MetalLB Installation Guide for Kubernetes

MetalLB is a load balancer implementation for bare metal Kubernetes clusters, providing network load balancing services that are typically only available in cloud environments.

Prerequisites

- Kubernetes cluster (version 1.13.0 or later)
- kubectl configured to access your cluster
- Admin privileges on the cluster
- Available IP address range for load balancer services

Method 1: Installation using Helm (Recommended)

Step 1: Add Helm Repository

bash

```
# Add MetalLB Helm repository
helm repo add metallb https://metallb.github.io/metallb

# Update Helm repositories
helm repo update
```

Step 2: Install MetalLB

bash

```
# Install MetalLB in metallb-system namespace
helm install metallb metallb/metallb \
  --namespace metallb-system \
  --create-namespace

# Verify installation
kubectl get pods -n metallb-system
kubectl get services -n metallb-system
```

Step 3: Configure MetalLB with Helm Values

Create a `metallb-values.yaml` file:

yaml

```yaml
# metallb-values.yaml
controller:
  image:
    repository: quay.io/metallb/controller
    tag: v0.13.12

  resources:
    limits:
      cpu: 100m
      memory: 100Mi
    requests:
      cpu: 100m
      memory: 100Mi

speaker:
  image:
    repository: quay.io/metallb/speaker
    tag: v0.13.12

  resources:
    limits:
      cpu: 100m
      memory: 100Mi
    requests:
      cpu: 100m
      memory: 100Mi

# Enable Prometheus metrics
prometheus:
  scrapeAnnotations: true
  metricsPort: 7472

# Node selector for speaker (optional)
speaker:
  nodeSelector:
    kubernetes.io/os: linux
```

Install with custom values:

bash

```
helm install metallb metallb/metallb \
  --namespace metallb-system \
  --create-namespace \
  --values metallb-values.yaml
```

Method 2: Installation using kubectl (YAML Manifests)

Step 1: Install MetalLB Components

bash

```
# Apply MetalLB manifests
kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.13.12/config/mani

# Verify installation
kubectl get pods -n metallb-system
kubectl get daemonset -n metallb-system
```

Step 2: Wait for Pods to be Ready

bash

```
# Wait for all pods to be ready
kubectl wait --namespace metallb-system \
  --for=condition=ready pod \
  --selector=app=metallb \
  --timeout=90s
```

Configuration

Layer 2 Configuration (Most Common)

Create an IPAddressPool and L2Advertisement:

yaml

```yaml
# metallb-config.yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: first-pool
  namespace: metallb-system
spec:
  addresses:
  - 192.168.1.240-192.168.1.250  # Replace with your IP range
  # Or use CIDR notation:
  # - 192.168.1.240/28
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: example
  namespace: metallb-system
spec:
  ipAddressPools:
  - first-pool
```

Apply the configuration:

bash

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

BGP Configuration (Advanced)

For BGP mode, create BGP configuration:

yaml

```yaml
# metallb-bgp-config.yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: production
  namespace: metallb-system
spec:
  addresses:
  - 192.168.10.0/24
---
apiVersion: metallb.io/v1beta2
kind: BGPPeer
metadata:
  name: sample
  namespace: metallb-system
spec:
  myASN: 64500
  peerASN: 64501
  peerAddress: 10.0.0.1
---
apiVersion: metallb.io/v1beta1
kind: BGPAdvertisement
metadata:
  name: example
  namespace: metallb-system
spec:
  ipAddressPools:
  - production
  peers:
  - sample
```

Apply BGP configuration:

bash

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

Testing the Installation

Create a Test Service

yaml

```yaml
# test-service.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector:
    app: nginx
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: LoadBalancer
```

Apply and test:

bash

```bash
# Apply test service
kubectl apply -f test-service.yaml

# Check if external IP is assigned
kubectl get service nginx-service

# Test the service (replace with assigned IP)
curl http://<EXTERNAL-IP>
```

Advanced Configuration

Multiple IP Pools

yaml

```yaml
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: production
  namespace: metallb-system
spec:
  addresses:
  - 192.168.1.240-192.168.1.250
---
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: development
  namespace: metallb-system
spec:
  addresses:
  - 192.168.2.240-192.168.2.250
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: production-advertisement
  namespace: metallb-system
spec:
  ipAddressPools:
  - production
  nodeSelectors:
  - matchLabels:
      environment: production
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: development-advertisement
  namespace: metallb-system
spec:
  ipAddressPools:
  - development
  nodeSelectors:
  - matchLabels:
      environment: development
```

