Helm chart - Wordpress Installation with MariaDB on Kubernetes

Nov 26, 2020 · 10 min read · HELM CHART

· Last Modified : Nov 26, 2020 Share on: Author : Rahul Wagh

Wordpress Helm Chart Installation

In this tutorial, we are going to install WordPress with MariaDB using the Helm Chart on Kubernetes cluster. With this installation, we are going to see - How we can upgrade as well as rollback the Helm Chart release of WordPress. This complete setup inherited the benefits of the Kubernetes .i.e. scalability and availability.

Since we are installing WordPress, so we need to have a database running behind the WordPress application. From the database standpoint, we are going to use MariaDB. Helm chart ships all these components in a single package, so that we need not worry about installing each component separately.

Throughout this session, we are going to use bitnami repo.

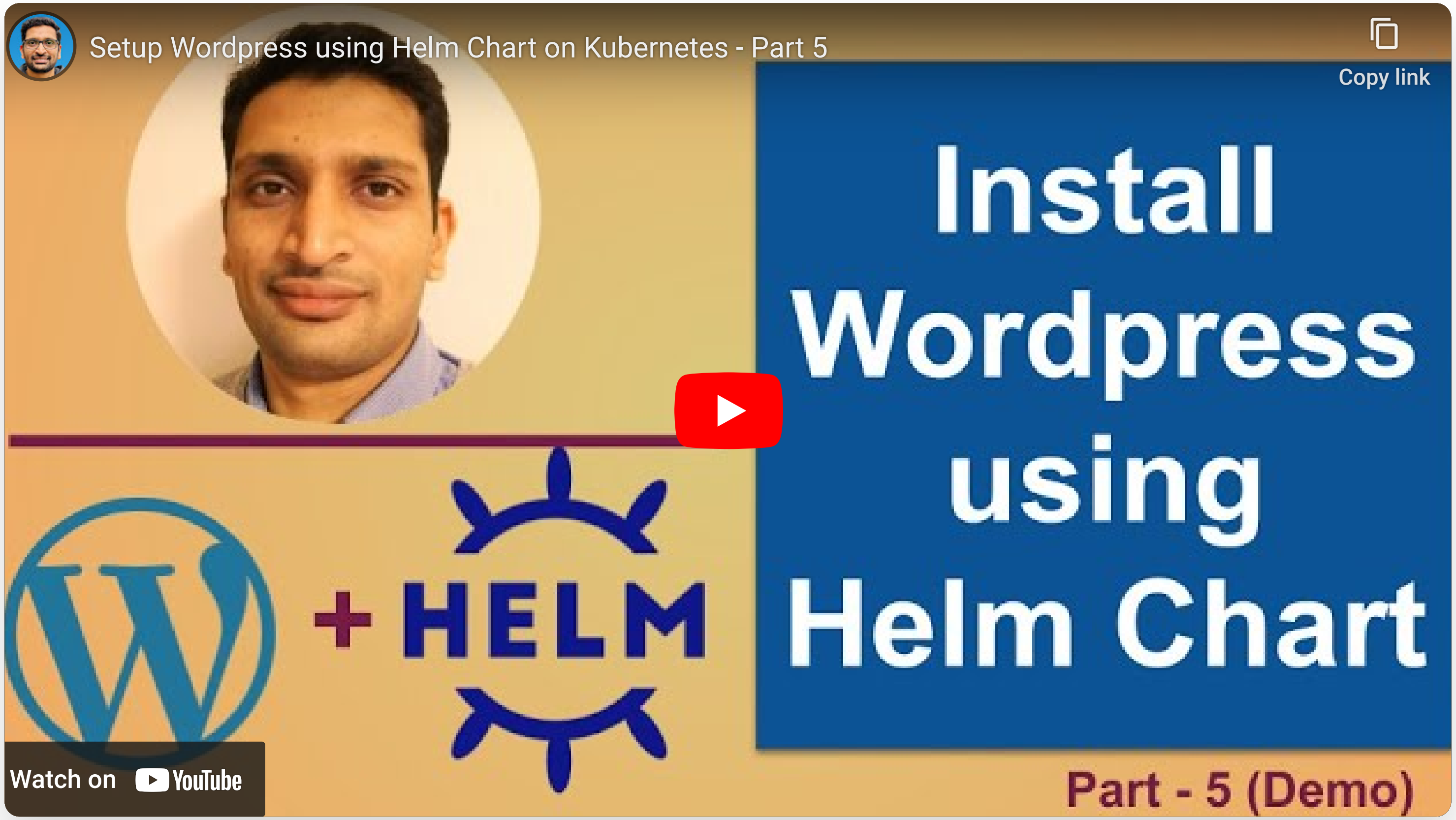
Table of Content

Search...

