Kubernetes Service Deployment using Helm Chart

Helm Chart : A Helm chart is a package that contains all the resources needed to deploy an application to a Kubernetes cluster. Basically we can define as a package manager for kubernetes.

Steps to Create the Helm Chart

Create the Helm Chart: To create a Helm chart, run the following command:

```
helm create <repo-name>
```

Example:

```
helm create nginx
```

Navigate to the Chart Directory: Once the chart is created, navigate to the chart directory:

```
cd nginx
```

1. Open this directory in your preferred text editor to modify the templates.
2. **Modify the Templates:** Modify the YAML template files inside the Helm chart to define the Kubernetes resources required for the Nginx deployment.

Configuration Breakdown

1. `values.yaml`

The `values.yaml` file defines the configurable parameters for the Helm chart. Key settings include:

Replica Count and Deployment Name:

```
replicaCount: 1
deploymentName: test-nginx-helm
namespace: nginx
```

Nginx Image Configuration: The image repository is set to `nginx`, and the `IfNotPresent` pull policy ensures that the image is pulled only if it isn't already available on the node.

```
image:
  repository: nginx
  tag: latest
  pullPolicy: IfNotPresent
```

Service Configuration: The service is of type `ClusterIP` and exposes port 80.

```
service:
  type: ClusterIP
  port: 80
```

Ingress Configuration: The ingress is configured with AWS NLB annotations, enabling secure HTTPS traffic using TLS with a self-signed certificate.

```
ingress:
 ingressName: nginx-nlb-ingress
 enabled: true
 className: nlb
 annotations:
   kubernetes.io/ingress.class: "nlb"
   service.beta.kubernetes.io/aws-load-balancer-type: "nlb"
   nginx.ingress.kubernetes.io/rewrite-target: /
   nginx.ingress.kubernetes.io/ssl-redirect: 'true'
   service.beta.kubernetes.io/aws-load-balancer-proxy-protocol: "*"
   service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
   service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
   service.beta.kubernetes.io/aws-load-balancer-backend-protocol: tcp
   acme.cert-manager.io/http01-edit-in-place: 'true'
   cert-manager.io/cluster-issuer: letsencrypt-prod
   kubernetes.io/tls-acme: 'true'
 hosts:
   - host: test-nginx.example.com
     paths:
       - path: /
         pathType: prefix
 tls:
  - secretName: chart-example-tls
    hosts:
      - test-nginx.example.com
```

Horizontal Pod Autoscaling (HPA): HPA is enabled to adjust the number of replicas based on CPU utilization (with a target of 70%).

```
autoscaling:
  enabled: true
  minReplicas: 1
  maxReplicas: 3
  targetCPUUtilizationPercentage: 70
```

Resource Allocation: Requests and limits for CPU and memory resources are defined.

```
resources:
  requests:
    memory: "128Mi"
    cpu: "100m"
  limits:
    memory: "128Mi"
    cpu: "100m"
```

Kubernetes Resource Templates

The Helm chart defines several Kubernetes resources, which are dynamically populated from the `values.yaml` file:

Namespace: `namespace.yaml`

This file defines the Kubernetes namespace for the deployment:

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: "{{ .Values.namespace }}"
```

Deployment: `deployment.yaml`

This file defines the Nginx deployment, which is tied to the values specified in `values.yaml`. If autoscaling is enabled, the replica count will adjust dynamically. Otherwise, it will use the static `replicaCount`.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: {{ .Values.deploymentName }}
 namespace: {{ .Values.namespace }}
 labels:
   {{- include "nginx.labels" . | nindent 4 }}
spec:
 {{- if not .Values.autoscaling.enabled }}
 replicas: {{ .Values.replicaCount }}
 {{- end }}
 selector:
   matchLabels:
     {{- include "nginx.selectorLabels" . | nindent 6 }}
 template:
   metadata:
     {{- with .Values.podAnnotations }}
     annotations:
       {{- toYaml . | nindent 8 }}
     {{- end }}
     labels:
       {{- include "nginx.labels" . | nindent 8 }}
       {{- with .Values.podLabels }}
       {{- toYaml . | nindent 8 }}
```

```
      {{- end }}
  spec:
    containers:
      - name: {{ .Chart.Name }}
        image: "{{ .Values.image.repository }}:{{ .Values.image.tag |
default .Chart.AppVersion }}"
        ports:
          - name: http
            containerPort: {{ .Values.service.port }}
            protocol: TCP
        resources:
          {{- toYaml .Values.resources | nindent 12 }}
```

Service: `service.yaml`

This file defines a `ClusterIP` service that exposes Nginx to the cluster on port 80.

```
apiVersion: v1
kind: Service
metadata:
 name: {{ .Values.depServiceName }}
 namespace: {{ .Values.namespace }}
 labels:
   {{- include "nginx.labels" . | nindent 4 }}
spec:
 type: {{ .Values.service.type }}
 ports:
   - port: {{ .Values.service.port }}
     targetPort: http
     protocol: TCP
     name: http
 selector:
   {{- include "nginx.selectorLabels" . | nindent 4 }}
```

