



MySQL/MariaDB

Project Astra

NetApp

October 18, 2020

This PDF was generated from <https://docs.netapp.com/us-en/project-astra/solutions/solutions/mariadb-deploy-from-helm-chart.html> on October 18, 2020. Always check docs.netapp.com for the latest.

Table of Contents

| | |
|--|---|
| MySQL/MariaDB | 1 |
| Deploy MariaDB From a Helm Chart | 1 |
| Deploy MySQL From a Helm Chart | 2 |
| Work With MySQL/MariaDB on Project Astra | 3 |

MySQL/MariaDB

Deploy MariaDB From a Helm Chart

Learn how to exercise the Project Astra Beta program workflow by deploying MariaDB from a Helm chart. After you deploy MariaDB on your cluster, you can manage the application with Project Astra.

MariaDB is a validated app for the Project Astra Beta program. [Learn the difference between Validated and Standard apps.](#)



The Project Astra Beta program only supports MySQL 0.3.22 and MariaDB 14.14.

System Requirements

In order to deploy MariaDB from a Helm chart for the Project Astra Beta program, you need the following:

- A GKE cluster which has been added to Project Astra.
- Updated versions of Helm (version 3.2+) and Kubectl installed.
- Kubeconfig configured using the gcloud tool with a command like `gcloud container clusters get-credentials my-cluster-name`

Install MariaDB

To exercise the Project Astra Beta program workflow, we recommend you use the Helm chart.

Deploy MariaDB with the command:

```
helm install mariadb bitnami/mariadb --namespace testdb --create-namespace --set db.database=test_db,db.user=test_db_user,db.password=NKhjs2wQPt8 > /dev/null 2>&1
```

This does the following:

- Creates the `testdb` namespace.
- Deploys MariaDB on the `testdb` namespace.
- Creates a database named `test_db`
- Creates a user `test_db_user` with password `NKhjs2wQPt8`



This method of setting the password at deployment is insecure. Only use this command when setting up MariaDB for a sandbox deployment to use Project Astra Beta program. We do not recommend this for a production environment.

After the Helm chart is deployed, it will be automatically discovered by Project Astra, at which point you can manage the app with Project Astra.

Deploy MySQL From a Helm Chart

Learn how to exercise the Project Astra Beta workflow by deploying MySQL from a Helm chart. After you deploy MySQL on your Kubernetes cluster, you can manage the application with Project Astra.

MariaDB and MySQL are validated apps for the Project Astra Beta program. [Learn the difference between Validated and Standard apps.](#)



The Project Astra beta program only supports MySQL 0.3.22 and MariaDB 14.14.

System Requirements

In order to deploy MySQL from a Helm chart for the Project Astra Beta program, you need the following:

- A GKE cluster which has been added to Project Astra.
- Updated versions of Helm (version 3.2+) and Kubectl installed.
- Kubeconfig configured using the gcloud tool with a command like `gcloud container clusters get-credentials my-cluster-name`



You must deploy your app after the cluster is added to Project Astra, not before.

Install MySQL

To exercise the Project Astra Beta workflow, we recommend the [standard stable chart](#).



You must deploy the Helm chart in a namespace other than the default.

Deploy MySQL with the command:

```
helm install mysql stable/mysql --namespace testdb--set
db.database=test_db,db.user=test_db_user,db.password=NKhjs2wQPt8
if you need to deploy mysql under a new namespace; please use the following command
helm install mysql stable/mysql --namespace testdb --create-namespace --set
db.database=test_db,db.user=test_db_user,db.password=NKhjs2wQPt8
```

This does the following:

- Creates the `testdb` namespace.
- Deploys MySQL on the `testdb` namespace.

- Creates a database named `test_db`
- Creates a user `test_db_user` with password `NKhjs2wQPt8`



This method of setting the password at deployment is insecure. You can use own secrets and config maps and passing along with helm command.

After the Helm chart is deployed, it will be automatically discovered by Project Astra, at which point you can manage the app with Project Astra.

Work With MySQL/MariaDB on Project Astra

This guide focuses on Helm as the preferred way to deploy Postgres apps. Plain YAML and Operator-based deployments may be covered in future guides.

For express instructions on launching MySQL/MariaDB on Project Astra, see [Deploy MySQL/MariaDB from a Helm Chart](#).