Categories

TERRAFORM 63 DOCKER 28
KUBERNETES 26 AWS 13
ANSIBLE 11 HELM-CHART 11
BLOGGING 6 SSL 6
SPRING-BOOT 5 QUARKUS 4
GITHUB 3 KUBESPRAY 3
PROMETHEUS-GRAFANA 3
VAGRANT 3
ALL CATEGORIES

Series



Setup Wordpress using Helm Chart on Kubernetes - Part 5
Copy link
Install Wordpress using Helm Chart
+ HELM
Part - 5 (Demo)
Watch on YouTube

1. Prerequisites

TERRAFORM	61
DOCKER	28
KUBERNETES	27
ANSIBLE	11
HELM-CHART	11
AWS	2
JENKINS	2
LINUX-COMMANDS	1

Tags

KUBERNETES	18
HELM-CHART	10
BLOGGING	4
QUARKUS	4
DOCKER	3
GITHUB	3
SSL	3
KUBESPRAY	2
SPRING-BOOT	2
ANSIBLE	1
HADOOP	1
INDEX	1
NGINX	1
TERRAFORM	1

Recent Posts

- Ansible Handlers Explained Real-World Use Cases & Examples
- Securing Sensitive Data in Terraform
- Boost Your AWS Security with Terraform : A Step-by-Step Guide
- How to Load Input Data from a File in Terraform?

Before we begin there are certain minimum setup requirements which we need to fulfill -

1. Kubernetes Cluster - v1.12+
2. Helm 3.0+
3. Persistent Volume with Dynamic Volume Provisioning (Click here for more details)
4. Cloud Service .i.e. - GCP, AWS, DigitalOcean, Openshift

> **Note 1.** - This complete setup cannot be performed on the Kubernetes cluster running on a virtual machine on your desktop or laptop. Since WordPress is a multitier application with a Database, so it needs `Dynamic Volume Provisioning` and it cannot be done on a local virtual machine.
>
> **Note 2.** - We are going to use Google Cloud Platform for this setup. But the same steps can be followed on any cloud platform.

2. Setup up Kubernetes cluster on GCP (Google Cloud Platform)

If you are a newbie who just started learning Kubernetes then do not worry Google Cloud platform provides you 300$ credit for 1 Year which you can use for this setup.

2.1 Login the Google Cloud Platform

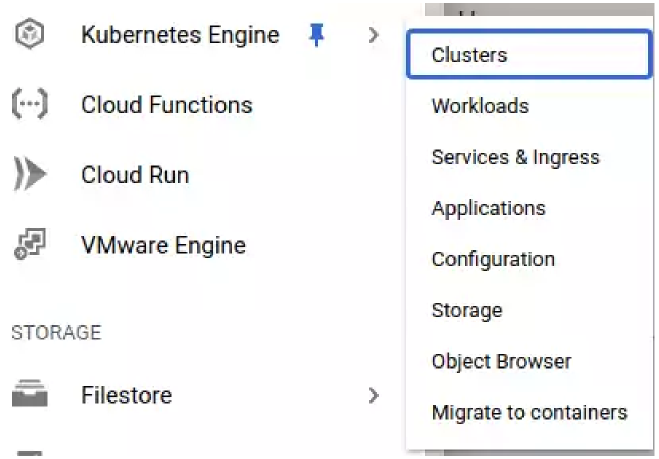
The first step would be to log in and after the login goto the left navigation menu and look for the `Compute` section and under that look for `Kubernetes Engine->Clusters`

- Can Terraform be used to provision on-premises infrastructure?
- Fixing the Terraform Error creating IAM Role. MalformedPolicyDocument Has prohibited field Resource
- In terraform how to handle null value with default value?
- Terraform use module output variables as inputs for another module?

