Use Astra

Astra Control Center

NetApp
February 28, 2022

Table of Contents

Use Astra

Manage apps

Start managing apps

After you add a cluster to Astra Control management, 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 and their resources.

App management requirements

Astra Control has the following 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

Astra Control supports the following application installation methods:

- **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 or manage existing apps on the cluster. Any app that is scoped to a namespace can be managed. After the pods are online, you can manage the app with Astra Control.

For help with deploying validated apps 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

Astra Control enables you to manage your apps at the namespace level or by Kubernetes label.

> Apps installed with Helm 2 are not supported.

You can perform the following activities to manage apps:

- Manage apps
  - Manage apps by namespace
  - Manage apps by Kubernetes label
- Ignore apps
- Unmanage apps

Astra Control itself is not a standard app; it is a "system app." You should not try to manage Astra Control itself. Astra Control itself isn't shown by default for management. To see system apps, use the "Show system apps" filter.

For instructions on how to manage apps using the Astra Control API, see the Astra Automation and API information.

After a data protection operation (clone, backup, restore) and subsequent persistent volume resize, there is up to a twenty-minute delay before the new volume size is shown in the UI. The data protection operation is successful within minutes, and you can use the management software for the storage backend to confirm the change in volume size.

Manage apps by namespace

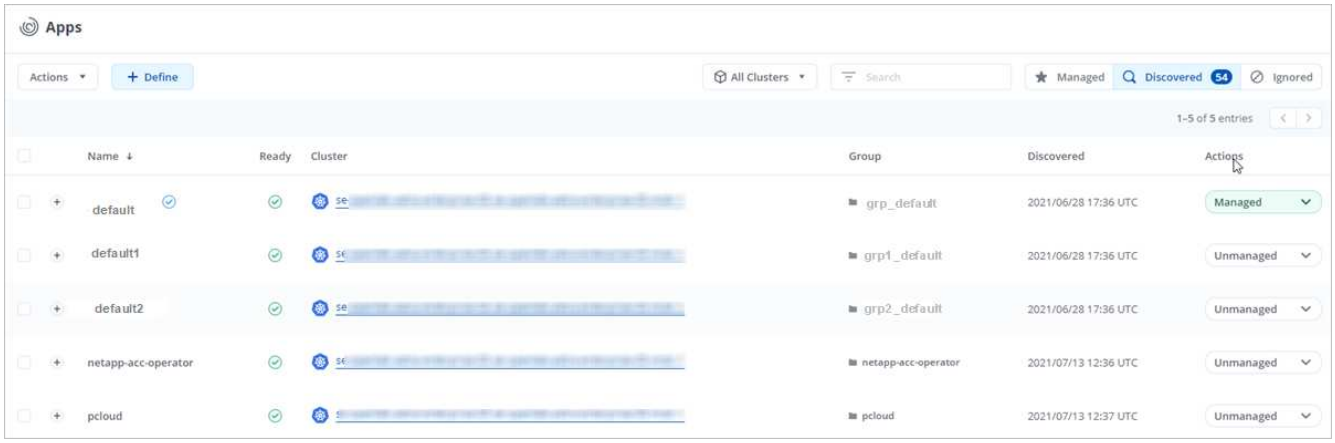
The **Discovered** section of the Apps page shows namespaces and any Helm-installed apps or custom-labeled apps in those namespaces. You can choose to manage each app individually or at the namespace level. It all comes down to the level of granularity that you need for data protection operations.

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 enables 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. From the left navigation bar, select **Applications**.
2. Select **Discovered**.