MariaDB and MySQL are validated apps for the Project Astra Beta program. [Learn the difference between Validated and Standard apps](#)



MySQL 0.3.22 and MariaDB 14.14 are the only versions supported in the Project Astra Beta program.

Requirements

In order to deploy MySQL/MariaDB from a Helm chart on a cluster registered with the Project Astra Beta program, you will need the following:

GKE Cluster

An up-to-date Kubernetes cluster (version 1.17+) which is connected to Project Astra. For help creating your GKE cluster and connecting it to Project Astra, see the [Getting Started Guide](#).

Kubectl

Kubectl is a standard tool for interacting with Kubernetes. For more information, see the guide [Install and Set Up kubectl](#) in the official Kubernetes documentation.

Kubeconfig

The Kubeconfig file contains the credentials which let kubectl communicate with your Kubernetes cluster. to learn how to download your GKE Kubeconfig file, see the Google Cloud guide for [configuring cluster access for kubectl](#).

Cloud Volume Service in Google Cloud Platform (CVS-GCP)

CVS is the storage layer and connective elements for Project Astra, respectively. More details on how to configure CVS on GCP may be found in the [workflow guide for CVS](#)[^].

Helm (v3)

Helm is a popular way to organize and install apps on Kubernetes. To install Helm on your local computer, follow [their handy install guide](#).

MySQL/MariaDB Requirements

For a MySQL/MariaDB application, Project Astra requires:

- `global.storageClass` value to be set to the storageClass representing either CVS or Trident (or, that storageClass is set as your cluster's default provisioner).

Install MariaDB/MySQL

For the Project Astra beta, we recommend the custom Helm chart we have created for this purpose. For instructions on how to deploy from this custom chart, see [Deploy MySQL/MariaDB from a Helm Chart](#).

The values need to be set to consume the volumes provisioned by CVS, be deployed in a namespace other than default, and your stateful app needs to be available to Project Astra.

By default the Bitnami chart uses a cluster's default storage class. Kubernetes clusters registered with project Astra beta use Trident CSI from NetApp. Trident automatically sets CVS as the default storage class. Use `kubectl get sc` to see what your cluster's storageClasses are. This produces output like the following:

| NAME | PROVISIONER | RECLAIMPOLICY | VOLUMEBINDINGMODE |
|------------------------------|-----------------------|---------------|-------------------|
| ALLOWVOLUMEEXPANSION | AGE | | |
| netapp-cvs-extreme | csi.trident.netapp.io | Delete | Immediate |
| true | 26h | | |
| netapp-cvs-premium (default) | csi.trident.netapp.io | Delete | Immediate |
| true | 26h | | |
| netapp-cvs-standard | csi.trident.netapp.io | Delete | Immediate |
| true | 26h | | |
| standard | kubernetes.io/gce-pd | Delete | Immediate |
| true | 27h | | |

You have two options for changing settings in your `values.yaml`. The first option is to open the file and edit it directly. The second option is to add an extra argument to your usual Helm CLI command.

To view and export `values.yaml`, use the `helm show` command:

```
# mariaDB
helm show values bitnami/mariadb
# mySQL
helm show values bitnami/mysql
```

or

```
# mariaDB
helm show values bitnami/mariadb > my-values.yaml
# mySQL
helm show values bitnami/mysql > my-values.yaml
```

This creates a `my-values.yaml` file in your local directory. That file is a copy of the official `values.yaml`.

Dry Run

Before deploying, you can do a dry run to make sure everything is set up correctly.

To do this, edit the values in the `my-values.yaml` file you created in the previous step. Test your deployment using the `-f my-values.yaml` and `--dry-run` flags:

```
# MariaDB
helm install -f my-values.yaml --namespace testdb --generate-name bitnami/mariadb --dry-run

# MySQL
helm install -f my-values.yaml --namespace testdb --generate-name bitnami/mysql --dry-run
```

If the output from our dry run looks correct, we may deploy to your cluster by removing `--dry-run`.

Before we can run the helm charts for real, you can choose to use an existing namespace or specify to create a new namespace with helm command like below;

```
# MariaDB
helm install -f my-values.yaml --namespace testdb --generate-name bitnami/mariadb
--create-namespace

# MySQL
helm install -f my-values.yaml --namespace testdb --generate-name bitnami/mysql --create-namespace
```

After deploying the application using Helm chart Project Astra will be automatically discover the

application. After a successful discovery you can manage the app with Project Astra.

Copyright Information

Copyright © 2020 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.