Rahul Wagh

Its all about Open Source and DevOps, here I talk about Kubernetes, Docker, Java, Spring boot and practices.

READ MORE



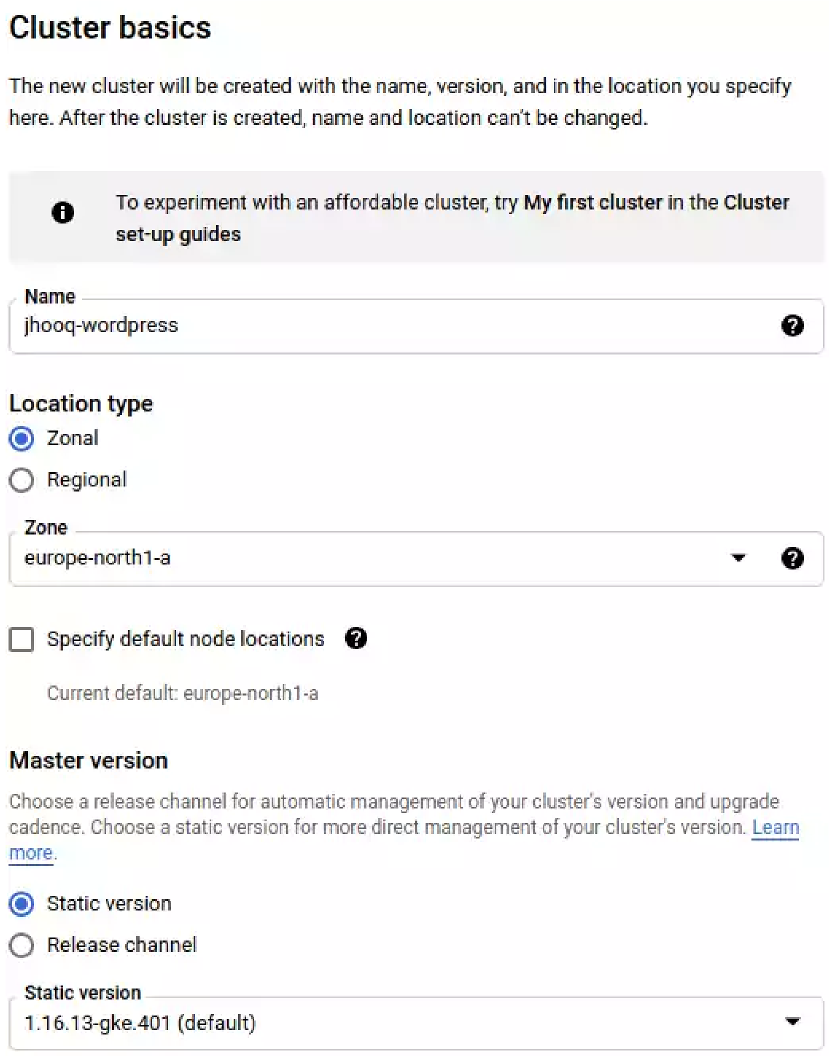
Google Cloud kubernetes engine

2.2 Fill in the Cluster Details

After selecting the `Kubernetes Engine->Clusters` option in the previous step now you need to fill in the details of the clusters

	Field	Value
1	Name	jhooq-wordpress
2	Zone	europe-north1-a
3	Master Version	Static Version 1.16.13-gke.401(default)

Refer to the following screenshot -



Cluster basics

The new cluster will be created with the name, version, and in the location you specify here. After the cluster is created, name and location can't be changed.

To experiment with an affordable cluster, try **My first cluster** in the **Cluster set-up guides**

Name: jhooq-wordpress

Location type

● Zonal

○ Regional

Zone: europe-north1-a

□ Specify default node locations

Current default: europe-north1-a

Master version

Choose a release channel for automatic management of your cluster's version and upgrade cadence. Choose a static version for more direct management of your cluster's version. Learn more.

