



Use Astra Control Service

Astra Control Service

NetApp
December 16, 2021

Table of Contents

- Use Astra Control Service 1
 - Log in to Astra Control Service..... 1
 - Manage and protect apps 1
 - View app and compute health 32
 - Manage buckets..... 36
 - Manage your account..... 39
 - Unmanage apps and clusters 47

Use Astra Control Service

Log in to Astra Control Service

Astra Control Service is accessible through a SaaS-based user interface by going to <https://astra.netapp.io>.



You can use single sign-on to log in using credentials from your corporate directory (federated identity). To learn more, go to the [Cloud Central Help Center](#) and then select **Cloud Central sign-in options**.

What you'll need

- A [Cloud Central user ID](#).
- A [new Astra Control account](#) or [an invitation to an existing account](#).
- A supported web browser.

Astra Control Service supports recent versions of Firefox, Safari, and Chrome with a minimum resolution of 1280 x 720.

Steps

1. Open a web browser and go to <https://astra.netapp.io>.
2. Log in using your NetApp Cloud Central credentials.

Manage and protect apps

Start managing apps

After you [add a Kubernetes cluster to Astra Control](#), you can install apps on the cluster (outside of Astra Control), and then go to the Apps page in Astra Control to start managing the apps.

App management requirements

Consider the following Astra Control app management requirements:

- **Licensing:** To manage apps using Astra Control Center, you need an Astra Control Center license.
- **Namespaces:** Astra Control requires that an app not span more than a single namespace, but a namespace can contain more than one app.
- **StorageClass:** If you install an app with a StorageClass explicitly set and you need to clone the app, the target cluster for the clone operation must have the originally specified StorageClass. Cloning an application with an explicitly set StorageClass to a cluster that does not have the same StorageClass will fail.
- **Kubernetes resources:** Apps that use Kubernetes Resources not collected by Astra Control might not have full app data management capabilities. Astra Control collects the following Kubernetes resources:
 - ClusterRole
 - ClusterRoleBinding
 - ConfigMap

- CustomResourceDefinition
- CustomResource
- DaemonSet
- Deployment
- DeploymentConfig
- Ingress
- MutatingWebhook
- PersistentVolumeClaim
- Pod
- ReplicaSet
- RoleBinding
- Role
- Route
- Secret
- Service
- ServiceAccount
- StatefulSet
- ValidatingWebhook

Supported app installation methods

You can use the following methods to install apps that you want to manage with Astra Control:

- **Manifest file:** Astra Control supports apps installed from a manifest file using kubectl. For example:

```
kubectl apply -f myapp.yaml
```

- **Helm 3:** If you use Helm to install apps, Astra Control requires Helm version 3. Managing and cloning apps installed with Helm 3 (or upgraded from Helm 2 to Helm 3) are fully supported. Managing apps installed with Helm 2 is not supported.
- **Operator-deployed apps:** Astra Control supports apps installed with namespace-scoped operators. These operators are generally designed with a "pass-by-value" rather than "pass-by-reference" architecture. The following are some operator apps that follow these patterns:
 - [Apache K8ssandra](#)
 - [Jenkins CI](#)
 - [Percona XtraDB Cluster](#)

Note that Astra Control might not be able to clone an operator that is designed with a "pass-by-reference" architecture (for example, the CockroachDB operator). During these types of cloning operations, the cloned operator attempts to reference Kubernetes secrets from the source operator despite having its own new secret as part of the cloning process. The clone operation might fail because Astra Control is unaware of the Kubernetes secrets in the source operator.



An operator and the app it installs must use the same namespace; you might need to modify the deployment .yaml file for the operator to ensure this is the case.

Install apps on your cluster

Now that you've added your cluster to Astra Control, you can install apps on the cluster. Persistent volumes will be provisioned on the new storage classes by default. After the pods are online, you can manage the app with Astra Control.

Astra Control will manage stateful apps only if the storage is on a storage class installed by Astra Control.

- [Learn about storage classes for GKE clusters](#)
- [Learn about storage classes for AKS clusters](#)

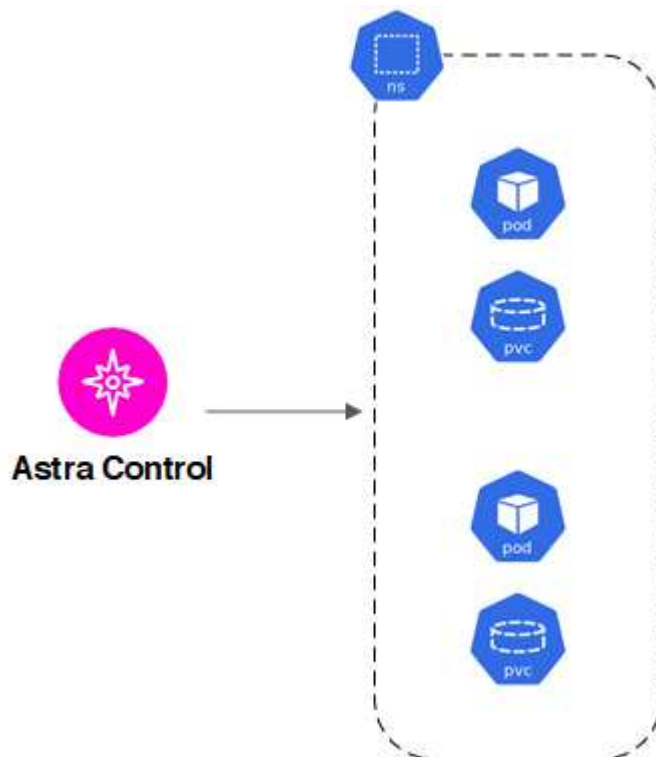
For help with deploying common applications from Helm charts, refer to the following:

- [Deploy MariaDB from a Helm chart](#)
- [Deploy MySQL from a Helm chart](#)
- [Deploy Postgres from a Helm chart](#)
- [Deploy Jenkins from a Helm chart](#)

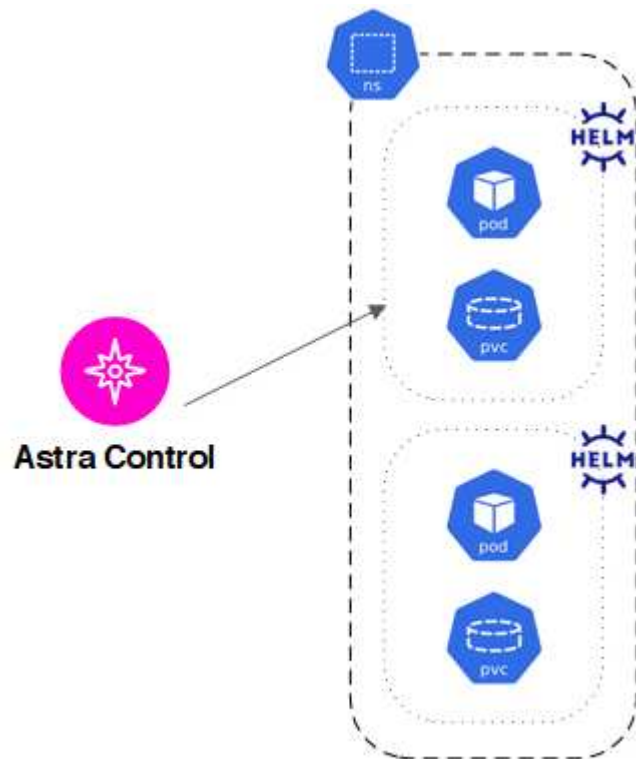
Manage apps

When Astra Control discovers the apps running on your clusters, they are unmanaged until you choose how you want to manage them. A managed application in Astra Control can be any of the following:

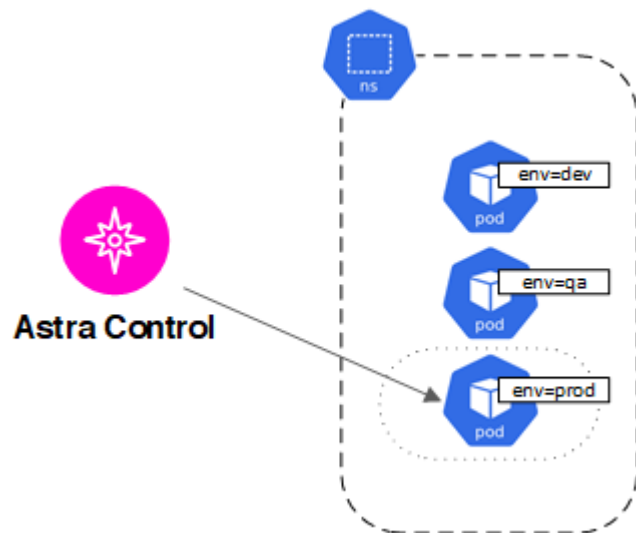
- A namespace, including all resources in that namespace



- An individual application deployed with helm3 within a namespace



- A group of resources that are identified by a Kubernetes label (this is called a *custom app* in Astra Control)



The sections below describe how to manage your apps using these options.

Manage apps by namespace

The **Discovered** section of the Apps page shows namespaces and the Helm-installed apps or custom-labeled apps in those namespaces. You can choose to manage each app individually or at the namespace level. It all comes down to the level of granularity that you need for data protection operations.

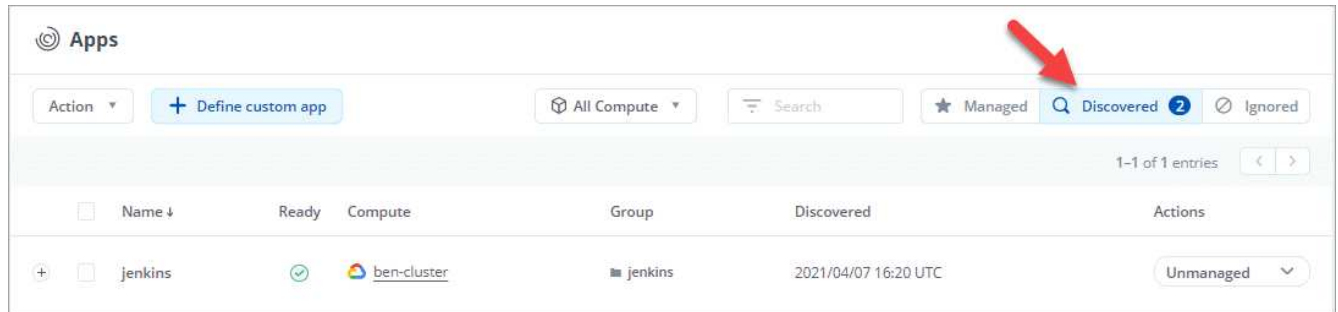
For example, you might want to set a backup policy for "maria" that has a weekly cadence, but you might need to back up "mariadb" (which is in the same namespace) more frequently than that. Based on those needs, you would need to manage the apps separately and not under a single namespace.

While Astra Control allows you to separately manage both levels of the hierarchy (the namespace and the

apps in that namespace), the best practice is to choose one or the other. Actions that you take in Astra Control can fail if the actions take place at the same time at both the namespace and app level.

Steps

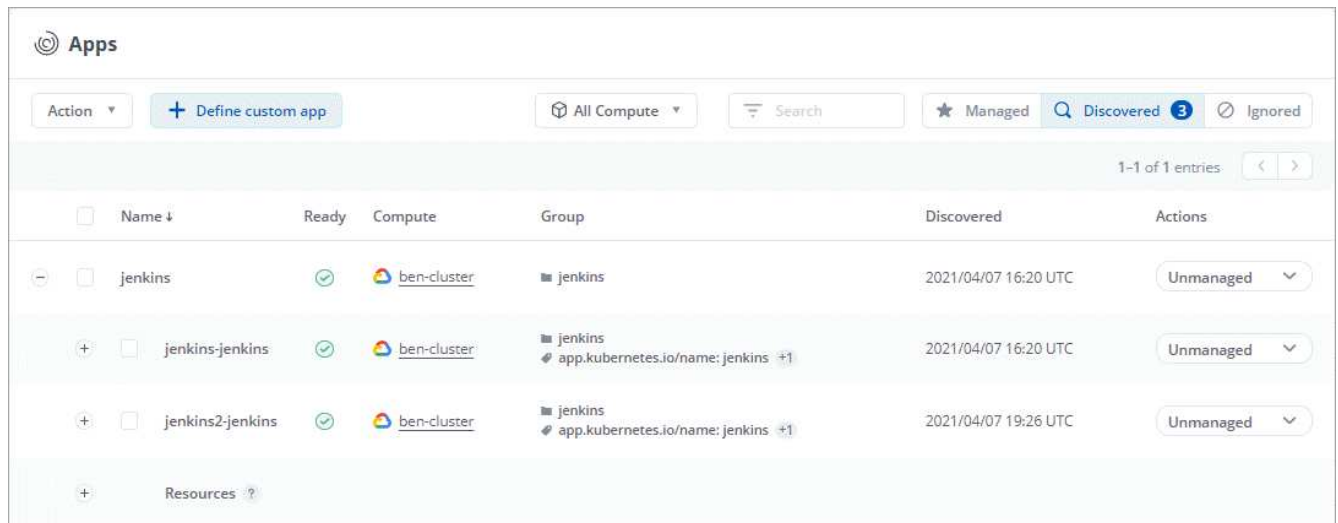
1. Select **Applications** and then select **Discovered**.



