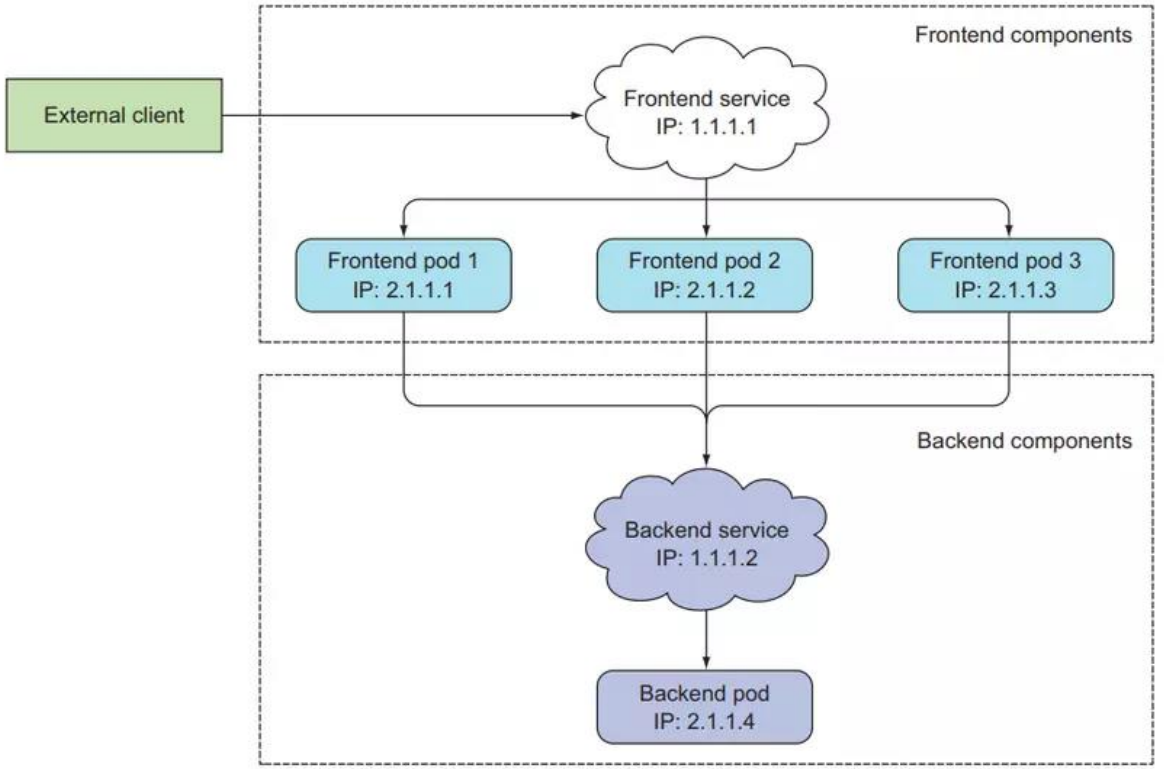
Lesson 4: Kubernetes Services and Networking

Lữ Thanh Tùng

Kubernetes Services

- Là một resouce sẽ tạo ra một mạng kết nối với một nhóm Pod phía sau nó, khi client giao tiếp với các Pod sẽ thông qua service thay vì giao tiếp trực tiếp.
- Mỗi service sẽ có một địa chỉ IP và port không đổi, trừ khi ta xóa nó đi và tạo lại. Client sẽ mở connection tới service, và connection đó sẽ được dẫn tới một trong những Pod ở phía sau.



Vai trò của Kubenetes Service trong việc kết nối đến Pod

- **Pods are ephemeral:** Pod được tạo ra, bị xóa, và thay thế bằng một pod khác bất cứ lúc nào. Khi pod mới tạo ra thì sẽ có một IP khác với pod cũ. Nếu ta dùng IP của Pod để tạo connection tới client thì lúc Pod được thay thế với IP khác thì ta phải update lại code.
- **Multiple Pod run same application:** Có nghĩa là ta sẽ có nhiều pod đang chạy một ứng dụng của chúng ta để tăng performance. Ví dụ khi ta dùng ReplicaSet với replicas = 3, nó sẽ tạo ra 3 Pod. Vậy làm sao ta biết được nên gửi request tới Pod nào?
  - => Service sẽ tạo ra một endpoint không đổi cho các Pod phía sau, client chỉ cần tương tác với endpoint này.
  - Một vấn đề nữa là khi có nhiều replica như vậy thì kubernetes sẽ tìm kiếm nhóm pod ấy bằng cách nào ?