3. View the list of discovered namespaces. Expand the namespace to view the apps and associated

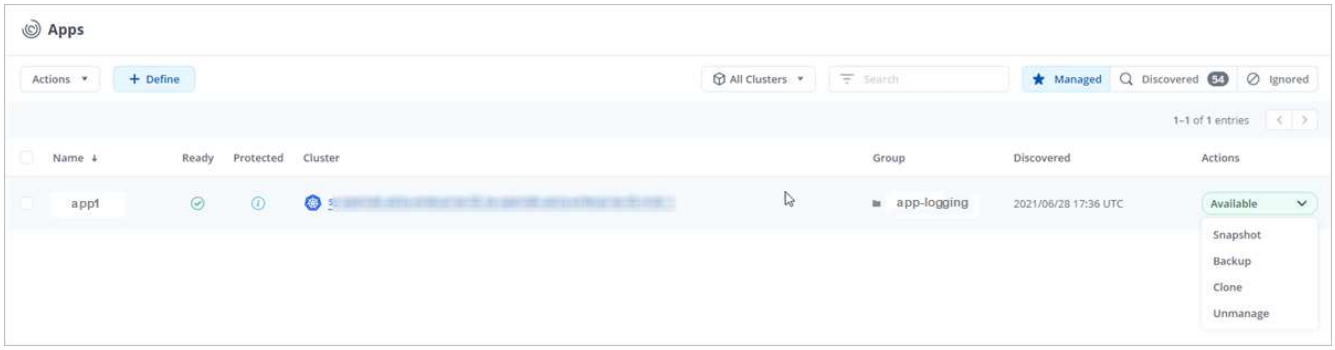
resources.

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

4. Look at the **Group** column to see which namespace the application is running in (it's designated with the folder icon).
5. Decide whether you want to manage each app individually or at the namespace level.
6. Find the app you want at the desired level in the hierarchy, and from the Actions menu, select **Manage**.
7. If you don't want to manage an app, from the Actions menu next to the app, select **Ignore**.

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

8. To see the list of managed apps, select **Managed** as the display filter.



   Notice the app you just added has a warning icon under the Protected column, indicating that it is not backed up and not scheduled for backups yet.

9. To see details of a particular app, select the app name.

Result

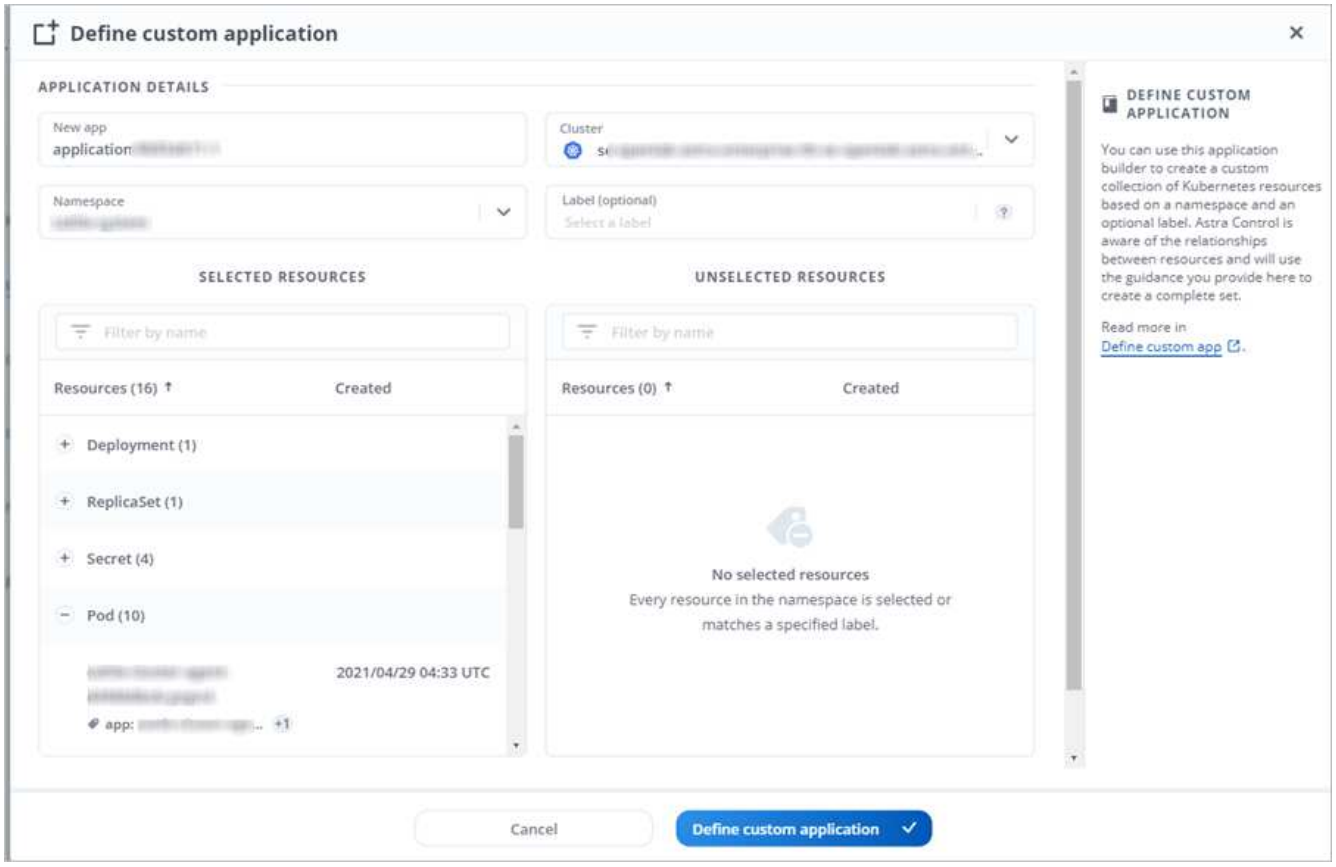
Apps that you chose to manage are now available from the **Managed** tab. Any ignored apps will move to the **Ignored** tab. Ideally, the Discovered tab will show zero apps, so that as new apps are installed, they are easier to find and manage.