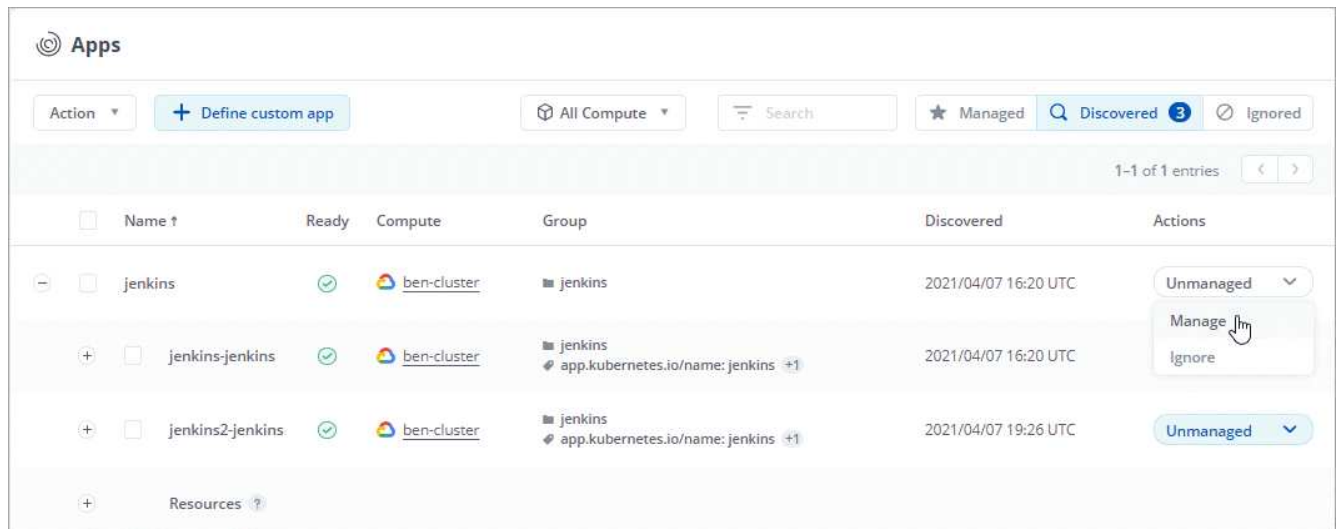
2. View the list of discovered namespaces and expand a namespace to view the apps and associated resources.

Astra Control shows you Helm apps and custom-labeled apps in namespace. If Helm labels are available, they're designated with a tag icon.

Here's an example with two apps in a namespace:

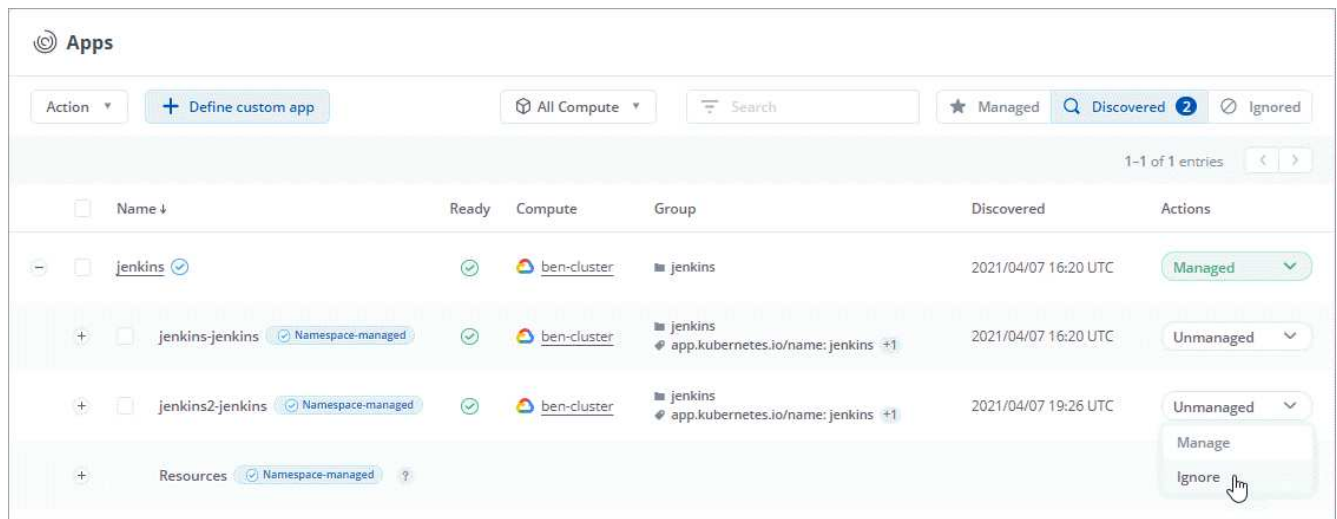


3. Decide whether you want to manage each app individually or at the namespace level.
4. At the desired level in the hierarchy, select the drop-down list in the **Actions** column and select **Manage**.



- If you don't want to manage an app, select the drop-down list in the **Actions** column for the desired app and select **Ignore**.

For example, if you wanted to manage all apps under the "jenkins" namespace together so that they have the same snapshot and backup policies, you would manage the namespace and ignore the apps in the namespace:



Result

Apps that you chose to manage are now available from the **Managed** tab. Any ignored apps will move to the **Ignored** tab. Ideally, the Discovered tab will show zero apps, so that as new apps are installed, they are easier to find and manage.

Manage apps by Kubernetes label

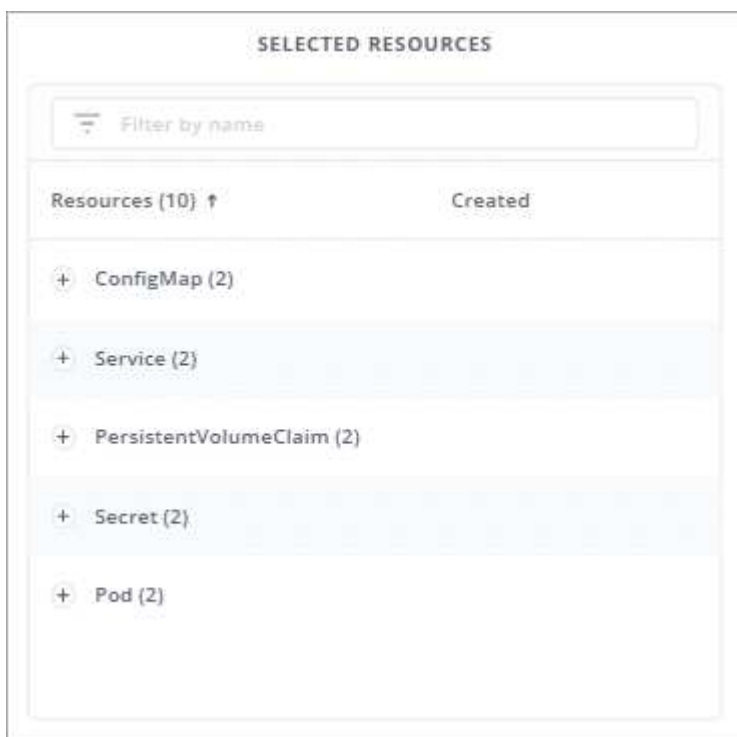
Astra Control includes an action at the top of the Apps page named **Define custom app**. You can use this action to manage apps that are identified with a Kubernetes label. [Learn more about defining apps by Kubernetes label.](#)

Steps

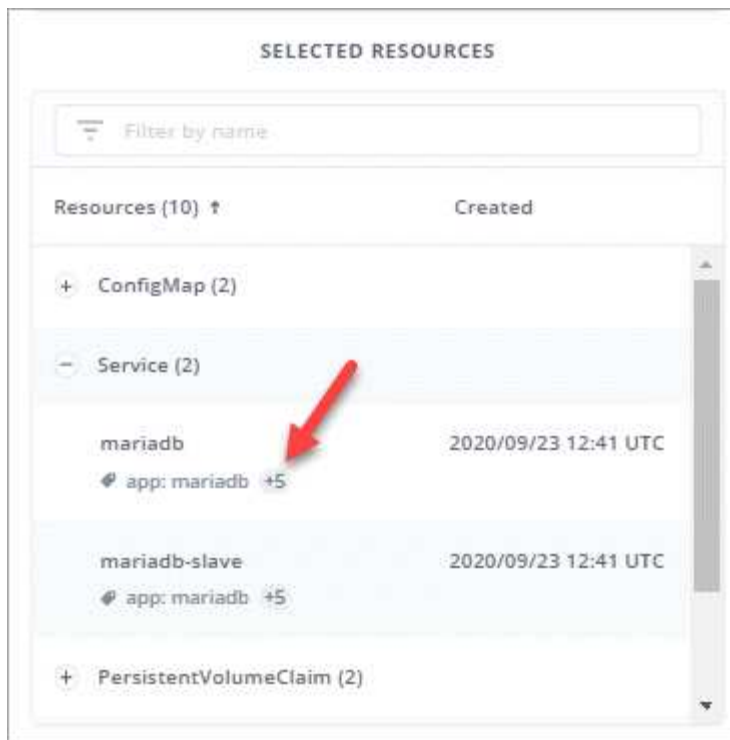
- Select **Applications > Define custom app**.

2. In the **Define Custom Application** dialog box, provide the required information to manage the app:
- a. **New App:** Enter the display name of the app.
 - b. **Cluster:** Select the cluster where the app resides.
 - c. **Namespace:** Select the namespace for the app.
 - d. **Label:** Enter a label or select a label from the resources below.
 - e. **Selected Resources:** View and manage the selected Kubernetes resources that you'd like to protect (pods, secrets, persistent volumes, and more).

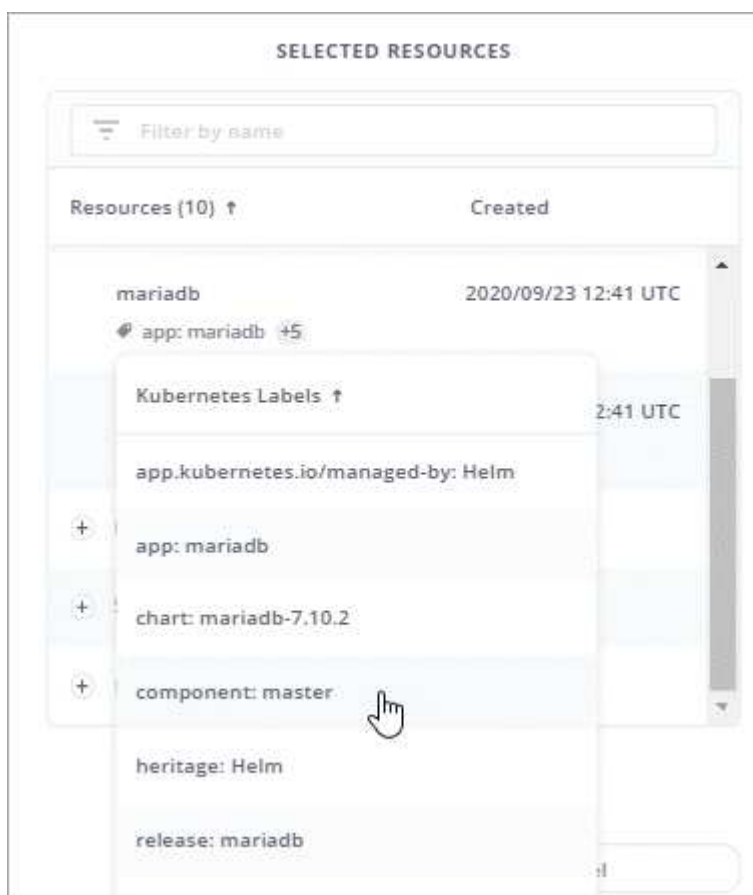
Here's an example:



- View the available labels by expanding a resource and selecting the number of labels.

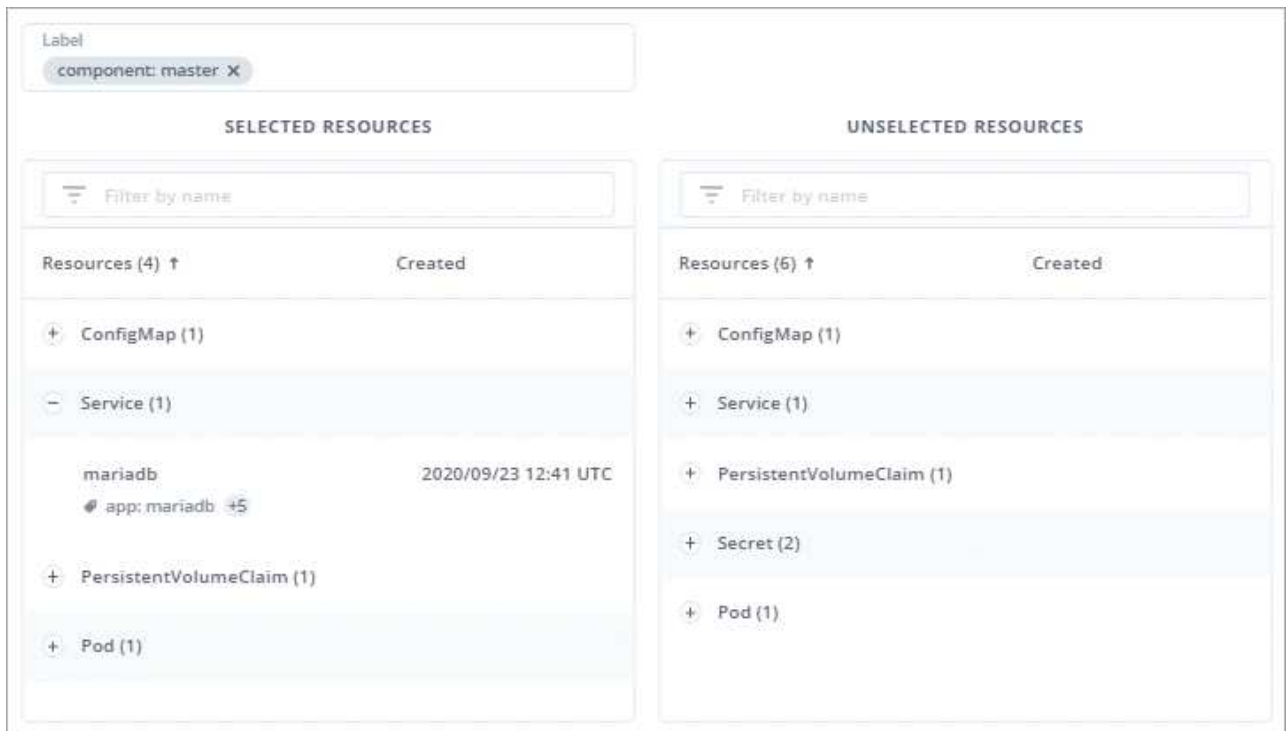


- Select one of the labels.