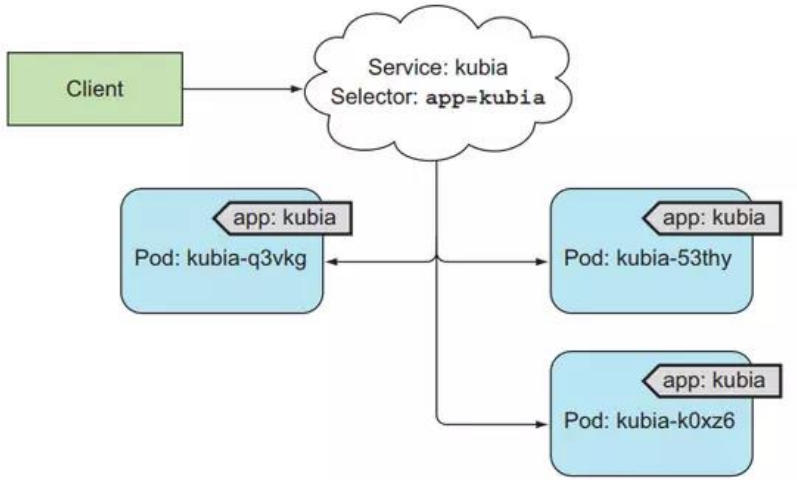


Figure 5.2 Label selectors determine which pods belong to the Service.

Tạo một Service

- Ta có thể tạo một Service đơn giản bằng cách tạo một file yaml như hình bên
  - Trong đó: port: 80 là cổng của service, còn targetPort: 9376 là cổng của Pod mà service sẽ kết nối tới

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

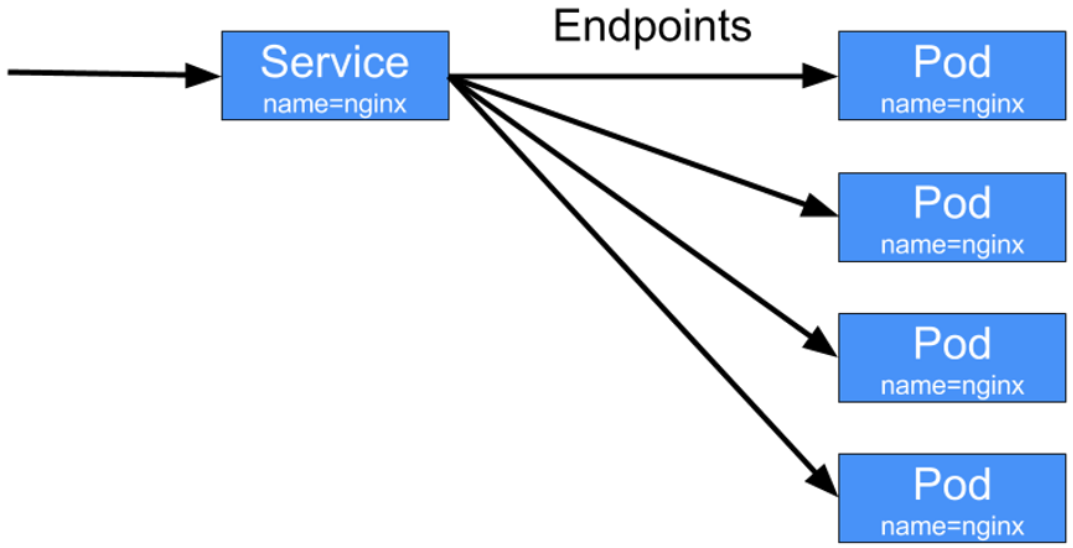
- Ta có thể đặt tên các cổng của Pod và đặt targetPort bằng tên của cổng đó
- Điều này mang lại rất nhiều tính linh hoạt để triển khai và phát triển Dịch vụ của . Ví dụ: Ta có thể thay đổi số cổng mà Pod hiển thị trong phiên bản tiếp theo của phần mềm phụ trợ mà không làm hỏng service.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    app.kubernetes.io/name: proxy
spec:
  containers:
  - name: nginx
    image: nginx:stable
    ports:
      - containerPort: 80
        name: http-web-svc