● Static version

○ Release channel

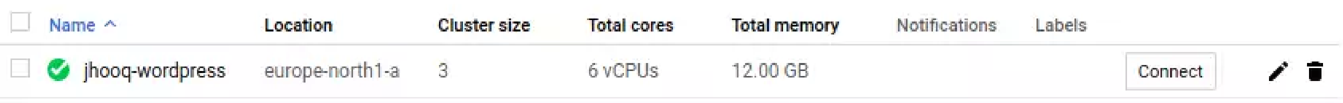
Static version: 1.16.13-gke.401 (default)

Cluster details form

After filling in the values click on create and your cluster should be ready in couple of minutes.

2.3 Connect to Kubernetes Cluster

Now once your Kubernetes cluster is ready you can click on `Connect` to connect your Kubernetes cluster.



□	Name ^	Location	Cluster size	Total cores	Total memory	Notifications	Labels		
□	✓ jhooq-wordpress	europe-north1-a	3	6 vCPUs	12.00 GB			Connect	✎ 🗑

Connect to Kubernetes cluster

After you click on the `Connect`, you will be prompted with the option to `Connect to the cluster`. Go and choose `Command-line access -> Run in Cloud Shell`

But copy the command before you click on `Run in Cloud Shell`.

Here is my command which I got -

```bash
gcloud container clusters get-credentials jhooq-wordpress --zone europe-north1-a --project jhooq-
```

The above command needs three parameters -

1. Cluster name - `jhooq-wordpress`
2. Zone - `europe-north1-a`
3. Project - `jhooq-sprinboot-k8s-demo`

Note - In a single project you can create multiple clusters.

2.4 Authorize Cloud Shell

After you click on `Run in Cloud Shell` there will be another prompt for `Authorize Cloud Shell`. Go ahead and click on `Authorize` and it should connect you to the kubernetes cluster

Authorize Cloud Shell

2.5 Verify the Kubernetes Cluster setup

The last step in the cluster setup would be to verify the cluster.

Run the following command to verify the cluster -

```bash
kubectl get nodes
```

The command should return you with -

```bash
NAME                                          STATUS   ROLES    AGE   VERSION
gke-jhooq-wordpress-default-pool-0e0e0b0b-85td   Ready    <none>   27m   v1.16.13-gke.401
gke-jhooq-wordpress-default-pool-0e0e0b0b-9h99   Ready    <none>   27m   v1.16.13-gke.401
gke-jhooq-wordpress-default-pool-0e0e0b0b-tzpz   Ready    <none>   27m   v1.16.13-gke.401
```

Now your Kubernetes cluster setup is complete on Google Cloud Platform

3. Add '/bitnami/wordpress' wordpress repo

The next step would be for you to add the `/bitnami/wordpress` repo to your helm installation.

3.1 Search for the 'wordpress' repo

First of all you need to check how many `wordpress` repositories are available on the Helm Hub.

Use the following command to search for the `wordpress` repositories.

```bash
helm search hub wordpress
```

After running the above command it should return you with the list of repos available on the Helm Hub.

```bash
URL                                                CHART VERSION  APP VERSION    DESCRIPTION
https://hub.helm.sh/charts/groundhog2k/wordpress   0.1.3          5.5.1-apache   A Helm chart fo
https://hub.helm.sh/charts/bitnami/wordpress       10.0.3         5.5.3          Web publishing
https://hub.helm.sh/charts/seccurecodebox/old-w... 2.1.0          4.0            Insecure & Outd
https://hub.helm.sh/charts/fasterbytecharts/wor... 0.8.4          v0.8.4         FasterBytes Wor
https://hub.helm.sh/charts/presslabs/wordpress-... 0.10.5         0.10.5         Presslabs WordP
https://hub.helm.sh/charts/presslabs/wordpress-... 0.10.3         v0.10.3        A Helm chart fo
https://hub.helm.sh/charts/fasterbytecharts/wor... 0.10.2         v0.10.2        A Helm chart fo
https://hub.helm.sh/charts/seccurecodebox/wpscan   2.1.0          latest         A Helm chart fo
https://hub.helm.sh/charts/presslabs/stack         0.10.3         v0.10.3        Open-Source Wor
https://hub.helm.sh/charts/fasterbytecharts/stack  0.10.2         v0.10.2        Open-Source Wor
```

If you look carefully at the results then we are interested in `https://hub.helm.sh/charts/bitnami/wordpress`.