After you choose a label, it displays in the **Label** field. Astra Control also updates the **Unselected Resources** section to show the resources that don't match the selected label.

f. **Unselected Resources:** Verify the app resources that you don't want to protect.



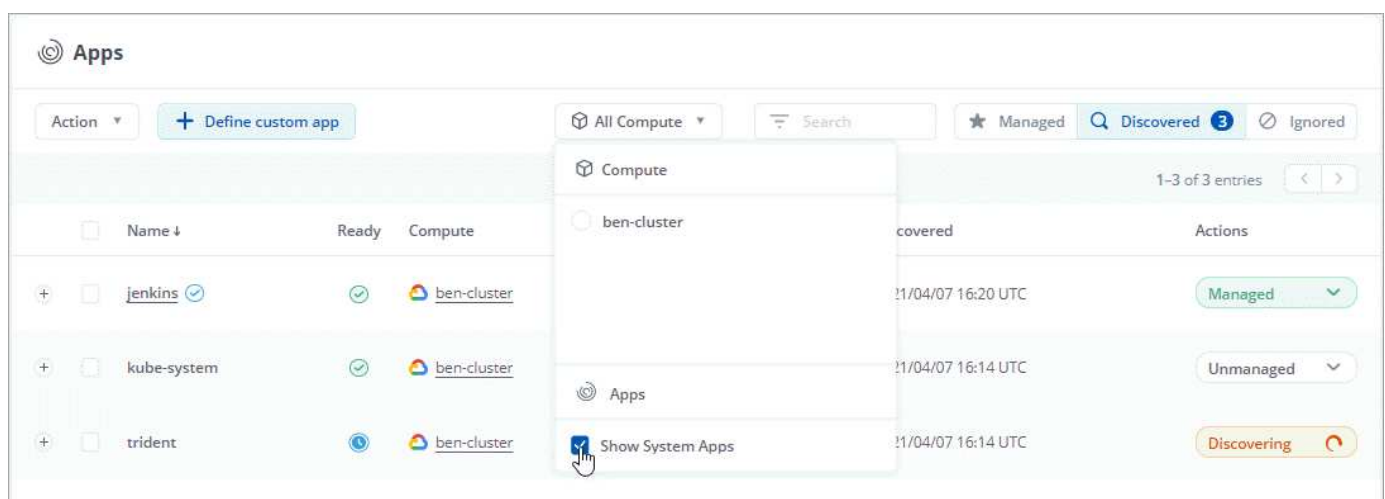
3. Select **Define Custom App**.

Result

Astra Control enables management of the app. You can now find it in the **Managed** tab.

What about system apps?

Astra Control also discovers the system apps running on a Kubernetes cluster. You can view them by filtering the Apps list.



We don't show you these system apps by default because it's rare that you'd need to back them up.

Protect apps with snapshots and backups

Protect your apps by taking snapshots and backups using an automated protection policy or on an ad-hoc basis.

Snapshots and backups

A *snapshot* is a point-in-time copy of an app that's stored on the same provisioned volume as the app. They are usually fast. Local snapshots are used to restore the application to an earlier point in time.

A *backup* is stored on object storage in the cloud. A backup can be slower to take compared to the local snapshots. But they can be accessed across regions in the cloud to enable app migrations. You can also choose a longer retention period for backups.



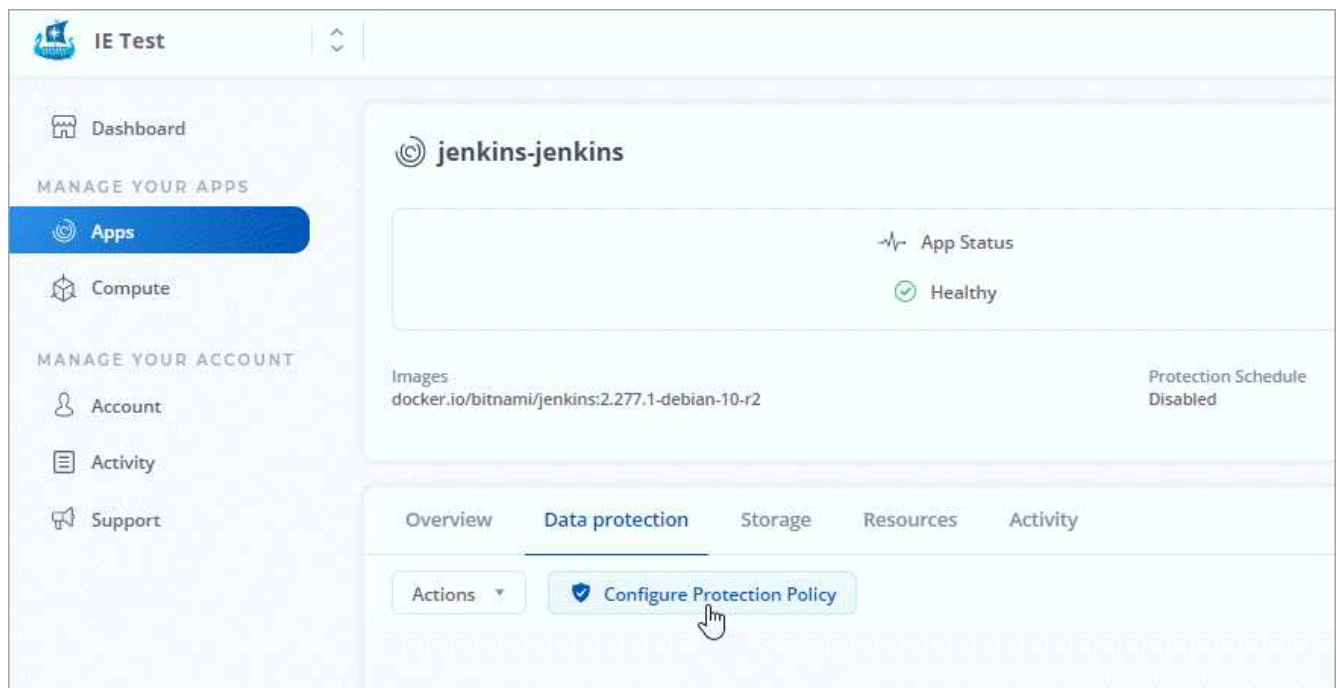
You can't be fully protected until you have a recent backup. This is important because backups are stored in an object store away from the persistent volumes. If a failure or accident wipes out the cluster and it's persistent storage, then you need a backup to recover. A snapshot wouldn't enable you to recover.

Configure a protection policy

A protection policy protects an app by creating snapshots, backups, or both at a defined schedule. You can choose to create snapshots and backups hourly, daily, weekly, and monthly, and you can specify the number of copies to retain.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select **Data Protection**.
3. Select **Configure Protection Policy**.



4. Define a protection schedule by choosing the number of snapshots and backups to keep for the hourly, daily, weekly, and monthly schedules.

You can define the hourly, daily, weekly, and monthly schedules concurrently. A schedule won't turn active until you set a retention level for snapshots and backups.

When you set a retention level for backups, you can choose the bucket where you'd like to store the backups.

The following example sets four protection schedules: hourly, daily, weekly, and monthly for snapshots and backups.

Configure protection policy STEP 1/2: DETAILS

PROTECTION SCHEDULE

- Hourly**: Every hour on the 0th minute, keep the last 4 snapshots
- Daily**: Daily at 05:00 (UTC), keep the last 7 snapshots
- Weekly**: Weekly on Mondays at 05:00 (UTC), keep the last 12 snapshots
- Monthly**: Every 1st of the month at 05:00 (UTC), keep the last 12 backups

● Hourly ● Daily ● Weekly ● **Monthly**

Day(s) of Month (optional): 1 x Time (UTC) (optional): 05:00 Snapshots to keep: 0 Backups to keep: 12

BACKUP DESTINATION

Bucket: ben-astra-bucket Default

OVERVIEW

Schedule and retention

Define a policy to continuously protect your application on a schedule and configure a retention count to get started.

For select stateful applications, expect I/O to pause for a short time during a backup or snapshot operation.

Read more in [Protection policies](#)

Application: maria
Namespace: maria
Cluster: david-ie-00

Cancel Review →

5. Select **Review**.

6. Select **Configure**.

Here's a video that shows each of these steps.

► <https://docs.netapp.com/us-en/astra-control-service/media/use/video-set-protection-policy.mp4> (video)

Result

Astra Control implements the data protection policy by creating and retaining snapshots and backups using the schedule and retention policy that you defined.

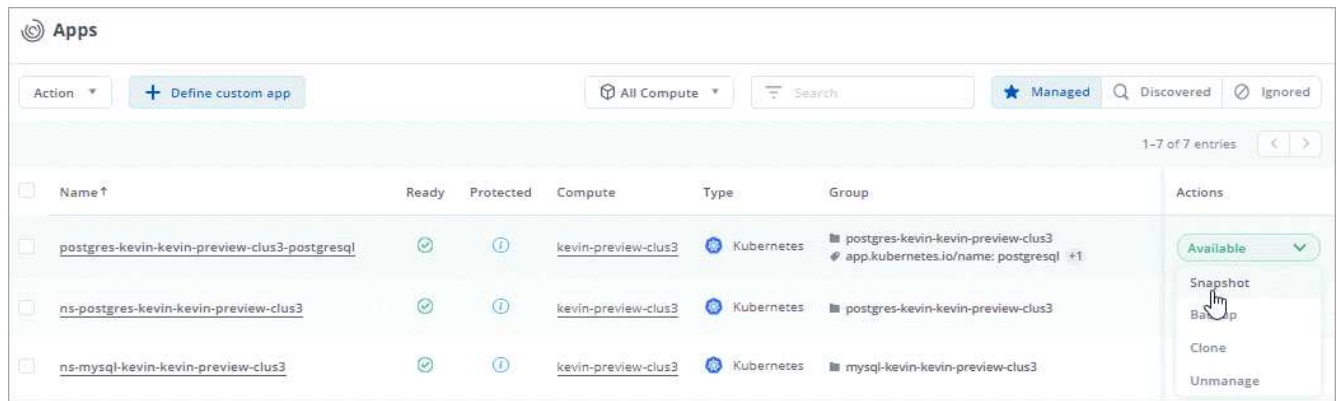
Create a snapshot

You can create an on-demand snapshot at any time.

Steps

1. Select **Applications**.
2. Select the drop-down list in the **Actions** column for the desired app.

3. Select **Snapshot**.



4. Customize the name of the snapshot and then select **Review Information**.

5. Review the snapshot summary and select **Snapshot App**.

Result

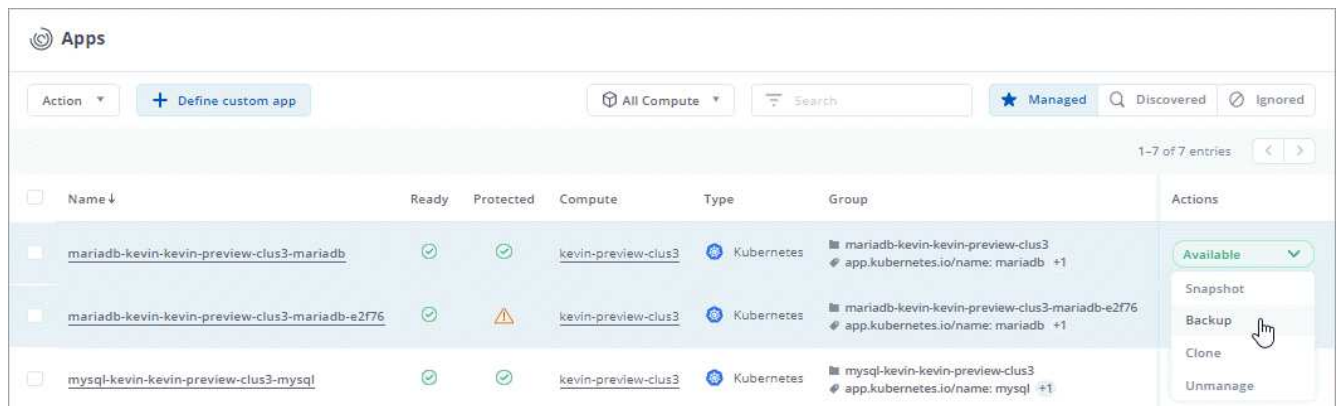
Astra Control creates a snapshot of the apps.

Create a backup

You can also back up an app at any time.

Steps

1. Select **Applications**.
2. Select the drop-down list in the **Actions** column for the desired app.
3. Select **Backup**.



4. Customize the name of the backup, choose whether to back up the app from an existing snapshot, and then select **Review Information**.