---
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app.kubernetes.io/name: proxy
  ports:
  - name: name-of-service-port
    protocol: TCP
    port: 80
    targetPort: http-web-svc
```

EndpointSlices

- Khi cấu hình service theo cách đã đề cập ở trên, service sẽ tìm đến các Pod có cùng tên hoặc cổng.
- Đôi khi, chúng ta cần trỏ đến một số Pod nhất định, thay vì đến tất cả các Pod,
hoặc services trỏ đến 1 service khác. Khi đó, ta sẽ dùng Endpoint
- Endpoint được khởi tạo như ví dụ bên, khi đó địa chỉ trỏ đến là các EndpointSubset: 192.0.2.45:9376
- EndpointSubset là một nhóm địa chỉ có một bộ cổng chung. Tập các subset này là kết quả của phép nhân Đề-các: addresss x port

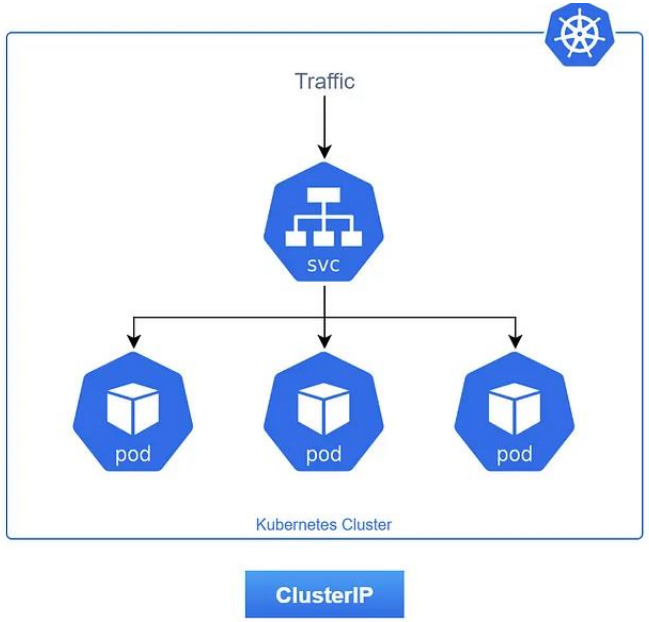



```yaml
apiVersion: v1
kind: Endpoints
metadata:
  name: my-service #<------ Should match the name of Service
subsets:
  - addresses:
      - ip: 192.0.2.45
    ports:
      - port: 9376
```

Các loại Service

1.ClusterIP

- ClusterIP service là loại service mặc định trong Kubernetes.
- ClusterIP service chỉ có thể được truy cập bởi các ứng dụng khác cùng nằm trong cụm của bạn.
- Các ứng dụng bên ngoài cụm sẽ không thể truy cập đến loại service này.
- Ta có thể khởi tạo loại service này như ví dụ bên:

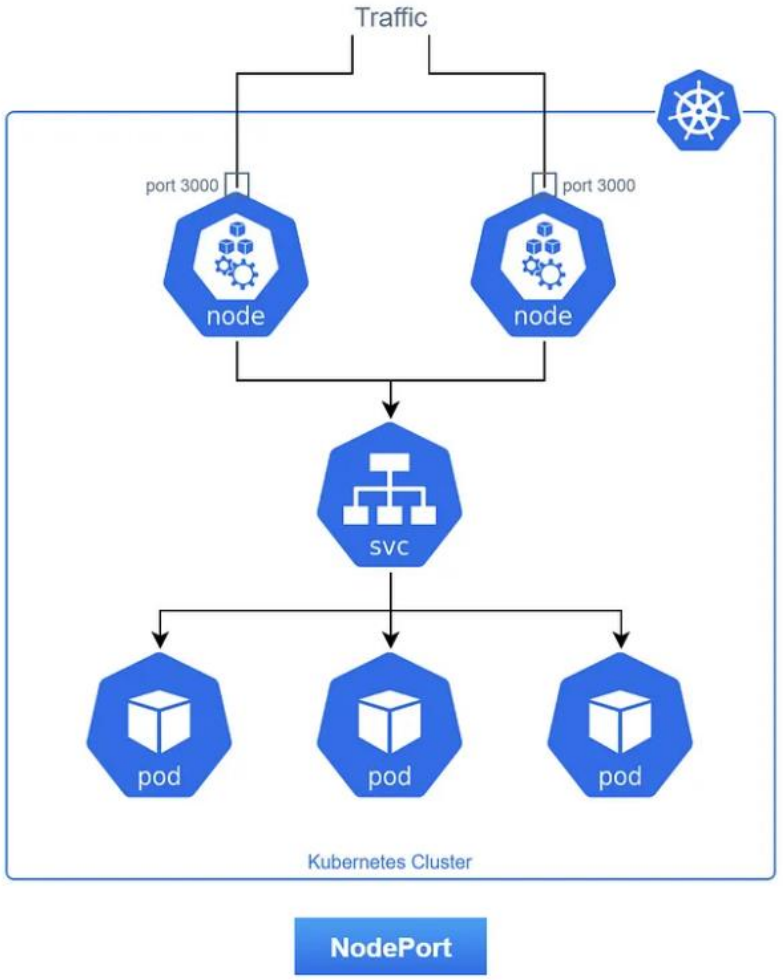



