Prometheus and Grafana using Helm

Introduction to Helm

Helm is a powerful package manager for Kubernetes, often referred to as the "Kubernetes package manager." It streamlines the process of deploying, managing, and scaling applications within a Kubernetes cluster by using predefined packages known as "charts."

What is Helm?

Helm simplifies the deployment and management of applications in Kubernetes by providing the following features:

1. Packaging:

- Helm uses charts to package Kubernetes applications. A chart is a collection of files that describe a related set of Kubernetes resources.

2. Versioning:

- Helm charts support versioning, making it easy to manage and deploy specific versions of applications.

3. Dependency Management:

- Helm charts can declare dependencies on other charts, making it easy to manage complex applications composed of multiple components.

4. Configuration Management:

- Helm allows users to override default chart configurations with custom values, enabling customized deployments.

5. Release Management:

- Helm uses the concept of releases to manage deployments. A release is a specific instance of a chart running in a Kubernetes cluster.

6. Rollbacks:

- Helm provides the capability to roll back to previous releases, facilitating easy recovery from failed updates or deployments.

Key Components of Helm

1. Helm Client:

- The command-line tool that users interact with to create, install, and manage Helm charts and releases.

2. Helm Chart:

- A collection of files that describe a set of Kubernetes resources required to run an application. It includes templates, configuration files, and metadata.

3. Helm Repository:

- A collection of charts that can be shared and distributed. Helm repositories can be public or private.

4. Helm Tiller (Helm v2 only):

- A server component that runs inside the Kubernetes cluster. Tiller has been removed in Helm v3, making Helm more secure by eliminating the need for a cluster-wide server.

Download Helm 3 Installation Script

```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
```

Provide Execution Permission to the Script

```
sudo chmod 700 get_helm.sh
```

Execute the Script to Install Helm 3

```
sudo ./get_helm.sh
```

Verify Helm Installation

helm version --client

Add Helm Repositories

```
helm repo add stable https://charts.helm.sh/stable
```

Add Prometheus Helm repo

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm search repo prometheus-community
```

Note:

- Prometheus and Grafana Helm charts have moved to the kube-prometheus-stack.

Install and Configure Prometheus and Grafana

Create Prometheus Namespace

```
kubectl create namespace prometheus
```

Install kube-prometheus-stack

The kube-prometheus-stack comes with a Grafana deployment embedded.

```
helm install stable prometheus-community/kube-prometheus-stack -n prometheus
```

Verify Prometheus and Grafana Installation

```
kubectl get pods -n prometheus
kubectl get svc -n prometheus
```

- This confirms that Prometheus and Grafana have been installed successfully using Helm.

Expose Prometheus and Grafana Outside the Cluster

To make Prometheus and Grafana available outside the cluster, use LoadBalancer or NodePort instead of ClusterIP.

Edit Prometheus and Grafana Services to Use LoadBalancer

```
kubectl patch svc stable-kube-prometheus-sta-prometheus -n prometheus -p '{"spec": {"type": "LoadBalancer"}}'
kubectl patch svc stable-grafana -n prometheus -p '{"spec": {"type": "LoadBalancer"}}'
```

Alternatively, Manually Edit the Services

```
kubectl edit svc stable-kube-prometheus-sta-prometheus -n prometheus
kubectl edit svc stable-grafana -n prometheus
```

Retrieve Grafana Admin Password

```
kubectl get secret --namespace prometheus stable-grafana -o jsonpath="{.data.admin-password}" | base64 --decode ; echo
```

Verify Service Type and Get Load Balancer URL

```
kubectl get svc -n prometheus
```

Access Prometheus and Grafana

- Check the LoadBalancer URL in your browser:
  - Prometheus: http://<LoadBalancer_URL>:9090
  - Grafana: http://<LoadBalancer_URL>:80