5. Review the backup summary and select **Backup App**.

Result

Astra Control creates a backup of the app.

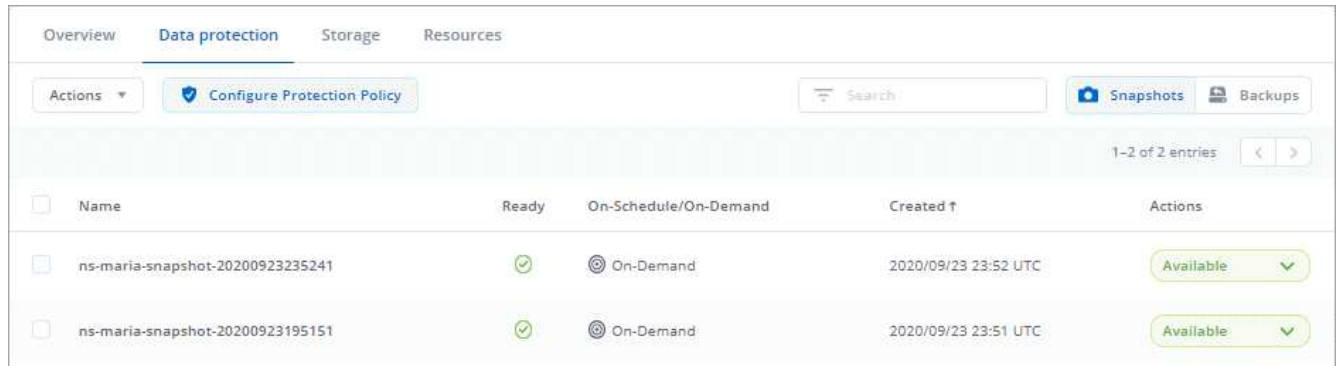
View snapshots and backups

You can view the snapshots and backups of an app from the Data Protection tab.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select **Data Protection**.

The snapshots display by default.



The screenshot shows the 'Data protection' tab in the Astra Control console. It displays a table of snapshots for a managed application. The table has columns for Name, Ready status, On-Schedule/On-Demand status, Created time, and Actions. Two snapshots are listed, both with a 'Ready' status and 'On-Demand' schedule. The 'Actions' column shows a dropdown menu with 'Available' selected.

Name	Ready	On-Schedule/On-Demand	Created ↑	Actions
ns-maria-snapshot-20200923235241	✓	⌚ On-Demand	2020/09/23 23:52 UTC	Available ✓
ns-maria-snapshot-20200923195151	✓	⌚ On-Demand	2020/09/23 23:51 UTC	Available ✓

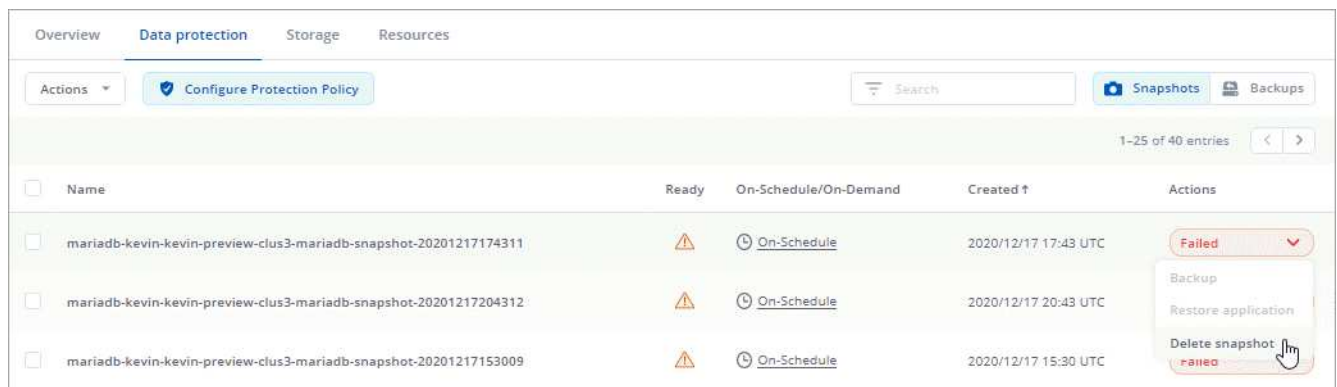
3. Select **Backups** to see the list of backups.

Delete snapshots

Delete the scheduled or on-demand snapshots that you no longer need.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select **Data Protection**.
3. Select the drop-down list in the **Actions** column for the desired snapshot.
4. Select **Delete snapshot**.



The screenshot shows the 'Data protection' tab in the Astra Control console. It displays a table of snapshots for a managed application. The table has columns for Name, Ready status, On-Schedule/On-Demand status, Created time, and Actions. Three snapshots are listed, all with a 'Ready' status and 'On-Schedule' schedule. The 'Actions' column shows a dropdown menu with 'Failed' selected. A mouse cursor is pointing at the 'Delete snapshot' option in the dropdown menu.

Name	Ready	On-Schedule/On-Demand	Created ↑	Actions
mariadb-kevin-kevin-preview-clus3-mariadb-snapshot-20201217174311	⚠	⌚ On-Schedule	2020/12/17 17:43 UTC	Failed ✓
mariadb-kevin-kevin-preview-clus3-mariadb-snapshot-20201217204312	⚠	⌚ On-Schedule	2020/12/17 20:43 UTC	Failed ✓
mariadb-kevin-kevin-preview-clus3-mariadb-snapshot-20201217153009	⚠	⌚ On-Schedule	2020/12/17 15:30 UTC	Failed ✓

5. Type the name of the snapshot to confirm deletion and then select **Yes, Delete snapshot**.

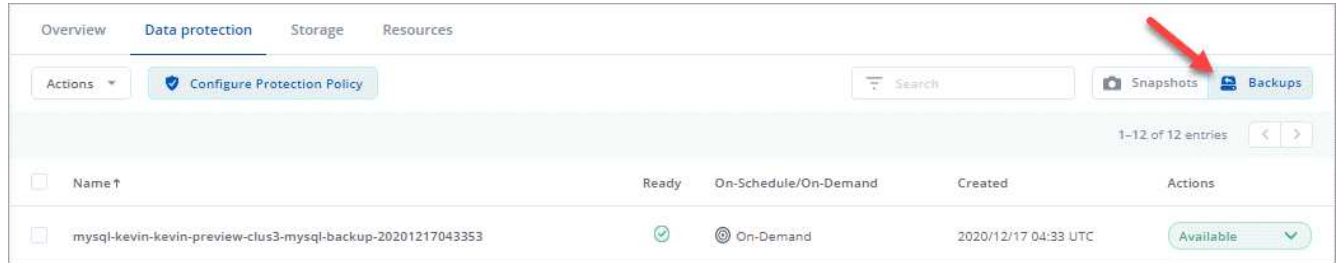
Result

Astra Control deletes the snapshot.

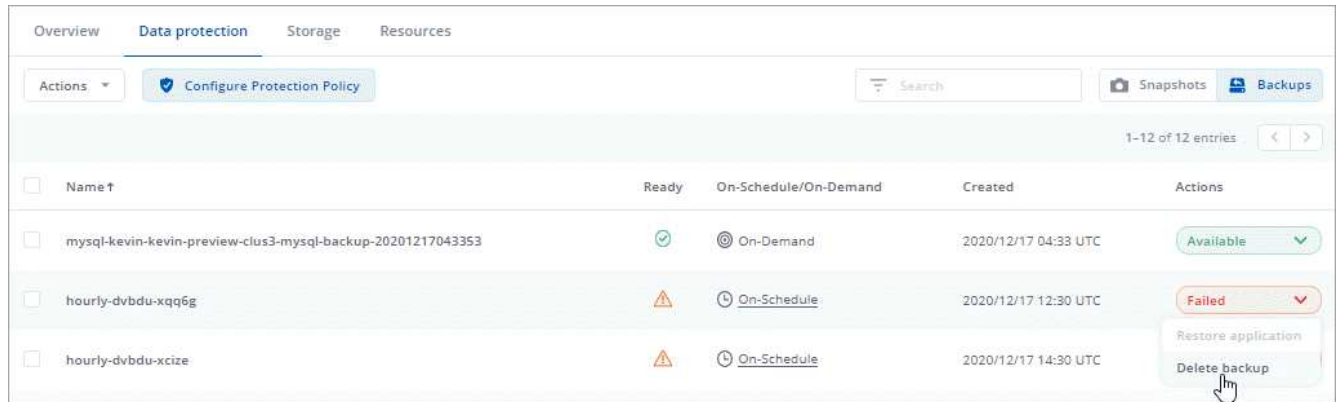
Delete backups

Delete the scheduled or on-demand backups that you no longer need.

1. Select **Applications** and then select the name of a managed app.
2. Select **Data Protection**.
3. Select **Backups**.



4. Select the drop-down list in the **Actions** column for the desired backup.
5. Select **Delete backup**.



6. Type the name of the backup to confirm deletion and then select **Yes, Delete backup**.

Result

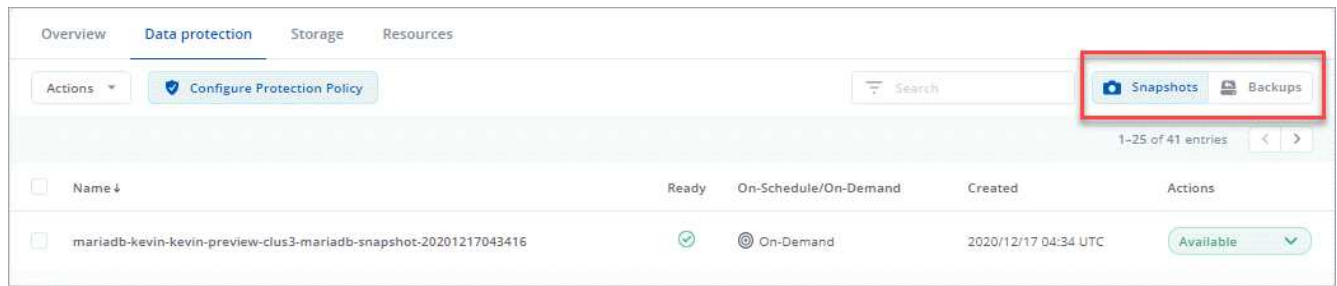
Astra Control deletes the backup.

Restore apps

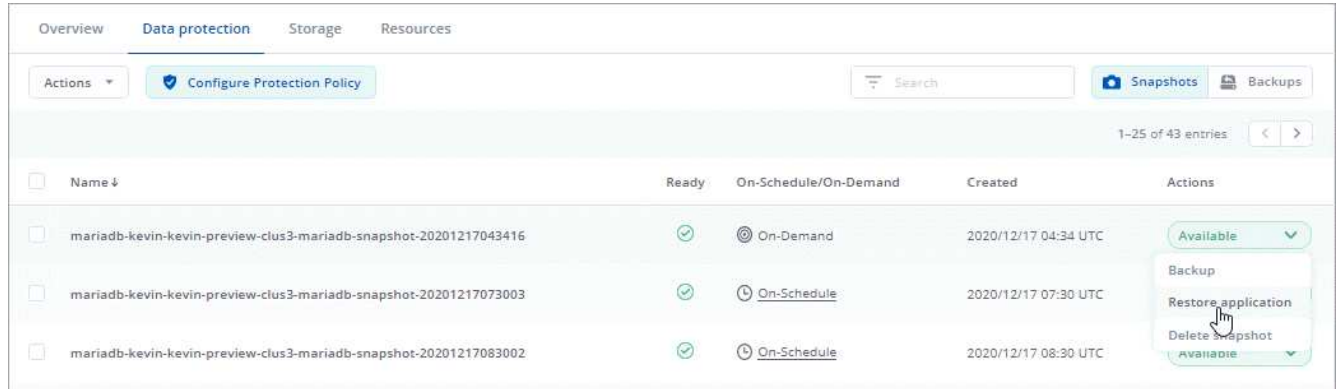
Astra Control can restore your application configuration and persistent storage from a snapshot or backup. Persistent storage backups are transferred from your object store, so restoring from an existing backup will complete the fastest.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select **Data protection**.
3. If you want to restore from a snapshot, keep **Snapshots** selected. Otherwise, select **Backups** to restore from a backup.



4. Select the drop-down list in the **Actions** column for the snapshot or backup from which you want to restore.
5. Select **Restore application**.



6. **Restore details:** Specify details for the restored app. To restore an app in-place (revert an app to an earlier version of itself), choose the same namespace and destination cluster that it is currently running in:
 - Enter a namespace for the app.
 - Choose the destination cluster for the app.
 - Click **Review**.
7. **Restore Summary:** Review details about the restore action, type "restore", and select **Restore**.

Result

Astra Control restores the app based on the information that you provided. If you restored the app in-place, the associated persistent volumes are deleted and recreated.

Clone and migrate apps

Clone an existing app to create a duplicate app on the same Kubernetes cluster or on another cluster. Cloning can help if you need to move applications and storage from one Kubernetes cluster to another. For example, you might want to move workloads through a CI/CD pipeline and across Kubernetes namespaces.

When Astra Control clones an app, it creates a clone of your application configuration and persistent storage.