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 custom apps by Kubernetes label.

Steps

1. From the left navigation bar, select **Applications**.
2. Select **Define**.



3. 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).
      - 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.
4. Select **Define custom application**.

Result

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

Ignore apps

If an app has been discovered, it appears in the Discovered list. In this case, you can clean up the Discovered list so that new apps that are newly installed are easier to find. Or, you might have apps that you are managing and later decide you no longer want to manage them. If you don't want to manage these apps, you can indicate that they should be ignored.

Also, you might want to manage apps under one namespace together (Namespace-managed). You can ignore apps that you want to exclude from the namespace.

Steps

1. From the left navigation bar, select **Applications**.
2. Select **Discovered** as the filter.
3. Select the app.
4. From the Actions menu, select **Ignore**.
5. To unignore, from the Actions menu, select **Unignore**.

Unmanage apps

When you no longer want to back up, snapshot, or clone an app, you can stop managing it.

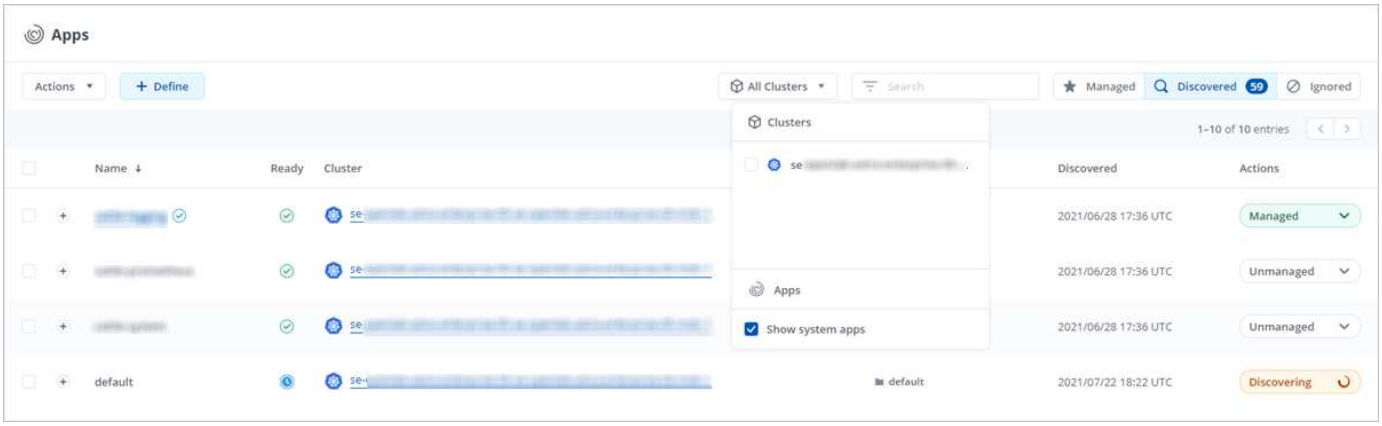
If you unmanage an app, any backups or snapshots that were created earlier will be lost.

Steps

1. From the left navigation bar, select **Applications**.
2. Select **Managed** as the filter.
3. Select the app.
4. From the Actions menu, select **Unmanage**.
5. Review the information.
6. Type "unmanage" to confirm.
7. Select **Yes, Unmanage Application**.