In case if the URL is too long to see then you can put `--max-col-width=0`, so that you can view the complete URL

```bash
helm search hub wordpress  --max-col-width=0
```

3.2 Add 'bitnami/wordpress' to your repo list of Helm Chart

After knowing the `repo url` now you can add it to your local Helm Chart repo list.

But before adding the `bitnami/wordpress` first check whether it already exists on your repo list or not?

```bash
helm repo list
```

If you haven't added the `bitnami/wordpress` before then it should not show in the list.

Alright, let us add it to your repo list -

```bash
helm repo add bitnami https://charts.bitnami.com/bitnami
```

Once you add it successfully then you should see the following message.

```bash
"bitnami" has been added to your repositories
```

To know more about the details of the Bitnami WordPress Helm Chart - Click Here

3.3 Check Wordpress Version

Before we go into the installation step of the chart, let's check the version of the WordPress which we are going to install.

Run the following command to get all the available versions -

```bash
helm search repo wordpress --versions
```

It will return a long list of all the version available for WordPress

```bash
NAME                CHART VERSION  APP VERSION    DESCRIPTION
bitnami/wordpress  10.0.3         5.5.3          Web publishing platform for building blogs and .
bitnami/wordpress  10.0.2         5.5.3          Web publishing platform for building blogs and .
bitnami/wordpress  10.0.1         5.5.3          Web publishing platform for building blogs and .
bitnami/wordpress  9.10.0         5.5.3          Web publishing platform for building blogs and .
bitnami/wordpress  9.9.3          5.5.3          Web publishing platform for building blogs and .
```

We will go with the latest version which is `10.0.3`

3.4 Readme and Values

There are a few more details which are provided along with the helm chart package.

Readme.md

This Readme.md contains the installation instructions and it can be viewed using the following command

```bash
helm show readme bitnami/wordpress --version 10.0.3
```

Values

If you are familiar with WordPress before then you need `username` and `password` to access the WordPress CMS, so you can view the default values

```bash
helm show values bitnami/wordpress --version 10.0.3
```

Note - Here you will get a long list of values but you can skip this part because we are going to set up the username and password in the next step.

4. Setup User account along with Username and Password for WordPress

As Wordpress is CMS, so we need to have a user account to access it.

To achieve this we are gonna create a complete user account and store it in `wordpress-values.yaml`. (*This YAML config will be used later during the installation*)

Here is the list of values for user account -

	Field	Value
1	**wordpressUsername**	jhooq
2	**wordpressPassword**	jhooq
3	**wordpressEmail**	contact@jhooq.com
4	**wordpressFirstName**	Rahul
5	**wordpressLastName**	Wagh
6	**wordpressBlogName**	Jhooq.com
7	**service.type**	LoadBalancer

Here are steps for creating the `wordpress-values.yaml`

1. Create `wordpress-values.yaml`

```bash
touch wordpress-values.yaml
```

2. Open in `vi` mode to update the file

```bash
touch wordpress-values.yaml
```

3. Copy and paste the following values

```bash
wordpressUsername: jhooq
wordpressPassword: jhooq
wordpressEmail: contact@jhooq.com
wordpressFirstName: Rahul
wordpressLastName: Wagh
wordpressBlogName: Jhooq.com
service:
  type: LoadBalancer
```

4. Save and Exit the file

5. Install the WordPress helm chart

Now we have completed all the pre-requisites for the installation. Let's start installing the WordPress helm chart

5.1 Create a namespace - nswordpress

I would like to install WordPress in the fresh workspace so run the following command to create the workspace.

```bash
kubectl create namespace nswordpress
```

5.2 Verify the namespace

You can verify the workspace by listing it out all the available workspace -

```bash
kubectl get namespace
```

It should return you with the `nswordpress` in the list

```bash
NAME                 STATUS   AGE
default              Active   29d
kube-node-lease      Active   29d
kube-public          Active   29d
kube-system          Active   29d
nswordpress          Active   17s
```

5.3 Install wordpress helm chart

Run the following command for installation

BASH

```
helm install wordpress bitnami/wordpress --values=wordpress-values.yaml --namespace nswordpress -
```

Here is the break down of the command

wordpress installation command

Once you execute this command then it should return you with the following output