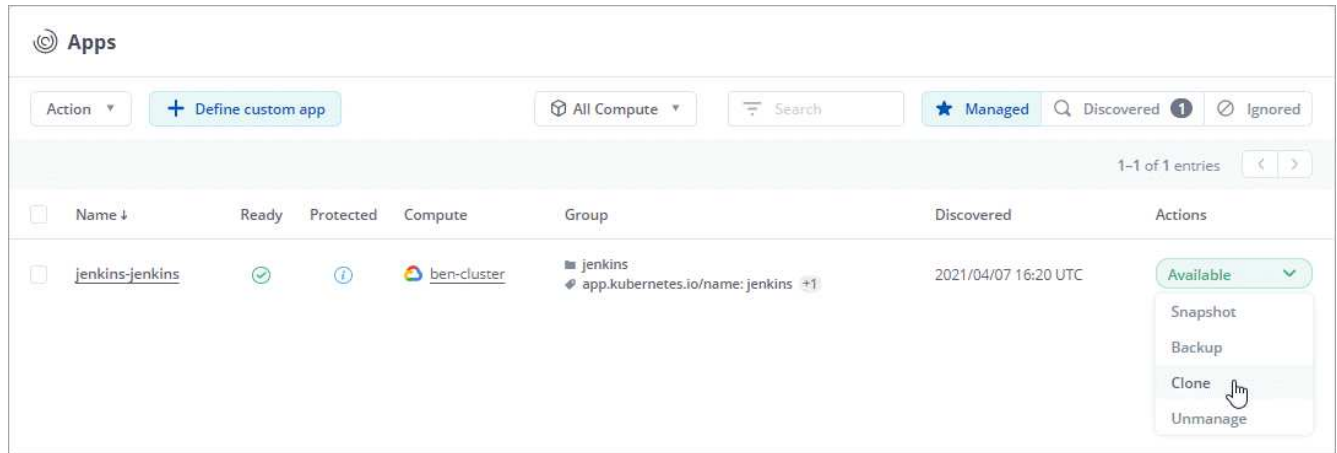
If you clone an operator-deployed instance of Jenkins CI, you need to manually restore the persistent data. This is a limitation of the app's deployment model.



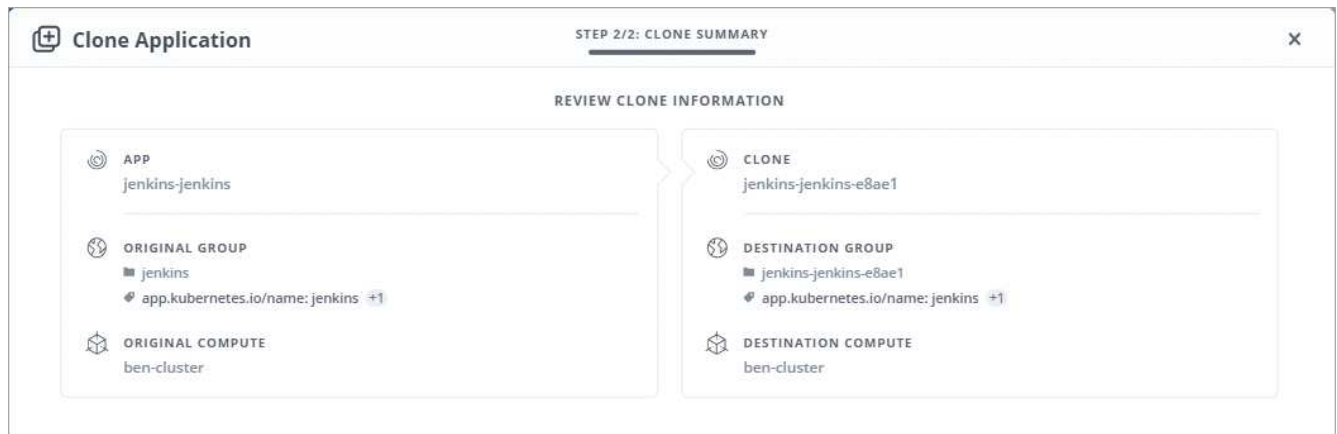
If you deploy an app with a StorageClass explicitly set and you need to clone the app, the target cluster must have the originally specified StorageClass. Cloning an application with an explicitly set StorageClass to a cluster that does not have the same StorageClass will fail.

Steps

1. Select **Applications**.
2. Select the drop-down list in the **Action** column for the desired app.
3. Select **Clone**.



4. **Clone details:** Specify details for the clone:
 - Keep the default name and namespace, or edit them.
 - Choose a destination cluster for the clone.
 - Choose whether you want to create the clone from an existing snapshot or backup. If you don't select this option, Astra Control creates the clone from the app's current state.
5. **Clone Summary:** Review the details about the clone and select **Clone App**.



Result

Astra Control clones that app based on the information that you provided.

Manage app execution hooks

An execution hook is a custom script that you can run before or after a snapshot of a managed app. For example, if you have a database app, you can use execution hooks to pause all database transactions before a snapshot, and resume transactions after the snapshot is complete. This ensures application-consistent snapshots.

Default execution hooks and regular expressions

For some apps, Astra Control comes with default execution hooks, provided by NetApp, that handle freeze and thaw operations before and after snapshots. Astra Control uses regular expressions to match an app's container image to these apps:

- MariaDB
 - Matching regular expression: `\bmariadb\b`
- MySQL
 - Matching regular expression: `\bmysql\b`
- PostgreSQL
 - Matching regular expression: `\bpostgresql\b`

If there is a match, the NetApp-provided default execution hooks for that app appear in the app's list of active execution hooks, and those hooks run automatically when snapshots of that app are taken. If one of your custom apps has a similar image name that happens to match one of the regular expressions (and you don't want to use the default execution hooks), you can either change the image name, or disable the default execution hook for that app and use a custom hook instead.

You cannot delete or modify the default execution hooks.

Important notes about custom execution hooks

Consider the following when planning execution hooks for your apps.

- Astra Control requires execution hooks to be written in the format of executable shell scripts.
- Script size is limited to 128KB.
- Astra Control uses execution hook settings and any matching criteria to determine which hooks are applicable to a snapshot.
- All execution hook failures are soft failures; other hooks and the snapshot are still attempted even if a hook fails. However, when a hook fails, a warning event is recorded in the **Activity** page event log.
- To create, edit, or delete execution hooks, you must be a user with Owner, Admin, or Member permissions.
- If an execution hook takes longer than 25 minutes to run, the hook will fail, creating an event log entry with a return code of "N/A". Any affected snapshot will time out and be marked as failed, with a resulting event log entry noting the timeout.



Since execution hooks often reduce or completely disable the functionality of the application they are running against, you should always try to minimize the time your custom execution hooks take to run.

When a snapshot is run, execution hook events take place in the following order:

1. Any applicable NetApp-provided default pre-snapshot execution hooks are run on the appropriate containers.
2. Any applicable custom pre-snapshot execution hooks are run on the appropriate containers. You can create and run as many custom pre-snapshot hooks as you need, but the order of execution of these hooks before the snapshot is neither guaranteed nor configurable.
3. The snapshot is performed.
4. Any applicable custom post-snapshot execution hooks are run on the appropriate containers. You can create and run as many custom post-snapshot hooks as you need, but the order of execution of these hooks after the snapshot is neither guaranteed nor configurable.
5. Any applicable NetApp-provided default post-snapshot execution hooks are run on the appropriate containers.



You should always test your execution hook scripts before enabling them in a production environment. You can use the 'kubectl exec' command to conveniently test the scripts. After you enable the execution hooks in a production environment, test the resulting snapshots to ensure they are consistent. You can do this by cloning the app to a temporary namespace, restoring the snapshot, and then testing the app.

View existing execution hooks

You can view existing custom or NetApp-provided default execution hooks for an app.

Steps

1. Go to **Applications** and then select the name of a managed app.
2. Select the **Execution hooks** tab.

You can view all enabled or disabled execution hooks in the resulting list. You can see a hook's status, source, and when it runs (pre- or post-snapshot). To view event logs surrounding execution hooks, go to the **Activity** page in the left-side navigation area.

Create a custom execution hook

You can create a custom execution hook for an app. See [Execution hook examples](#) for hook examples. You need to have Owner, Admin, or Member permissions to create execution hooks.



When you create a custom shell script to use as an execution hook, remember to specify the appropriate shell at the beginning of the file, unless you are running linux commands or providing the full path to an executable.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select the **Execution hooks** tab.
3. Select **Add a new hook**.
4. In the **Hook Details** area, depending on when the hook should run, choose **Pre-Snapshot** or **Post-Snapshot**.
5. Enter a unique name for the hook.
6. (Optional) Enter any arguments to pass to the hook during execution, pressing the Enter key after each argument you enter to record each one.

7. In the **Container Images** area, if the hook should run against all container images contained within the application, enable the **Apply to all container images** check box. If instead the hook should act only on one or more specified container images, enter the container image names in the **Container image names to match** field.
8. In the **Script** area, do one of the following:
 - Upload a custom script.
 - a. Select the **Upload file** option.
 - b. Browse to a file and upload it.
 - c. Give the script a unique name.
 - d. (Optional) Enter any notes other administrators should know about the script.
 - Paste in a custom script from the clipboard.
 - a. Select the **Paste from clipboard** option.
 - b. Select the text field and paste the script text into the field.
 - c. Give the script a unique name.
 - d. (Optional) Enter any notes other administrators should know about the script.
9. Select **Add hook**.

Disable an execution hook

You can disable an execution hook if you want to temporarily prevent it from running before or after a snapshot of an app. You need to have Owner, Admin, or Member permissions to disable execution hooks.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select the **Execution hooks** tab.
3. Select the **Actions** dropdown for a hook that you wish to disable.
4. Select **Disable**.

Delete an execution hook

You can remove an execution hook entirely if you no longer need it. You need to have Owner, Admin, or Member permissions to delete execution hooks.

Steps

1. Select **Applications** and then select the name of a managed app.
2. Select the **Execution hooks** tab.
3. Select the **Actions** dropdown for a hook that you wish to delete.
4. Select **Delete**.

Execution hook examples

Use the following examples to get an idea of how to structure your execution hooks. You can use these hooks as templates, or as test scripts.

Simple success example

This is an example of a simple hook that succeeds and writes a message to standard output and standard error.

```
#!/bin/sh

# success_sample.sh
#
# A simple noop success hook script for testing purposes.
#
# args: None
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#
```

```
# log something to stdout
info "running success_sample.sh"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

Simple success example (bash version)

This is an example of a simple hook that succeeds and writes a message to standard output and standard error, written for bash.

```
#!/bin/bash

# success_sample.bash
#
# A simple noop success hook script for testing purposes.
#
# args: None

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
```

```

error() {
    msg "ERROR: $" 1>&2
}

#
# main
#

# log something to stdout
info "running success_sample.bash"

# exit with 0 to indicate success
info "exit 0"
exit 0

```

Simple success example (zsh version)

This is an example of a simple hook that succeeds and writes a message to standard output and standard error, written for Z shell.

```

#!/bin/zsh

# success_sample.zsh
#
# A simple noop success hook script for testing purposes.
#
# args: None
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#

```



```

info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running success_sample.zsh"

# exit with 0 to indicate success
info "exit 0"
exit 0

```

Success with arguments example

The following example demonstrates how you can use args in a hook.

```

#!/bin/sh

# success_sample_args.sh
#
# A simple success hook script with args for testing purposes.
#
# args: Up to two optional args that are echoed to stdout
#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

```

```

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running success_sample_args.sh"

# collect args
arg1=$1
arg2=$2

# output args and arg count to stdout
info "number of args: $#\"
info "arg1 ${arg1}\"
info "arg2 ${arg2}\"

# exit with 0 to indicate success
info "exit 0\"
exit 0

```

Pre-snapshot / post-snapshot hook example

The following example demonstrates how the same script can be used for both a pre-snapshot and a post-snapshot hook.

```
#!/bin/sh
```

```

# success_sample_pre_post.sh
#
# A simple success hook script example with an arg for testing purposes
# to demonstrate how the same script can be used for both a prehook and
posthook
#
# args: [pre|post]

# unique error codes for every error case
ebase=100
eusage=$((ebase+1))
ebadstage=$((ebase+2))
epre=$((ebase+3))
epost=$((ebase+4))

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

```

```

#
# Would run prehook steps here
#
prehook() {
    info "Running noop prehook"
    return 0
}

#
# Would run posthook steps here
#
posthook() {
    info "Running noop posthook"
    return 0
}

#
# main
#

# check arg
stage=$1
if [ -z "${stage}" ]; then
    echo "Usage: $0 <pre|post>"
    exit ${eusage}
fi

if [ "${stage}" != "pre" ] && [ "${stage}" != "post" ]; then
    echo "Invalid arg: ${stage}"
    exit ${ebadstage}
fi

# log something to stdout
info "running success_sample_pre_post.sh"

if [ "${stage}" = "pre" ]; then
    prehook
    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during prehook"
    fi
fi

if [ "${stage}" = "post" ]; then
    posthook

```

```

    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during posthook"
    fi
fi
exit ${rc}

```

Failure example

The following example demonstrates how you can handle failures in a hook.

```

#!/bin/sh

# failure_sample_arg_exit_code.sh
#
# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#

```

```

error() {
    msg "ERROR: $" 1>&2
}

#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"

argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

# exit with specified exit code
exit ${argexitcode}

```

Verbose failure example

The following example demonstrates how you can handle failures in a hook, with more verbose logging.

```

#!/bin/sh

# failure_sample_verbose.sh
#
# A simple failure hook script with args for testing purposes.
#
# args: [The number of lines to output to stdout]

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#

```

```

# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running failure_sample_verbose.sh"

# output arg value to stdout
linecount=$1
info "line count ${linecount}"

# write out a line to stdout based on line count arg
i=1
while [ "$i" -le ${linecount} ]; do
    info "This is line ${i} from failure_sample_verbose.sh"
    i=$(( i + 1 ))
done

error "exiting with error code 8"
exit 8

```

Failure with an exit code example

The following example demonstrates a hook failing with an exit code.