What about system apps?

Astra Control also discovers the system apps running on a Kubernetes cluster. You can display system apps by selecting the **Show system apps** checkbox under the Cluster filter in the toolbar.



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

Astra Control itself is not a standard app; it is a "system app." You should not try to manage Astra Control itself. Astra Control itself isn’t shown by default for management. To see system apps, use the “Show system apps” filter.

Find more information

- Use the Astra Control API

Define a custom app example

Creating a custom app lets you group elements of your Kubernetes cluster into a single app.

A custom app gives you more granular control over what to include in an Astra Control operation, including:

- Clone
- Snapshot
- Backup
- Protection Policy

In most cases you will want to use Astra Control's features on your entire app. However, you can also create a custom app to use these features by the labels you assign to Kubernetes objects in a namespace.

To create a custom app, go to the Apps page and select **+ Define**.

As you make your selections, the Custom App window shows you which resources will be included or excluded from your custom app. This helps you make sure you are choosing the correct criteria for defining your custom app.

Custom apps can be created only within a specified namespace on a single cluster. Astra Control does not support the ability for a custom app to span multiple namespaces or clusters.

A label is a key/value pair you can assign to Kubernetes objects for identification. Labels make it easier to sort, organize, and find your Kubernetes objects. To learn more about Kubernetes labels, see the official Kubernetes documentation.

Overlapping policies for the same resource under different names can cause data conflicts. If you create a custom app for a resource, be sure it's not being cloned or backed up under any other policies.

Example: Separate Protection Policy for canary release

In this example, the devops team is managing a canary release deployment. Their cluster has three pods running NginX. Two of the pods are dedicated to the stable release. The third pod is for the canary release.

The devops team's Kubernetes admin adds the label `deployment=stable` to the stable release pods. The team adds the label `deployment=canary` to the canary release pod.

The team's stable release includes a requirement for hourly snapshots and daily backups. The canary release is more ephemeral, so they want to create a less aggressive, short-term Protection Policy for anything labeled `deployment=canary`.

In order to avoid possible data conflicts, the admin will create two custom apps: one for the canary release, and one for the stable release. This keeps the backups, snapshots, and clone operations separate for the two groups of Kubernetes objects.

Steps

1. After the team adds the cluster to Astra Control, the next step is to define a custom app. To do this, the team selects the **+ Define** button on the Apps page.
2. In the pop-up window which appears, the team sets `devops-canary-deployment` as the app name. The team chooses the cluster in the **Cluster** drop-down, then the app's namespace from the **Namespace** drop-down.
3. The team can either type `deployment=canary` in the **Labels** field, or select that label from the resources listed below.
4. After defining the custom app for the canary deployment, the team repeats the process for the stable deployment.

When the team has finished creating the two custom apps, they can treat these resources as any other Astra Control application. They can clone them, create backups and snapshots, and create a custom Protection Policy for each group of resources based on the Kubernetes labels.

Protect apps

Protection overview

You can create backups, clones, snapshots, and protection policies for your apps using Astra Control Center. Backing up your apps helps your services and associated data be as available as possible; during a disaster scenario, restoring from backup can ensure full recovery of an app and its associated data with minimal disruption. Backups, clones, and snapshots can help protect against common threats such as ransomware, accidental data loss, and environmental disasters. Learn about the available types of data protection in Astra Control Center, and when to use them.

App protection workflow

You can use the following example workflow to get started protecting your apps.

[One] Back up all apps

To make sure that your apps are immediately protected, create a manual backup of all apps.

[Two] Configure a protection policy for each app

To automate future backups and snapshots, configure a protection policy for each app. As an example, you can start with weekly backups and daily snapshots, with one month retention for both. Automating backups and snapshots with a protection policy is strongly recommended over manual backups and snapshots.

[Three] Optional: Adjust the protection policies

As apps and their usage patterns change, adjust the protection policies as needed to provide the best protection.

[Four] In case of a disaster, restore your apps

If data loss occurs, you can recover by restoring the latest backup first for each app. You can then restore the latest snapshot (if available).

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. You can use the Astra UI or the Astra Control API to protect apps.

If you use Helm to deploy apps, Astra Control Center requires Helm version 3. Managing and cloning apps deployed with Helm 3 (or upgraded from Helm 2 to Helm 3) are fully supported. Apps deployed with Helm 2 are not supported.