BASH

```
NAME: wordpress
LAST DEPLOYED: Mon Nov 23 19:39:36 2020
NAMESPACE: nswordpress
STATUS: deployed
REVISION: 1
NOTES:
** Please be patient while the chart is being deployed **

Your WordPress site can be accessed through the following DNS name from within your cluster:

    wordpress.nswordpress.svc.cluster.local (port 80)

To access your WordPress site from outside the cluster follow the steps below:

1. Get the WordPress URL by running these commands:

   export NODE_PORT=$(kubectl get --namespace nswordpress -o jsonpath="{.spec.ports[0].nodePort}
   export NODE_IP=$(kubectl get nodes --namespace nswordpress -o jsonpath="{.items[0].status.adc
   echo "WordPress URL: http://$NODE_IP:$NODE_PORT/"
   echo "WordPress Admin URL: http://$NODE_IP:$NODE_PORT/admin"

2. Open a browser and access WordPress using the obtained URL.

3. Login with the following credentials below to see your blog:

```

```bash
   echo Username: jhooq
   echo Password: $(kubectl get secret --namespace nswordpress wordpress -o jsonpath="{.data.worc
```

Do not worry we are gonna again break it down to understand it more -

5.4 How to find access URL for WordPress?

If you look at the previous output then here is a command which you need to execute from the output

```bash
export NODE_PORT=$(kubectl get --namespace nswordpress -o jsonpath="{.spec.ports[0].nodePort}" se
   export NODE_IP=$(kubectl get nodes --namespace nswordpress -o jsonpath="{.items[0].status.addr
   echo "WordPress URL: http://$NODE_IP:$NODE_PORT/"
```

It will return you with URL with IP address (This IP address can be different in your case)-

```bash
http://23.231.938.998
```

5.5 How to find access URL for WordPress Admin Portal?

Its also given in the previous output, so you just need to copy and execute the following command

```bash
echo "WordPress Admin URL: http://$NODE_IP:$NODE_PORT/admin"
```

It will return you with URL with IP address -

```bash
http://23.231.938.998/admin
```

And you can use the `username - jhooq` and `password - jhooq` for accessing the WordPress.

6. Troubleshooting any deployment and service inside the Kubernetes cluster

During the complete installation process if you run into any issue then I would recommend starting with the following steps for trouble shooting

6.1 Check all the Kubernetes resources status?

You can run the following command to know all the status of all the deployed resources inside the namespace - `nswordpress`

```bash
watch -x kubectl get all --namespace nswordpress
```

All the deployment and service status should be `Running`

7. Conclusion

Here is what we did

1. Setup the Kubernetes cluster on Google Cloud Platform
2. Added `/bitnami/wordpress` repo to helm repo list

3. Setup the user account for WordPress CMS
4. Performed the WordPress installation
5. Finally looked at the troubleshooting steps

> **Read More** -
>
> 1. Helm chart - How to Add/Install plugins
> 2. Getting started with Helm Chart
> 3. Helm chart - WordPress Installation with MariaDB on Kubernetes
> 4. Helm chart - Build you first helm chart with Spring Boot
> 5. Helm Chart - Convert Kubernetes YAML into Helm Chart YAML
> 6. Helm Chart - Pass environment variables
> 7. Helm Chart - Plugin
> 8. Helm Chart - Dry Run Install
> 9. Helm Chart - How to create multiple values files inside helm chart?
> 10. Helmfile - How to use Helmfile for managing helm chart?

Posts in this series

- How to use Helmfile for managing helm chart?
- How to create multiple values files inside helm chart?
- Pass environment variables into Helm Chart?
- How to fix - Helm install unknown flag --name/Error must either provide a name or specify --generate-name?
- Understanding Helm dry run for template debugging
- How to fix - Error create failed to create Secret invalid metadata.name Invalid value DNS-1123 subdomain must consist of lower case alphanumeric characters - or ., and must start and end with an alphanumeric character (e.g. example.com, regex used for validation is)

- Convert Kubernetes deployment YAML into Helm Chart YAML
- Helm chart - Wordpress Installation with MariaDB on Kubernetes
- Helm chart - How to Add/Install plugins
- Getting Started with Helm Chart
- Building first Helm Chart with Spring Boot Microservices