```

#!/bin/sh

# failure_sample_arg_exit_code.sh
#

```

```

# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"

argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

```



```
# exit with specified exit code
exit ${argexitcode}
```

Success after failure example

The following example demonstrates a hook failing the first time it is run, but succeeding after the second run.

```
#!/bin/sh

# failure_then_success_sample.sh
#
# A hook script that fails on initial run but succeeds on second run for
# testing purposes.
#
# Helpful for testing retry logic for post hooks.
#
# args: None
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}

#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}
```

```
#
# main
#


# log something to stdout
info "running failure_success sample.sh"


if [ -e /tmp/hook-test.junk ] ; then
    info "File does exist. Removing /tmp/hook-test.junk"
    rm /tmp/hook-test.junk
    info "Second run so returning exit code 0"
    exit 0
else
    info "File does not exist. Creating /tmp/hook-test.junk"
    echo "test" > /tmp/hook-test.junk
    error "Failed first run, returning exit code 5"
    exit 5
fi
```

View app and compute health

View a summary of app and cluster health


Click the **Dashboard** to see a high-level view of your apps, clusters, and their health.

 **Welcome to LongBoat IE**

You are currently on the Free Plan and your limit is set to 10 applications. If you want to upgrade to the Premium Plan, please click the button below.



[Manage Plans](#) →



Resources summary



**Apps**


4/10

Managed

 All Healthy 



 Not Fully Protected  4

 Discovered  0


**Compute**

1


Managed

 All Healthy 


Getting started

 [Manage Kubernetes compute](#)


Add compute to install the Astra storage operator

 [Add your applications](#)

Install your stateful applications onto Managed Compute and Astra storage classes

 [Manage applications](#)

Enable your applications to be protected and cloned

 [Invite users](#)

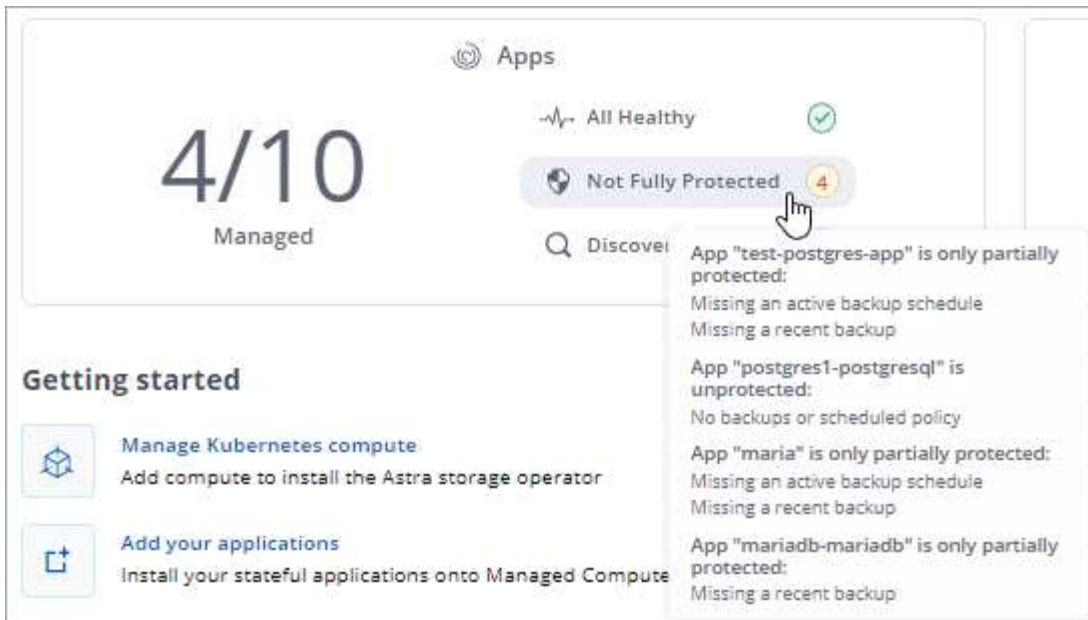
Share access to your account with colleagues

The Apps tile helps you identify the following:

- How many apps you're currently managing.
- Whether those managed apps are healthy.
- Whether the apps are fully protected (they're protected if recent backups are available).
- The number of apps that were discovered, but are not yet managed.

Ideally, this number would be zero because you would either manage or ignore apps after they're discovered. And then you would monitor the number of discovered apps on the Dashboard to identify when developers add new apps to a cluster.

Note that these aren't just numbers or statuses—you can drill down from each of these. For example, if apps aren't fully protected, you can hover over the icon to identify which apps aren't fully protected, which includes a reason why.



The Clusters tile provides similar details about the health of the cluster and you can drill down to get more details just like you can with an app.

View the health and details of clusters

After you add Kubernetes clusters to Astra Control, you can view details about the cluster, such as its location, the worker nodes, persistent volumes, and storage classes.

Steps

1. Select **Clusters**.
2. Select the cluster name.
3. View the information in the **Overview** and **Storage** tabs to find the information that you're looking for.
 - **Overview**: Details about the worker nodes, including their state.
 - **Storage**: The persistent volumes associated with the cluster, including the storage class and state.
 - **Activity**: The Astra activities related to the cluster.

<div> ben-cluster Available </div> <div> <div>Version</div> <div>v1.18.16-gke.302</div> </div> <div> <div>Location</div> <div> northamerica-northeast1</div> </div> <div> <div>Provisioners</div> <div>Trident 21.01.2-custom+49f...</div> </div>						
<div> <div>Overview</div> <div>Storage</div> <div>Activity</div> </div> <div> <div>Search</div> <div>1-7 of 7 entries</div> </div>						
Worker Nodes ↓	Node size	Public IP	Memory	CPUs	Created	State
gke-ben-cluster-default-pool-6e469af1-j0xx	e2-medium	35.203.80.144	3.84 GiB	2 vCPUs	2021/04/07 15:51 UTC	Running
gke-ben-cluster-default-pool-6e469af1-kqn3	e2-medium	35.203.108.92	3.84 GiB	2 vCPUs	2021/04/07 15:51 UTC	Running
gke-ben-cluster-default-pool-e493d6b5-7hr	e2-medium	34.95.15.242	3.84 GiB	2 vCPUs	2021/04/07 15:51 UTC	Running

View the health and details of an app

After you start managing an app, Astra Control provides details about the app that enables you to identify its status (whether it's healthy), its protection status (whether it's fully protected in case of failure), the pods, persistent storage, and more.

The screenshot displays the Astra Control interface for the 'jenkins-jenkins' application. At the top, the application name is shown with a status badge 'Available'. Below this, two summary boxes are present: 'App Status' showing 'Healthy' and 'App Protection Status' showing 'Partially Protected'. A row of four informational cards follows: 'Images' (docker.io/bitnami/jenkins:2.277.1-debian-10-r2), 'Protection Schedule' (Every hour on the hour, Daily at 02:00 (UTC), Weekly on Mondays at 02:00..., Every 1st of the month at 02:00...), 'Group' (jenkins, app.kubernetes.io/...), and 'Compute' (ben-cluster). Below these is a tabbed interface with 'Overview' selected. A search bar and '1-1 of 1 entries' are visible. A table lists the pod details:

Pod ↓	Ready	Node	Created	State
jenkins-7c9c88f745-dxqg2 app.kubernetes.io/instance: jenkins, app.kubernetes.io/managed-by: Helm +2	✓	gke-ben-cluster-default-pool-6e469af1-j0xx	2021/04/07 16:19 UTC	Available

Steps

1. Select **Applications** and then select the name of an app.
2. Select around to find the information that you're looking for:

App Status

Provides a status that reflects the app's state in Kubernetes. For example, are pods and persistent volumes online? If an app is unhealthy, you'll need to go and troubleshoot the issue on the cluster by looking at Kubernetes logs. Astra Control doesn't provide information to help you fix a broken app.

App Protection Status

Provides a status of how well the app is protected:

- **Fully protected:** The app has an active backup schedule and a successful backup that's less than a week old
- **Partially protected:** The app has an active backup schedule, an active snapshot schedule, or a successful backup or snapshot
- **Unprotected:** Apps that are neither fully protected or partially protected.

You can't be fully protected until you have a recent backup. This is important because backups are stored in an object store away from the persistent volumes. If a failure or accident wipes out the cluster and it's persistent storage, then you need a backup to recover. A snapshot wouldn't enable you to recover.

Overview

Information about the state of the pods that are associated with the app.

Data protection

Enables you to configure a data protection policy and to view the existing snapshots and backups.

Storage

Shows you the app-level persistent volumes. The state of a persistent volume is from the perspective of the Kubernetes cluster.

Resources

Enables you to verify which resources are being backed up and managed.

Activity

The Astra Control activities related to the app.

Manage buckets

Manage the buckets that Astra uses for backups and clones by adding additional buckets and by changing the default bucket for the Kubernetes clusters in your cloud provider.

Only Admins can add and modify buckets.

How Astra Control uses buckets

When you start managing your first Kubernetes cluster, Astra Control Service creates the default bucket for your cloud provider in the same geography as the managed cluster.

Astra Control Service uses this default bucket for the backups and clones that you create. You can then use the backups to restore and clone apps between clusters.

If you add additional buckets to Astra Control Service, you can select from those buckets when you create a protection policy. You can also change the default bucket that Astra Control Service uses for ad-hoc backups and clones.



Astra Control Service checks whether a destination bucket is accessible prior to starting a backup or a clone.

View existing buckets

View the list of buckets that are available to Astra Control Service to determine their status and to identify the default bucket for your cloud provider.

A bucket can have any of the following states:

Pending

After you add a bucket, it starts in the pending state while Astra Control looks at it for the first time.

Available

The bucket is available for use by Astra Control.

Removed

The bucket isn't operational at the moment. Hover your mouse over the status icon to identify what the problem is.