Horizontal Pod Autoscaler (HPA): `hpa.yaml`

If autoscaling is enabled, the HPA adjusts the number of replicas based on CPU utilization.

```
{{- if .Values.autoscaling.enabled }}
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
 name: {{ .Values.hpaName }}
 namespace: {{ .Values.namespace }}
 labels:
   {{- include "nginx.labels" . | nindent 4 }}
spec:
 scaleTargetRef:
   apiVersion: apps/v1
   kind: Deployment
   name: {{ .Values.deploymentName }}
 minReplicas: {{ .Values.autoscaling.minReplicas }}
 maxReplicas: {{ .Values.autoscaling.maxReplicas }}
 metrics:
   {{- if .Values.autoscaling.targetCPUUtilizationPercentage }}
   - type: Resource
     resource:
       name: cpu
       target:
         type: Utilization
         averageUtilization: {{
.Values.autoscaling.targetCPUUtilizationPercentage }}
   {{- end }}
{{- end }}
```

Ingress: `ingress.yaml`

This file configures the Nginx ingress with AWS NLB annotations and enables TLS using a secret for secure HTTPS traffic.

```
{{- if .Values.ingress.enabled -}}
{{- $fullName := include "nginx.fullname" . -}}
{{- $svcPort := .Values.service.port -}}
{{- if and .Values.ingress.className (not (semverCompare ">=1.18-0"
.Capabilities.KubeVersion.GitVersion)) }}
 {{- if not (hasKey .Values.ingress.annotations
"kubernetes.io/ingress.class") }}
 {{- $_ := set .Values.ingress.annotations "kubernetes.io/ingress.class"
.Values.ingress.className}}
 {{- end }}
{{- end }}
{{- if semverCompare ">=1.19-0" .Capabilities.KubeVersion.GitVersion -}}
apiVersion: networking.k8s.io/v1
{{- else if semverCompare ">=1.14-0" .Capabilities.KubeVersion.GitVersion
-}}
apiVersion: networking.k8s.io/v1beta1
{{- else -}}
apiVersion: extensions/v1beta1
{{- end }}
kind: Ingress
metadata:
 name: {{ .Values.ingress.ingressName }}
 namespace: {{ .Values.namespace }}
 labels:
   {{- include "nginx.labels" . | nindent 4 }}
 {{- with .Values.ingress.annotations }}
 annotations:
   {{- toYaml . | nindent 4 }}
 {{- end }}
spec:
 {{- if and .Values.ingress.className (semverCompare ">=1.18-0"
.Capabilities.KubeVersion.GitVersion) }}
 ingressClassName: {{ .Values.ingress.className }}
 {{- end }}
```

```
 {{- if .Values.ingress.tls }}
 tls:
   {{- range .Values.ingress.tls }}
   - hosts:
       {{- range .hosts }}
       - {{ . | quote }}
       {{- end }}
     secretName: {{ .secretName }}
   {{- end }}
 {{- end }}
 rules:
   {{- range .Values.ingress.hosts }}
   - host: {{ .host | quote }}
     http:
       paths:
         {{- range .paths }}
         - path: {{ .path }}
           {{- if and .pathType (semverCompare ">=1.18-0"
$.Capabilities.KubeVersion.GitVersion) }}
           pathType: {{ .pathType }}
           {{- end }}
           backend:
             {{- if semverCompare ">=1.19-0"
$.Capabilities.KubeVersion.GitVersion }}
             service:
               name: {{ .Values.depServiceName }}
               port:
                 number: {{ .Values.service.port }}
             {{- else }}
             serviceName: {{ $fullName }}
             servicePort: {{ $svcPort }}
             {{- end }}
         {{- end }}
   {{- end }}
{{- end }}
```

AWS Load Balancer Controller Integration

To integrate the AWS Load Balancer with the Nginx ingress, the **AWS Load Balancer Controller** should be deployed in the Kubernetes cluster. This controller automatically provisions and manages an NLB, ensuring that traffic is routed to the Nginx service according to the ingress rules.

Annotations for NLB Integration:

- **`service.beta.kubernetes.io/aws-load-balancer-type: "nlb"`**: Specifies that the load balancer type should be an NLB.
- **`service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing`:** Configures the NLB to be internet-facing.
- **`service.beta.kubernetes.io/aws-load-balancer-backend-protocol: tcp`**: Defines the backend protocol for the NLB as TCP.
- **`service.beta.kubernetes.io/aws-load-balancer-proxy-protocol: "*"`**: Enables proxy protocol support for the NLB.

To install the AWS Load Balancer Controller via Helm, use the following command:

```
helm install aws-load-balancer-controller
eks/aws-load-balancer-controller
```

This controller will automatically create and manage the NLB and ensure that traffic is routed to the Nginx service as defined in the ingress configuration.

To install helm chart from local helm directory

helm install nginx-test ./nginx

Explanation:

- **`helm install`**: This command is used to install a Helm chart.
- **`nginx-test`**: This is the name you want to assign to the release. It is the name that will be used to reference the deployed chart.
- **`./nginx`**: This specifies the path to the chart. In this case, `nginx` is the chart directory located in the current directory (denoted by `./`).

Breakdown:

- **`nginx-test`**: The name of the Helm release.
- **`./nginx`**: Refers to the `nginx` chart, located in the current directory.

This command installs the Helm chart located in the `nginx/` directory and names the release `nginx-test`. The `./` indicates that the chart is in the current working directory.

To check the chart is install or not

helm list -n <namespace>