```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-backend-service
spec:
  type: ClusterIP # Optional field (default)
  clusterIP: 10.10.0.1 # within service cluster ip range
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 8080
```

Các loại Service

2.NodePort

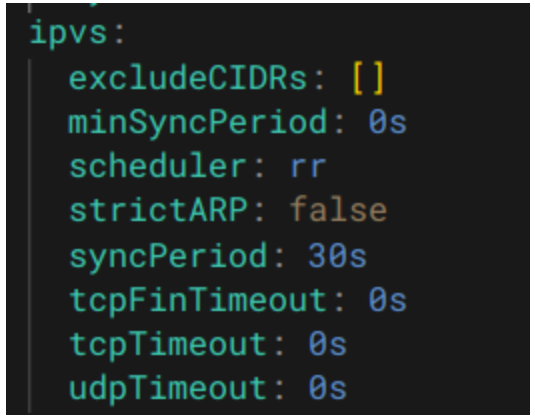
- Service kiểu NodePort là một cách đơn giản nhất để truy cập đến service từ phía bên ngoài cụm
- Service NodePort sẽ mở một port (trong khoảng 30000-32767) để tiếp nhận lưu lượng truy cập từ bên ngoài. Ta có thể trực tiếp chỉ định port này trong file yaml của mình hoặc để Kubernetes tự động chỉ định.
- Các request ở trên port đã được định nghĩa ở NodePort đều được gửi đến service đó. Vì vậy, cần chắc chắn không có dịch vụ nào cùng sử dụng port này
- Địa chỉ IP của NodePort có thể bị thay đổi



Các loại Service

2.NodePort

- Cơ chế cân bằng tải trong NodePort thuộc về **kube-proxy .** Kube-proxy chạy trên mọi node và chịu trách nhiệm chặn các kết nối đến địa chỉ Cluster IP và cân bằng tải đến nhóm các pod phía sau mỗi service
- Cơ chế cân bằng tải mặc định của kube-proxy là sẽ random theo tỷ lệ số node nhận tải mà không quan tâm đến node có đang làm việc hay không (ví dụ có 4 node thì tỷ lệ nhận tải của mỗi node là ¼). Nếu muốn các cơ chế khác thì phải chỉnh sửa trong cấu hìnhhình



Một đoạn config của kube-proxy, cơ chế scheduler của cân bằng tải là Round Robin (rr) thay vì autoauto

Các loại Service

2.NodePort

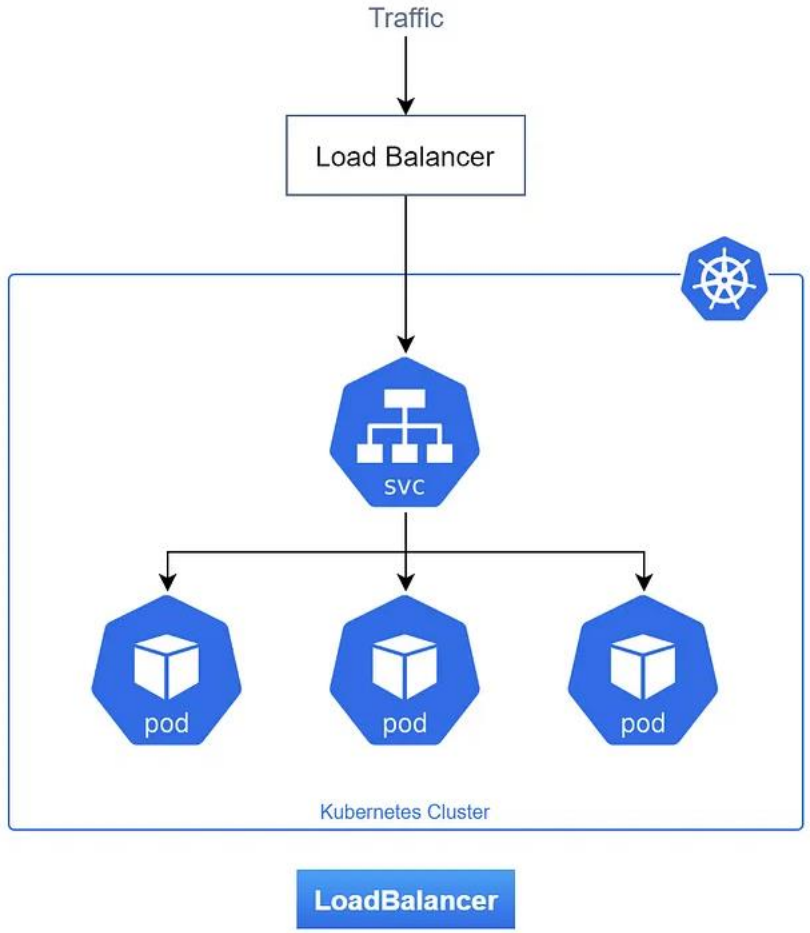
- Ta có thể tạo NodePort bằng cách tạo 1 file yaml như ví dụ bên:

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-frontend-service
spec:
  type: NodePort
  selector:
    app: web
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 8080
    nodePort: 30000 # 30000-32767, Optional field
```