Service-Specific Pool Selection

yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  annotations:
    metallb.universe.tf/address-pool: production
spec:
  selector:
    app: nginx
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
  type: LoadBalancer
```

Shared IP Addresses

yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: service-1
  annotations:
    metallb.universe.tf/allow-shared-ip: "shared-ip-key"
spec:
  selector:
    app: app-1
  ports:
  - protocol: TCP
    port: 80
    targetPort: 8080
  type: LoadBalancer
---
apiVersion: v1
kind: Service
metadata:
  name: service-2
  annotations:
    metallb.universe.tf/allow-shared-ip: "shared-ip-key"
spec:
  selector:
    app: app-2
  ports:
  - protocol: TCP
    port: 8080
    targetPort: 8080
  type: LoadBalancer
```

Monitoring and Troubleshooting

Check MetalLB Status

bash

```
# Check controller logs
kubectl logs -n metallb-system -l app=metallb,component=controller

# Check speaker logs
kubectl logs -n metallb-system -l app=metallb,component=speaker

# Check configuration
kubectl get ipaddresspool -n metallb-system
kubectl get l2advertisement -n metallb-system
kubectl get bgpadvertisement -n metallb-system
kubectl get bgppeer -n metallb-system

# Check service status
kubectl get services --all-namespaces -o wide
```

Common Issues and Solutions

1. **No External IP Assigned**

   bash

   ```
   # Check if IPAddressPool is configured
   kubectl get ipaddresspool -n metallb-system

   # Check if L2Advertisement exists
   kubectl get l2advertisement -n metallb-system

   # Check controller logs
   kubectl logs -n metallb-system -l component=controller
   ```

2. **External IP Not Reachable**

   bash

   ```
   # Check speaker logs on specific node
   kubectl logs -n metallb-system -l component=speaker --field-selector spec.nodeName=

   # Verify IP range is correct
   kubectl describe ipaddresspool -n metallb-system
   ```

3. **BGP Issues**

bash

```
# Check BGP peer status
kubectl get bgppeer -n metallb-system -o yaml

# Check BGP advertisements
kubectl get bgpadvertisement -n metallb-system -o yaml
```

Security Considerations

RBAC Configuration

yaml

```yaml
# metallb-rbac.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: metallb-controller
  namespace: metallb-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: metallb-controller
rules:
- apiGroups: [""]
  resources: ["services"]
  verbs: ["get", "list", "watch", "update"]
- apiGroups: [""]
  resources: ["events"]
  verbs: ["create", "patch"]
- apiGroups: ["metallb.io"]
  resources: ["ipaddresspools", "l2advertisements", "bgpadvertisements", "bgppeers"]
  verbs: ["get", "list", "watch"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: metallb-controller
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: metallb-controller
subjects:
- kind: ServiceAccount
  name: metallb-controller
  namespace: metallb-system
```

Network Policies

yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: metallb-controller
  namespace: metallb-system
spec:
  podSelector:
    matchLabels:
      app: metallb
      component: controller
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - namespaceSelector: {}
  egress:
  - to: []
```

Upgrade MetalLB

Using Helm

bash

```bash
# Update Helm repository
helm repo update

# Upgrade MetalLB
helm upgrade metallb metallb/metallb -n metallb-system

# Check upgrade status
kubectl rollout status deployment/metallb-controller -n metallb-system
```

Using kubectl

bash

```bash
# Apply new version
kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.13.12/config/mani

# Check rollout status
kubectl rollout status daemonset/metallb-speaker -n metallb-system
```

Uninstalling MetalLB

Using Helm

bash

```
helm uninstall metallb -n metallb-system
kubectl delete namespace metallb-system
```

Using kubectl

bash

```
kubectl delete -f https://raw.githubusercontent.com/metallb/metallb/v0.13.12/config/mar
```

This guide provides comprehensive installation and configuration options for MetalLB in your Kubernetes cluster, enabling load balancer services in bare metal environments.