Load balancer and DNS on your own Kubernetes cluster

Set Up Your Kubernetes Cluster

First, ensure you have a running Kubernetes cluster. You can use tools like Minikube, K3s, or MicroK8s for a lightweight setup.

Install MetalLB

MetalLB is a popular choice for load balancing in bare-metal Kubernetes clusters. It allows you to create LoadBalancer services in environments that don’t have a cloud provider.

1. **Install MetalLB:

    ```
    kubectl apply -f
    https://raw.githubusercontent.com/metallb/metallb/v0.9.3/manifests/namespace.yaml
    kubectl apply -f
    https://raw.githubusercontent.com/metallb/metallb/v0.9.3/manifests/metallb.yaml
    ```

2. **Configure MetalLB:**

    Create a ConfigMap to define the IP address pool MetalLB can use:

    ```
    apiVersion: v1
    kind: ConfigMap
    metadata:
      namespace: metallb-system
      name: config
    data:
      config: |
        address-pools:
        - name: default
          protocol: layer2
          addresses:
          - 192.168.1.240-192.168.1.250
    ```

Apply the ConfigMap:

```
kubectl apply -f metallb-config.yaml
```

Set Up CoreDNS for Local DNS

CoreDNS can be used to manage DNS within your cluster.

1. **Deploy CoreDNS:**

   CoreDNS is usually deployed by default in Kubernetes clusters. If not, you can deploy it manually:

   ```
   kubectl apply -f https://storage.googleapis.com/kubernetes-the-hard-way/coredns.yaml
   ```

2. **Configure CoreDNS:**

   Edit the CoreDNS ConfigMap to add your custom DNS entries:

   ```
   kubectl edit configmap coredns -n kube-system
   ```

   Add a new zone for your local domain:

   ```
   mydomain.local:53 {
       errors
       cache 30
       forward . 8.8.8.8
   }
   ```

Create Ingress Resources

Ingress resources allow you to define how to route traffic to your services.

1. **Install an Ingress Controller:**

   NGINX is a common choice for an ingress controller:

   ```
   kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/cloud/deploy.yaml
   ```

2. **Create an Ingress Resource:**

   Define an ingress resource to route traffic to your service:

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: example-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - host: myapp.mydomain.local
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: my-service
            port:
              number: 80
```

Apply the ingress resource:

```
kubectl apply -f ingress.yaml
```

Update Local DNS Resolver

Finally, update your local DNS resolver to point to the CoreDNS server in your cluster. This can usually be done by editing the `/etc/hosts` file or configuring your router to use the CoreDNS IP.

By following these steps, you should be able to set up a load balancer and DNS on your Kubernetes cluster, allowing you to use personal DNS addresses locally with ingress resources to route traffic to your services[1][2][3].

- Kubernetes Load Balancer: Expert Guide With Examples
- GitHub - loadworks/dnslb: DNS-based Load Balancer for applications
- How to Create a Kubernetes Cluster and Load Balancer for Local