When you create a project for hosting an app on an OpenShift cluster, the project (or Kubernetes namespace) is assigned a SecurityContext UID. To enable Astra Control Center to protect your app and move the app to another cluster or project in OpenShift, you need to add policies that enable the app to run as any UID. As an example, the following OpenShift CLI commands grant the appropriate policies to a WordPress app.

```
oc new-project wordpress
oc adm policy add-scc-to-group anyuid system:serviceaccounts:wordpress
oc adm policy add-scc-to-user privileged -z default -n wordpress
```

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. As an example, a protection policy might create weekly backups and daily snapshots, and retain the backups and snapshots for one month. How often you create snapshots and backups and how long you retain them depends on the needs of your organization.

Steps

1. Select **Applications** and then select the name of an 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 hourly, daily, weekly, and monthly.

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

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

5. Select **Review**.
6. Select **Set Protection Policy.**

Result

Astra Control Center 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**.
5. Review the snapshot summary and select **Snapshot**.

Result

The snapshot process begins. A snapshot is successful when the status is **Available** in the **Actions** column on the **Data protection** > **Snapshots** page.

Create a backup

You can also back up an app at any time.

> S3 buckets in Astra Control Center do not report available capacity. Before backing up or cloning apps managed by Astra Control Center, check bucket information in the ONTAP or StorageGRID management system.

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.
5. Choose whether to back up the app from an existing snapshot. If you select this option, you can choose from a list of existing snapshots.
6. Choose a destination for the backup by selecting from the list of storage buckets.
7. Select **Review**.
8. Review the backup summary and select **Backup**.

Result

Astra Control Center creates a backup of the app.

> If your network has an outage or is abnormally slow, a backup operation might time out. This causes the backup to fail.

> There is no way to stop a running backup. If you need to delete the backup, wait until it has completed and then use the instructions in Delete backups. To delete a failed backup, use the Astra Control API.

After a data protection operation (clone, backup, restore) and subsequent persistent volume resize, there is up to a twenty-minute delay before the new volume size is shown in the UI. The data protection operation is successful within minutes, and you can use the management software for the storage backend to confirm the change in volume size.

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 an app.
2. Select **Data Protection**.

   The snapshots display by default.

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 an app.
2. Select **Data Protection**.
3. Select the drop-down list in the **Actions** column for the desired snapshot.
4. Select **Delete snapshot**.
5. Type the word "delete" to confirm deletion and then select **Yes, Delete snapshot**.

Result

Astra Control Center deletes the snapshot.

Delete backups

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

There is no way to stop a running backup. If you need to delete the backup, wait until it has completed and then use these instructions. To delete a failed backup, use the Astra Control API.

1. Select **Applications** and then select the name of an 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 word "delete" to confirm deletion and then select **Yes, Delete backup**.

Result

Astra Control Center deletes the backup.

Restore apps

Astra Control can restore your application from a snapshot or backup. Restoring from an existing snapshot will be faster when restoring the application to the same cluster. You can use the Astra Control UI or the Astra Control API to restore apps.

If you use Helm to deploy apps, Astra Control Center requires Helm version 3. Managing and cloning apps deployed with Helm 3 (or upgraded from Helm 2 to Helm 3) are fully supported. Apps deployed with Helm 2 are not supported.

If you restore to a different cluster, ensure that the cluster is using the same persistent volume access mode (for example, ReadWriteMany). The restore operation will fail if the destination persistent volume access mode is different.

When you create a project for hosting an app on an OpenShift cluster, the project (or Kubernetes namespace) is assigned a SecurityContext UID. To enable Astra Control Center to protect your app and move the app to another cluster or project in OpenShift, you need to add policies that enable the app to run as any UID. As an example, the following OpenShift CLI commands grant the appropriate policies to a WordPress app.

```
oc new-project wordpress
oc adm policy add-scc-to-group anyuid system:serviceaccounts:wordpress
oc adm policy add-scc-to-user privileged -z default -n wordpress
```

Steps

1. Select **Applications** and then select the name of an app.
2. Select **Data protection**.
3. If you want to restore from a snapshot, keep the **Snapshots** icon selected. Otherwise, select the **Backups** icon 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. By default, the current cluster and namespace appear. Leave these values intact to restore an app in-place, which reverts the app to an earlier version of itself. Change these values if you want to restore to a different cluster or namespace.
   - Enter a name and namespace for the app.
   - Choose the destination cluster for the app.
   - Select **Review**.