Các loại Service

3.LoadBalancer

- Service LoadBalancer được sử dụng khi chúng ta muốn có một địa chỉ IP duy nhất để chuyển tiếp tất cả request đến service
- Service LoadBalancer sẽ tạo ra service NodePort và gửi một thông báo tới nhà cung cấp dịch vụ Kubernetes yêu cầu loadbalancer được thiết lập trỏ đến tất cả các node IP bên ngoài và nodePort cụ thể.
- Service LoadBalancer sẽ khả dụng khi nhà cung cấp dịch vụ Kubernetes hỗ trợ thiết lập bộ cân bằng tải bên ngoài (external load balancers), nếu không có thì LoadBalancer sẽ tương tự như NodePort.



Kubernetes Network - Service Discovery

- Kubernetes sẽ hỗ trợ hai chế độ cho service discovery là environment variables và DNS

Environment variables:

- Khi một Pod được chạy trên một Node, kubelet sẽ thêm một tập hợp các biến môi trường cho từng Dịch vụ đang hoạt động.

DNS:

- Kubernetes sẽ tạo các DNS record cho các Pod và Service , ta có thể kết nối với các service thông qua DNS thay vì địa chỉ IP
- Ví dụ: nếu ta có một Service được gọi là my-service trong namespace của Kubernetes tên là: my-ns, thì Control plane và DNS Service sẽ tạo một bản ghi DNS là my-service.my-ns
    - Các Pod ở trong namespace my-ns có thể tìm thấy service thông qua tên của nó là my-services
- **Ưu điểm: Dễ dàng gọi đến các service trong cụm kubernetes mà không cần biết đến địa chỉ IP của chúng.**

Create a Service manifest file to expose a Deployment as a network service.

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  labels:
    app: nginx
spec:
  selector:
    labels: nginx
  ports:
  - port: 90
    targetPort: 80
```

```
tung@tung-ideapad-5-pro:~/kubernetes$ ls
hello-world.yaml  nginx-deploy.yaml  nginx-service.yaml
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl apply -f nginx-service.yaml
service/nginx-service created
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl get services
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP   10d
nginx-service   ClusterIP   10.100.187.129   <none>        90/TCP    12s
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl get depoyments
error: the server doesn't have a resource type "depoyments"
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   10/10   10           10          4d1h
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl get pods
NAME                                READY   STATUS             RESTARTS         AGE
hello-world-pod                     0/1     CrashLoopBackOff   66 (3m44s ago)   6d23h
nginx-deployment-66f8758855-4kk2k   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-67md8   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-9xnd2   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-g7glf   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-h2q4t   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-kfbvx   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-mmcng   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-rmpsw   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-tv8qm   1/1     Running            2 (121m ago)     11h
nginx-deployment-66f8758855-wc7k7   1/1     Running            2 (121m ago)     11h
```

Use kubectl port-forward

```
tung@tung-ideapad-5-pro:~/kubernetes$ kubectl port-forward -n default pods/nginx-deployment-66f8758855-4kk2k 8015:80
Forwarding from 127.0.0.1:8015 -> 80
Forwarding from [::1]:8015 -> 80
Handling connection for 8015
Handling connection for 8015
```