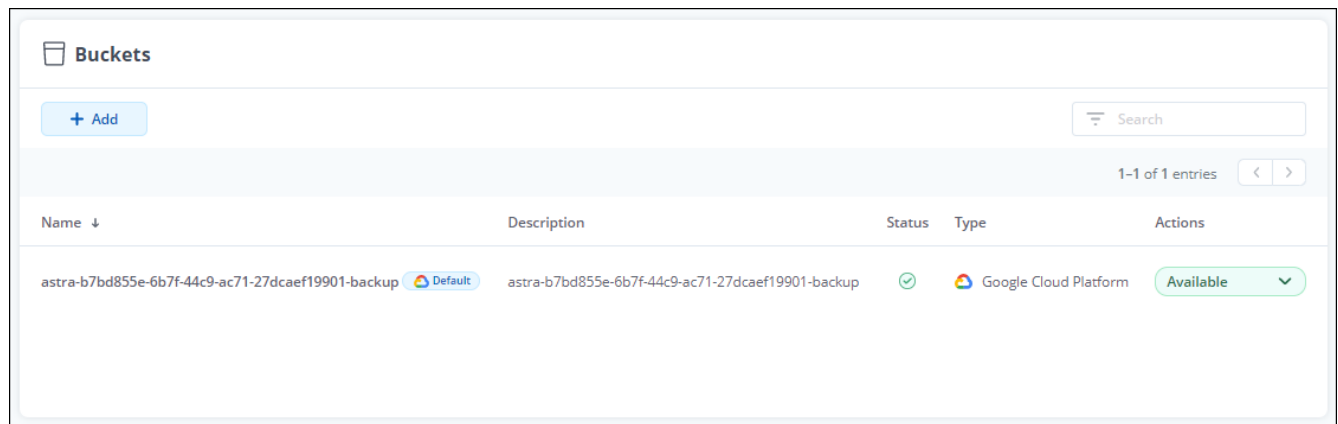
If a bucket is in the Removed state, you can still set it as the default bucket and assign it to a protection schedule. But if the bucket isn't in the Available state by the time a data protection operation starts, then that operation will fail.

Step

1. Under **Manage your storage**, select **Buckets**.

The list of buckets available to Astra Control Service displays.

As you can see from the following example, there is only one bucket available: the default bucket that Astra created.



Name ↓	Description	Status	Type	Actions
astra-b7bd855e-6b7f-44c9-ac71-27dcaef19901-backup	astra-b7bd855e-6b7f-44c9-ac71-27dcaef19901-backup	✓	Google Cloud Platform	Available

Add an additional bucket

After you start managing a cluster in your cloud provider, you can add additional buckets at any time. This enables you to choose between buckets when creating a protection policy and to change the default bucket for ad-hoc backups and clones.

Note that Astra Control Service doesn't enable you to remove a bucket after you've added it.

What you'll need

- The name of an existing bucket in your cloud provider.
- If your bucket is in Azure, it must belong to the resource group named *astra-backup-rg*.

Steps

1. Under **Manage your storage**, select **Buckets**.
2. Select **Add** and follow the prompts to add the bucket.
 - **Type**: Choose your cloud provider.

Your cloud provider is available only after Astra Control Service has started managing a cluster that's running in that cloud provider.

- **Existing bucket name**: Enter the name of the bucket.
- **Description**: Optionally enter a description of the bucket.

- **Make this bucket the default bucket for this cloud:** Choose whether you would like to use this bucket as the default bucket for ad-hoc backups and clones.
- **Select credentials:** Choose the credentials that provide Astra Control Service with the permissions that it needs to manage the bucket.

Here's an example that shows adding a new bucket in Google Cloud Platform.

3. Select **Add** to add the bucket.

Result

Astra Control Service adds the additional bucket. You can now choose the bucket when creating a protection policy.

Change the default bucket

Change the default bucket that Astra Control Service should use for backups and clones. Each cloud provider has its own default bucket.

Astra Control Service uses the default bucket for a cloud provider for ad-hoc backups and for ad-hoc clones when you don't choose to clone from an existing backup.

Steps

1. Under **Manage your storage**, select **Buckets**.
2. Select the drop-down list in the **Actions** column for the bucket that you want to edit.
3. Select **Make this bucket the default bucket for this cloud**.
4. Select **Update**.

Manage your account

Set up billing

Astra Control's Free Plan enables you to manage up to 10 apps in your account. If you want to manage more than 10 apps, then you'll need to set up billing by upgrading from the Free Plan to the Premium Plan.

Billing overview

There are two types of costs associated with using Astra Control Service: charges from NetApp for the Astra Control Service and charges from your cloud provider for persistent volumes and object storage.

Astra Control Service billing

Astra Control Service offers three plans:

Free Plan

Manage up to 10 apps for free.

Premium PayGo

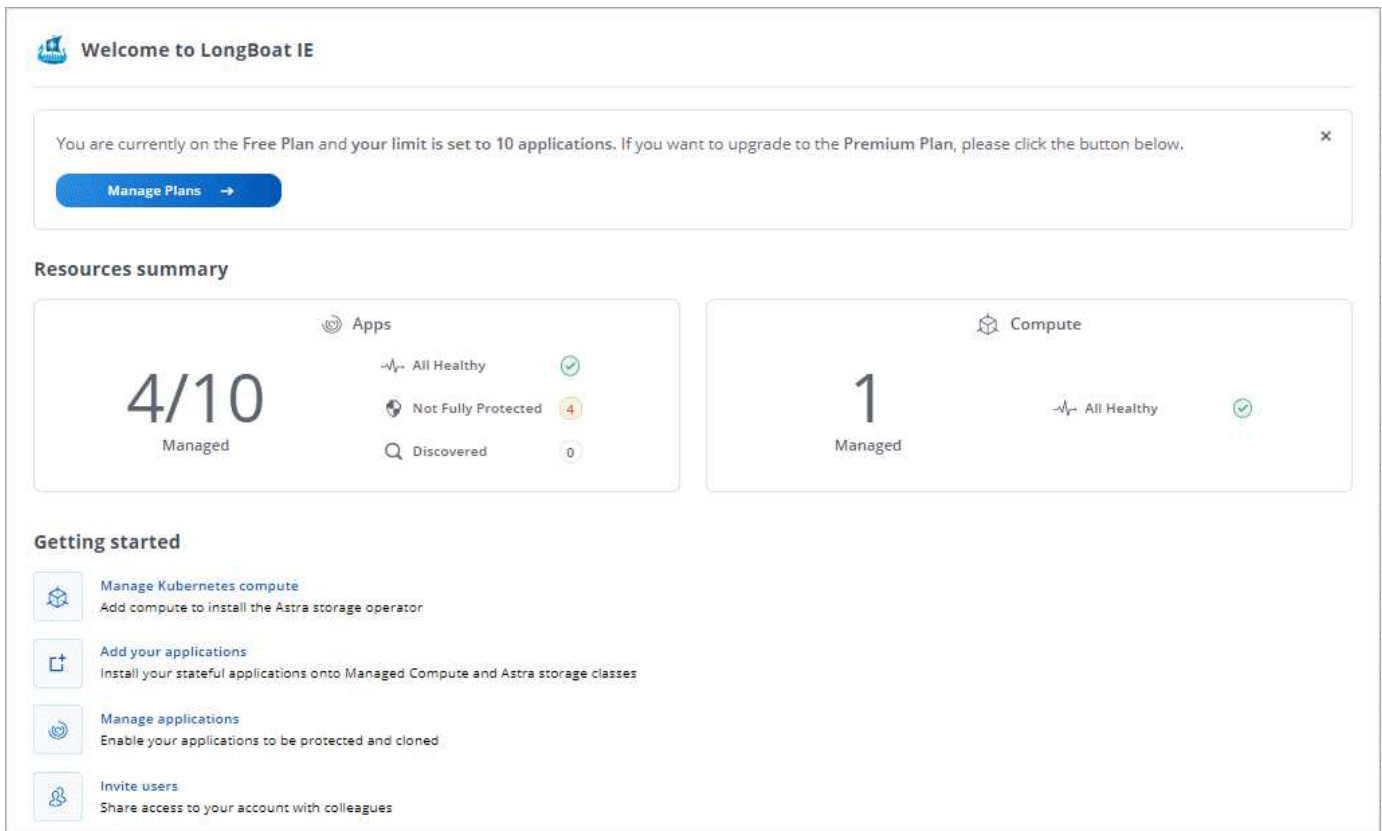
Manage an unlimited amount of apps at a rate of \$.005 per minute, per app.

Premium Subscription

Pre-pay at a discounted rate with an annual subscription that enables you to manage up to 10 apps per *application pack*. Contact NetApp Sales to purchase as many packs as needed for your organization—for example, purchase 3 packs to manage 30 apps from Astra Control Service. If you manage more apps than allowed by your annual subscription, then you'll be charged at the overage rate of \$0.005 per minute, per application (the same as Premium PayGo).

If you don't have an Astra Control account yet, purchasing the Premium Subscription automatically creates an Astra Control account for you. If you have an existing Free Plan, then you're automatically converted to the Premium Subscription.

When you create an Astra Control account, you're automatically signed up for the Free Plan. Astra Control's Dashboard shows you how many apps you're currently managing out of the 10 free apps that you're allowed:



If you try to manage an 11th app, Astra Control notifies you that you've reached the limit of the Free Plan. It then prompts you to upgrade from the Free Plan to a Premium Plan.

You can upgrade to a Premium Plan at any time. After you upgrade, Astra Control starts charging you for *all* managed apps in the account. The first 10 apps don't stay in the Free Plan.

Google Cloud billing

When you manage GKE clusters with Astra Control Service, persistent volumes are backed by NetApp Cloud Volumes Service and backups of your apps are stored in a Google Cloud Storage bucket.

- [View pricing details for Cloud Volumes Service.](#)

Note that Astra Control Service supports all service types and service levels. The service type that you use depends on your [Google Cloud region](#).

- [View pricing details for Google Cloud storage buckets.](#)

Microsoft Azure billing

When you manage AKS clusters with Astra Control Service, persistent volumes are backed by Azure NetApp Files and backups of your apps are stored in an Azure Blob container.

- [View pricing details for Azure NetApp Files.](#)
- [View pricing details for Microsoft Azure Blob storage.](#)

Important notes

- Your billing plan is per Astra Control account.

If you have multiple accounts, then each has its own billing plan.

- Your Astra Control bill includes charges for managing your Kubernetes apps. You're charged separately by your cloud provider for the storage backend for persistent volumes.

[Learn more about Astra Control pricing.](#)

- Each billing period ends on the last day of the month.
- You can't downgrade from a Premium Plan to the Free Plan.

Upgrade from the Free Plan to the Premium PayGo Plan

Upgrade your billing plan at any time to start managing more than 10 apps from Astra Control by paying as you go. All you need is a valid credit card.

Steps

1. Select **Account** and then select **Billing**.
2. Under **Plans**, go to **Premium PayGo** and select **Upgrade Now**.
3. Provide payment details for a valid credit card and select **Upgrade to Premium Plan**.



Astra Control will email you if the credit card is nearing expiration.

Result

You can now manage more than 10 apps. Astra Control starts charging you for *all* apps that you're currently managing.

Upgrade from the Free Plan to the Premium Subscription

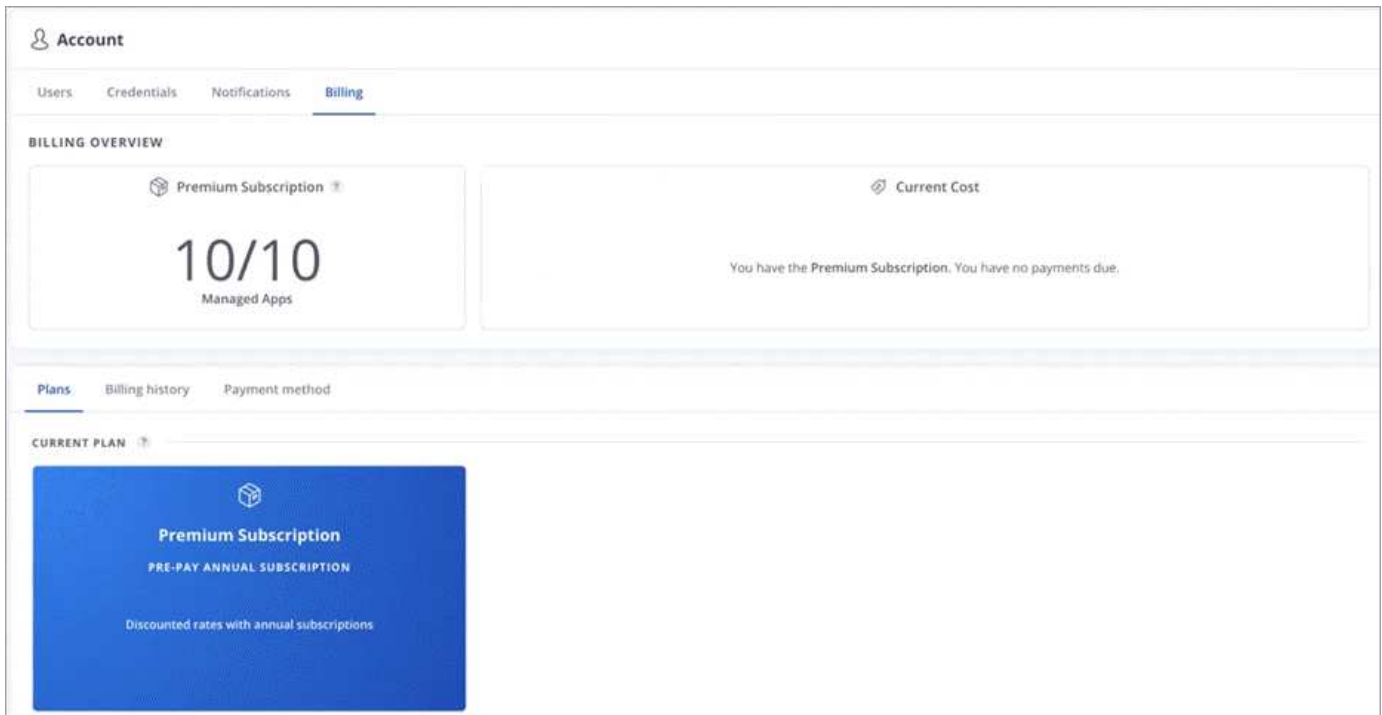
Contact NetApp Sales to pre-pay at a discounted rate with an annual subscription.

Steps

1. Select **Account** and then select **Billing**.
2. Under **Plans**, go to **Premium Subscription** and select **Contact Sales**.
3. Provide details to the sales team to start the process.

Result

A NetApp Sales representative will contact you to process your purchase order. After the order is complete, Astra Control will reflect your current plan on the Billing tab.



View your current costs and billing history

Astra Control shows you your current monthly costs, as well as a detailed billing history by app.

Steps

1. Select **Account** and then select **Billing**.

Your current costs appear under the billing overview.

2. To view the billing history by app, select **Billing history**.

Astra Control shows you the usage minutes and cost for each app. A usage minute is how many minutes Astra Control managed your app during a billing period.

3. Select the drop-down list to select a previous month.

Change the credit card for Premium PayGo

If needed, you can change the credit card that Astra Control has on file for billing.

Steps

1. Select **Account > Billing > Payment method**.
2. Select the configure icon.
3. Modify the credit card.

Invite and remove users

Invite users to join your Astra Control account and remove users that should no longer have access to the account.

Invite users

Account Owners and Admins can invite other users to join the Astra Control account.

Steps

1. Make sure that the user has a [Cloud Central login](#).
2. Select **Account**.
3. In the **Users** tab, select **+ Invite users**.
4. Enter the user's name, email address, and their role.

Note the following:

- The email address must match the email address that the user used to sign up to Cloud Central.
- Each role provides the following permissions:
 - An **Owner** has Admin permissions and can delete accounts.
 - An **Admin** has Member permissions and can invite other users.