1. **Restore Summary**: Review details about the restore action, type "restore", and select **Restore**.

Result

Astra Control Center restores the app based on the information that you provided. If you restored the app in-place, the contents of any existing persistent volumes are replaced with the contents of persistent volumes from the restored app.

After a data protection operation (clone, backup, restore) and subsequent persistent volume resize, there is up to a twenty-minute delay before the new volume size is shown in the UI. The data protection operation is successful within minutes, and you can use the management software for the storage backend to confirm the change in volume size.

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. You can use the Astra UI or the Astra Control API to clone and migrate apps.

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.

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 clone an app between clusters, the source and destination clusters must be the same distribution of OpenShift. For example, if you clone an app from an OpenShift 4.7 cluster, use a destination cluster that is also OpenShift 4.7.

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

S3 buckets in Astra Control Center do not report available capacity. Before backing up or cloning apps managed by Astra Control Center, check bucket information in the ONTAP or StorageGRID management system.

When you create a project for hosting an app on an OpenShift cluster, the project (or Kubernetes namespace) is assigned a SecurityContext UID. To enable Astra Control Center to protect your app and move the app to another cluster or project in OpenShift, you need to add policies that enable the app to run as any UID. As an example, the following OpenShift CLI commands grant the appropriate policies to a WordPress app.

```
oc new-project wordpress
oc adm policy add-scc-to-group anyuid system:serviceaccounts:wordpress
oc adm policy add-scc-to-user privileged -z default -n wordpress
```

What you'll need

To clone apps to a different cluster, you need a default bucket. When you add your first bucket, it becomes the default bucket.

Steps

1. Select **Applications**.
2. Do one of the following:

- Select the drop-down list in the **Actions** column for the desired app.
- Select the name of the desired app, and select the status drop-down list at the top right of the page.

3. Select **Clone**.
4. **Clone details**: Specify details for the clone:
   - Enter a name.
   - Enter a namespace for the clone.
   - 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 Center creates the clone from the app's current state.
5. **Source**: If you chose to clone from an existing snapshot or backup, choose the snapshot or backup that you'd like to use.
6. Select **Review**.
7. **Clone Summary**: Review the details about the clone and select **Clone**.

Result

Astra Control Center clones that app based on the information that you provided. The clone operation is successful when the new app clone is in the `Available` state on the **Applications** page.

> After a data protection operation (clone, backup, restore) and subsequent persistent volume resize, there is up to a twenty-minute delay before the new volume size is shown in the UI. The data protection operation is successful within minutes, and you can use the management software for the storage backend to confirm the change in volume size.

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
#

#
```

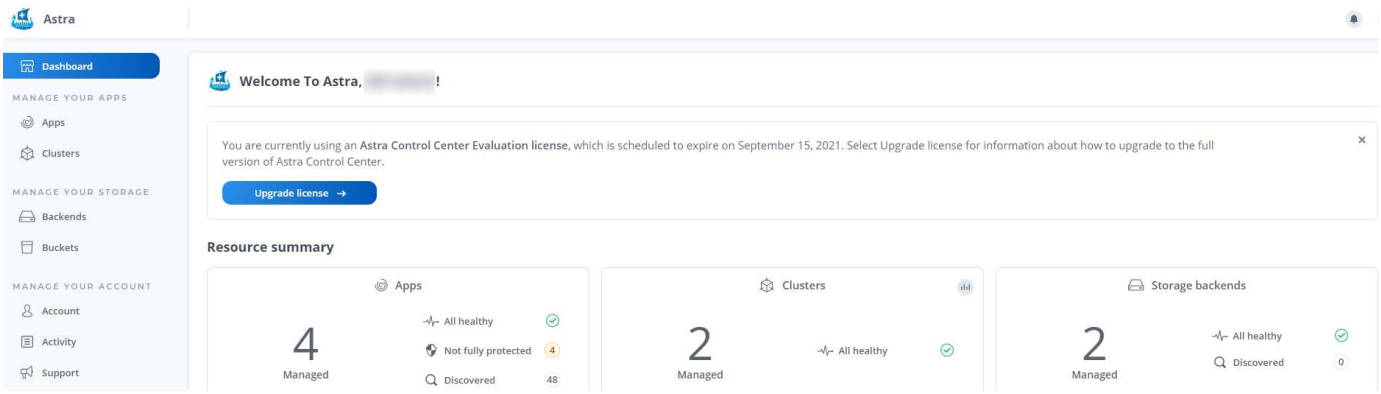
```
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# 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 cluster health

View a summary of app and cluster health

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



These aren't just static 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.

