**。Ingress Controller 的作用就相当于是 Nginx 服务，实际上，k8s 官方支持和维护的三个 Ingress Controller 里，就有基于 nginx 实现的 **ingress-nginx**，另外两个是 **AWS** 和 **GCE**。

除此之外，还有很多**三方实现**，例如：

- F5 BIG-IP 的 Container Ingress Services for Kubernetes 让你能够使用 Ingress 来配置 F5 BIG-IP 虚拟服务器
- Gloo 是一个开源的、基于 Envoy 的 Ingress 控制器，能够提供 API 网关功能
- HAProxy Ingress 针对 HAProxy 的 Ingress 控制器

- Istio Ingress 是一个基于 Istio 的 Ingress 控制器
- NGINX Ingress Controller for Kubernetes 能够与 NGINX Web 服务器（作为代理）一起使用
- Traefik Kubernetes Ingress provider 是一个用于 Traefik 代理的 Ingress 控制器

既然有这么多类型的实现，我们就有可能会有需要部署多个 Ingress Controller 的场景。当集群中部署了多个 Ingress Controller 的时候，如何知道 Ingress 中的规则应该使用哪个 Controller 呢？这时就需要用到 Ingress Class 了。

Ingress Class

不同类型的 Ingress Controller 对应的 Ingress 配置通常也是不同的，当集群中存在多于一个的 Ingress Controller 时，就需要为 Ingress 指定对应的 Controller 是哪个。

kubernetes.io/ingress.class

在 Kubernetes 1.18 之前，通常是在 Ingress 资源上通过 `kubernetes.io/ingress.class` 注解来指定使用的 Ingress Controller。虽然这个注解从未被正式定义过，但确是被各个 Ingress Controller 所广泛支持的。

这个注解的值，一般是具体 Ingress Controller 所提供的默认值，如 `nginx`、`gce`、`traefik`、`kong` 等，可使用约定关键字，或查阅对应文档获得此默认值，如 Kong 的 Kubernetes Ingress Controller 文档中 **Configuring the controller ingress class** 部分。

在创建 Ingress Controller 部署对象时，我们也可以通过 `--ingress-class` 来指定此 Ingress Controller 具体 Deployment 的对应 class，如：

```yaml
spec:
  template:
    spec:
      containers:
        - name: nginx-ingress-internal-controller
          args:
            - /nginx-ingress-controller
            - '--ingress-class=nginx-internal'
            - '--configmap=ingress/nginx-ingress-internal-controller'
```

Ingress 中指定使用的 Ingress Controller 时，可参考如下方式：

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: ingress-issue2349
  annotations:
    kubernetes.io/ingress.class: nginx-internal
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
```

```
    paths:
    - backend:
        serviceName: echoheaders
        servicePort: 80
      path: /jsonRPC

```

IngressClass

Kubernetes 1.18 起，正式提供了一个 **IngressClass** 资源，作用与 `kubernetes.io/ingress.class` 注解类似，例如：

YAML

```yaml
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: external-lb
spec:
  controller: nginx-ingress-internal-controller
  parameters:
    apiGroup: k8s.example.com
    kind: IngressParameters
    name: external-lb
```

其中重要的属性是 `metadata.name` 和 `spec.controller`，前者是这个 IngressClass 的名称，需要设定在 Ingress 中，后者是 Ingress Controller 的名称。

Ingress 中的 `spec.ingressClassName` 属性，可以用来指定对应的 IngressClass，并进而由 IngressClass 关联到对应的 Ingress Controller，如：

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: spring-k8s
spec:
  ingressClassName: external-lb
  defaultBackend:
    service:
      name: spring-k8s
      port:
        number: 80
```

注意，`spec.ingressClassName` 与 `metadata.annotations.kubernetes.io/ingress.class` 的作用并不完全相同，因为 `ingressClassName` 字段引用的是 IngressClass 资源的名称，IngressClass 资源中，除了指定了 Ingress Controller 的名称之外，还可能会通过 `spec.parameters` 属性定义一些额外的配置。

默认 Ingress Class

在集群中，我们可以设定一个默认的 Ingress Class，以便处理所有没有指定 Ingress Class 的 Ingress 资源。

在 `IngressClass` 资源上，我们可以通过将 `ingressclass.kubernetes.io/is-default-class` 注解的值设定为 `true`，来使没有设置 `ingressClassName` 的 Ingress 使用此默认的 `IngressClass`。

> 注意：当存在多个默认 Ingress Class 时，新的 Ingress 如果没有指定 `ingressClassName` 则不会被允许创建。解决这个问题只需确保集群中最多只能有一个 IngressClass 被标记为默认。

多个 Ingress Controller

除了可能会有多个不同类型的 Ingress Controller 之外，还可能存在多个相同类型的 Ingress Controller，比如部署了两个 NGINX Ingress Controller，一个负责处理外网访问，一个负责处理内网访问。

此时也可以通过上面的方式，为每个 Controller 设定唯一的一个 class。

当多个 controller 的 class 不唯一，或者 controller 和 Ingress 都没有指定 class 又没有默认的 class 时，会导致所有符合条件的 Ingress Controller 竞争满足 Ingress 配置，可能会导致不可预测的结果。

参考资料

- Rewrite
- Multiple Ingress controllers
- Ingress not working when kubernetes.io/ingress.class used
- kubernetes中常用的Ingress Controller