 - A **Member** can fully manage apps and clusters.
 - A **Viewer** can view resources.

5. Select **Send invite(s)**.

Result

The user will receive an email that invites them to join your account.

Change a user's role

An Account Owner can change the role of all users, while an Account Admin can change the role of users who have the Admin, Member, or Viewer role.

Steps

1. Select **Account**.
2. In the **Users** tab, select the drop-down list in the **Role** column for the user.
3. Select a new role and then select **Change Role** when prompted.

Result

Astra Control updates the user's permissions based on the new role that you selected.

Remove users

An Account Owner can remove other users from the account at any time.

Steps

1. Select **Account**.
2. In the **Users** tab, select the users that you want to remove.
3. Select **Actions** and select **Remove user/s**.
4. When you're prompted, confirm deletion by typing the user's name and then select **Yes, Remove User**.

Result

Astra Control removes the user from the account.

Add and remove credentials

Add and remove cloud provider credentials from your account at any time. Astra Control uses these credentials to discover a Kubernetes cluster, the apps on the cluster, and to provision resources on your behalf.

Note that all users in Astra Control share the same sets of credentials.

Add credentials

The most common way to add credentials to Astra Control is when you manage clusters, but you can also add credentials from the Account page. The credentials will then be available to choose when you manage additional Kubernetes clusters.

What you'll need

- For GKE, you should have the service account key file for a service account that has the required permissions. [Learn how to set up a service account](#).
- For AKS, you should have the JSON file that contains the output from the Azure CLI when you created the service principal. [Learn how to set up a service principal](#).

You'll also need your Azure subscription ID, if you didn't add it to the JSON file.

Steps

1. Select **Account > Credentials**.
2. Select **Add Credentials**.
3. Select either **Microsoft Azure** or **Google Cloud Platform**.
4. Enter a name for the credentials that distinguishes them from other credentials in Astra Control.
5. Provide the required credentials.
 - a. **Microsoft Azure:** Provide Astra Control with details about your Azure service principal by uploading a JSON file or by pasting the contents of that JSON file from your clipboard.

The JSON file should contain the output from the Azure CLI when you created the service principal. It can also include your subscription ID so it's automatically added to Astra Control. Otherwise, you need to manually enter the ID after providing the JSON.
 - b. **Google Cloud Platform:** Provide the Google Cloud service account key file either by uploading the file or by pasting the contents from your clipboard.
6. Select **Add Credentials**.

Result

The credentials are now available to select when you add a cluster to Astra Control.

Remove credentials

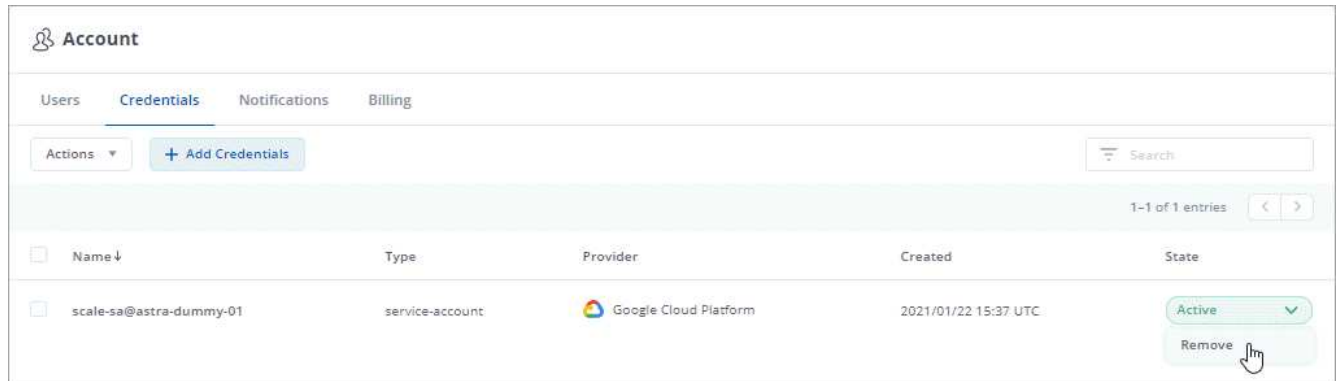
Remove credentials from an account at any time. You should only remove credentials after [unmanaging all clusters](#).



The first set of credentials that you add to Astra Control is always in use because Astra Control uses the credentials to authenticate to the backup bucket. It's best not to remove these credentials.

Steps

1. Select **Account > Credentials**.
2. Select the drop-down list in the **State** column for the credentials that you want to remove.
3. Select **Remove**.



4. Type the name of the credentials to confirm deletion and then select **Yes, Remove Credentials**.

Result

Astra Control removes the credentials from the account.

View account activity

You can view details about the activities in your Astra Control account. For example, when new users were invited, when a cluster was added, or when a snapshot was taken. You also have the ability to export your account activity to a CSV file.

Steps to view all account activity in Astra Control

1. Select **Activity**.
2. Use the filters to narrow down the list of activities or use the search box to find exactly what you're looking for.
3. Select **Export to CSV** to download your account activity to a CSV file.

Steps to view account activity for a specific app

1. Select **Applications** and then select the name of an app.
2. Select **Activity**.

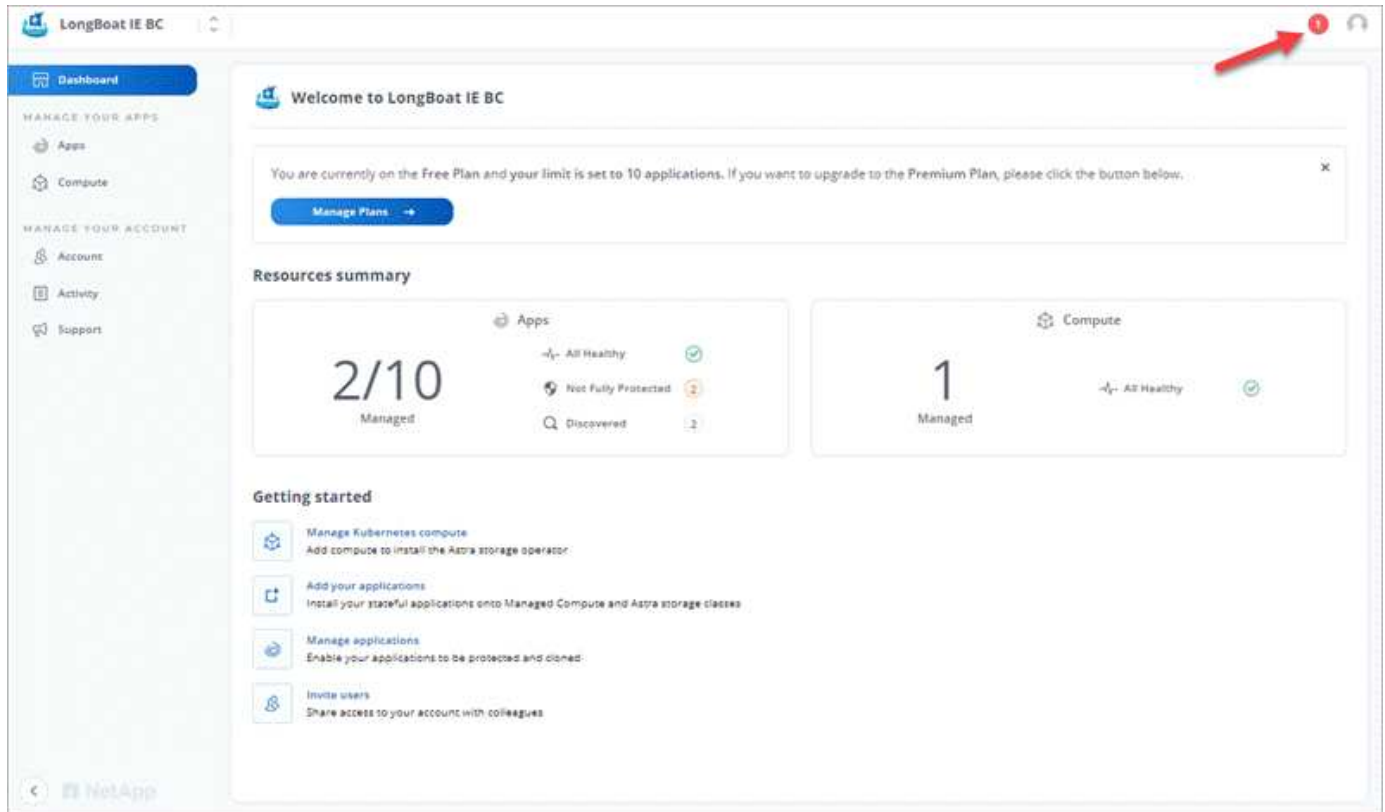
Steps to view account activity for clusters

1. Select **Clusters** and then select the name of the cluster.
2. Select **Activity**.

View and manage notifications

Astra Control notifies you when actions have completed or failed. For example, you'll see a notification if a backup of an app completed successfully.

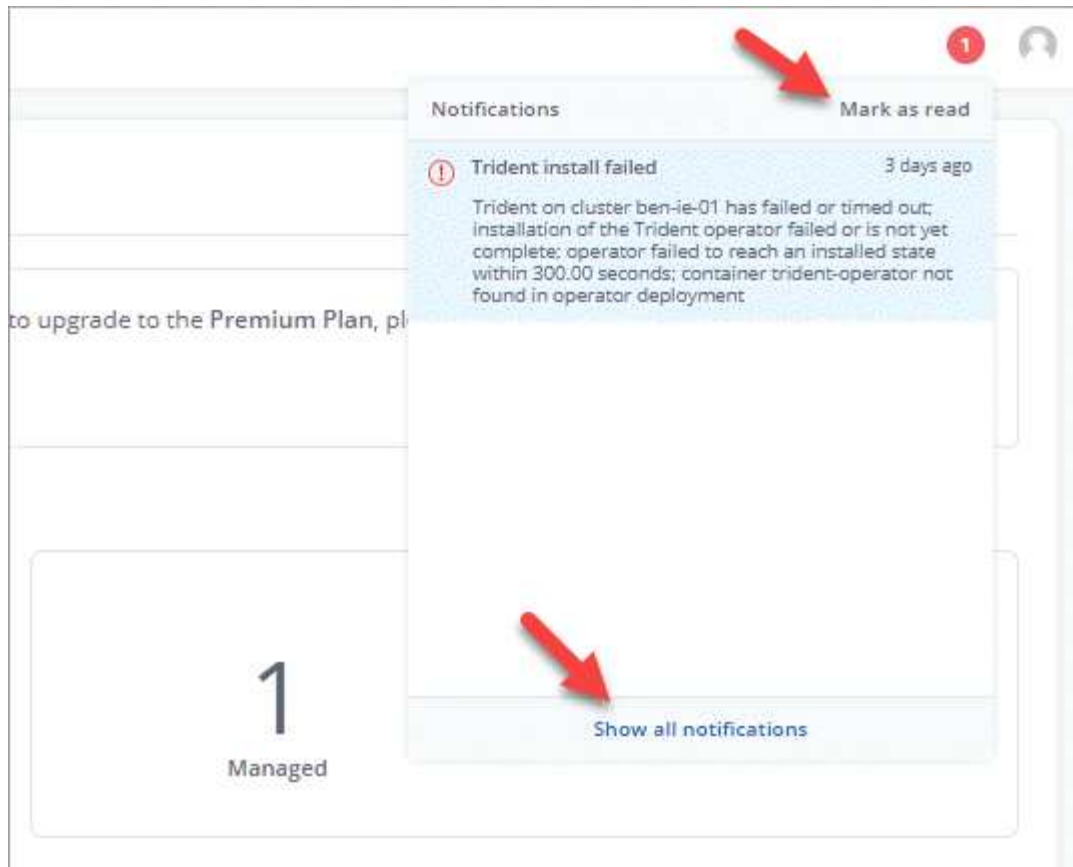
The number of unread notifications is available in the top right of the interface:



You can view these notifications and mark them as read (this can come in handy if you like to clear unread notifications like we do).

Steps

1. Select the number of unread notifications in the top right.



2. Review the notifications and then select **Mark as read** or **Show all notifications**.

If you selected **Show all notifications**, the Notifications page loads.

3. On the **Notifications** page, view the notifications, select the ones that you want to mark as read, select **Action** and select **Mark as read**.

Close your account

If you no longer need your Astra Control account, you can close it at any time.



Buckets that Astra Control automatically created will be automatically deleted when you close your account.

Steps

1. [Unmanage all apps and clusters](#).
2. [Remove credentials from Astra Control](#).
3. Select **Account > Billing > Payment method**.
4. Select **Close Account**.
5. Enter your account name and confirm to close the account.

Unmanage apps and clusters

Remove any apps or clusters that you no longer want to manage from Astra Control.

Stop managing an app

Stop managing apps that you no longer want to back up, snapshot, or clone from Astra Control.

- Any existing backups and snapshots will be deleted.
- Applications and data remain available.

Steps

1. Select **Applications**.
2. Select the checkbox for the apps that you no longer want to manage.
3. Select the **Action** drop-down and select **Unmanage application/s**.
4. Confirm that you want to unmanage the apps and then select **Yes, Unmanage Applications**.

Result

Astra Control stops managing the app.

Stop managing a cluster

Stop managing the cluster that you no longer want to manage from Astra Control. As a best practice, we recommend that you remove the cluster from Astra Control before you delete it through GCP.

- This action stops your cluster from being managed by Astra Control. It doesn't make any changes to the cluster's configuration and it doesn't delete the cluster.
- Astra Trident won't be uninstalled from the cluster. [Learn how to uninstall Astra Trident](#).

Steps

1. Select **Clusters**.
2. Select the checkbox for the cluster that you no longer want to manage.
3. Select the **Actions** drop-down and select **Unmanage**.
4. Confirm that you want to unmanage the cluster and then select **Yes, Unmanage cluster**.

Result

Astra Control stops managing the cluster.

Deleting clusters from your cloud provider

Before you delete a Kubernetes cluster that has persistent volumes (PV) residing on NetApp storage classes, you need to first delete the persistent volume claims (PVC) following one of the methods below. Deleting the PVC and PV before deleting the cluster ensures that you don't receive unexpected bills from your cloud provider.

- **Method #1:** Delete the application workload namespaces from the cluster. Do *not* delete the Trident namespace.
- **Method #2:** Delete the PVCs and the pods, or the deployment where the PVs are mounted.

When you manage a Kubernetes cluster from Astra Control, applications on that cluster use Cloud Volumes Service or Azure NetApp Files as the storage backend for persistent volumes. If you delete the cluster from your cloud provider without first removing the PVs, the backend volumes are *not* deleted along with the cluster.

Using one of the above methods will delete the corresponding PVs from your cluster. Make sure that there are no PVs residing on NetApp storage classes on the cluster before you delete it.

If you didn't delete the persistent volumes before you deleted the cluster, then you'll need to manually delete the backend volumes from Cloud Volumes Service for Google Cloud or from Azure NetApp Files.

Copyright Information

Copyright © 2021 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.