Applications tile

The **Applications** tile helps you identify the following:

- How many apps you're currently managing with Astra.
- 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.

Clusters tile

The **Clusters** tile provides similar details about the health of the clusters that you are managing by using Astra Control Center, and you can drill down to get more details just like you can with an app.

Storage backends tile

The **Storage backends** tile provides information to help you identify the health of storage backends including:

- How many storage backends are managed
- Whether these managed backends are healthy

- Whether the backends are fully protected
- The number of backends that are discovered, but are not yet managed.

View the health and details of clusters

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

Steps

1. In the Astra Control Center UI, select **Clusters**.
2. On the **Clusters** page, select the cluster whose details you want to view.
3. View the information on the **Overview**, **Storage**, and **Activity** 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 compute, including the storage class and state.
   - **Activity**: Shows the activities related to the cluster.

> ⓘ You can also view cluster information starting from the Astra Control Center **Dashboard**. On the **Clusters** tab under **Resource summary**, you can select the managed clusters, which takes you to the **Clusters** page. After you get to the **Clusters** page, follow the steps outlined above.

View the health and details of an app

After you start managing an app, Astra 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.

Steps

1. In the Astra Control Center UI, select **Applications** and then select the name of an app.
2. 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 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
Shows the activities related to the app.

You can also view app information starting from the Astra Control Center **Dashboard**. On the **Applications** tab under **Resource summary**, you can select the managed apps, which takes you to the **Applications** page. After you get to the **Applications** page, follow the steps outlined above.

Manage your account

Manage users

You can add, remove, and edit users of your Astra Control Center installation using the Astra Control Center UI. You can use the Astra UI or the Astra Control API to manage users.

Add users

Account Owners and Admins can add more users to the Astra Control Center installation.

Steps

1. In the **Manage Your Account** navigation area, select **Account**.
2. Select the **Users** tab.
3. Select **Add User**.
4. Enter the user's name, email address, and a temporary password.

   The user will need to change the password upon first login.

5. Select a user role with the appropriate system permissions.

   Each role provides the following permissions:

   - A **Viewer** can view resources.

- A **Member** has Viewer role permissions and can manage apps and clusters, but cannot unmanage apps or clusters, or delete snapshots or backups.
- An **Admin** has Member role permissions and can add and remove any other users except the Owner.
- An **Owner** has Admin role permissions and can add and remove any user accounts.

6. Select **Add**.

Manage passwords

You can manage passwords for user accounts in Astra Control Center.

Change your password

You can change the password of your user account at any time.

Steps

1. Select the User icon at the top right of the screen.
2. Select **Profile**.
3. Select the **Actions** drop-down list, and select **Change Password**.
4. Enter a password that conforms to the password requirements.
5. Enter the password again to confirm.
6. Select **Change password**.

Reset another user's password

If your account has Admin or Owner role permissions, you can reset passwords for other user accounts as well as your own. When you reset a password, you assign a temporary password that the user will have to change upon logging in.

Steps

1. In the **Manage Your Account** navigation area, select **Account**.
2. In the **Users** tab, select the drop-down list in the **State** column for the user.
3. Select **Reset Password**.
4. Enter a temporary password that conforms to the password requirements.
5. Enter the password again to confirm.

   Next time the user logs in, the user will be prompted to change the password.

6. Select **Reset password**.

Change a user's role

Users with the Owner role can change the role of all users, while users with the Admin role can change the role of users who have the Admin, Member, or Viewer role.

Steps

1. In the **Manage Your Account** navigation area, 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 Center updates the user's permissions based on the new role that you selected.

Remove users

Users with the Owner or Admin role can remove other users from the account at any time.

Steps

1. In the **Manage Your Account** navigation area, select **Account**.
2. In the **Users** tab, select the checkbox in the row of each user that you want to remove.
3. Select **Actions** and select **Remove user/s**.
4. When you're prompted, confirm deletion by typing the word "remove" and then select **Yes, Remove User**.

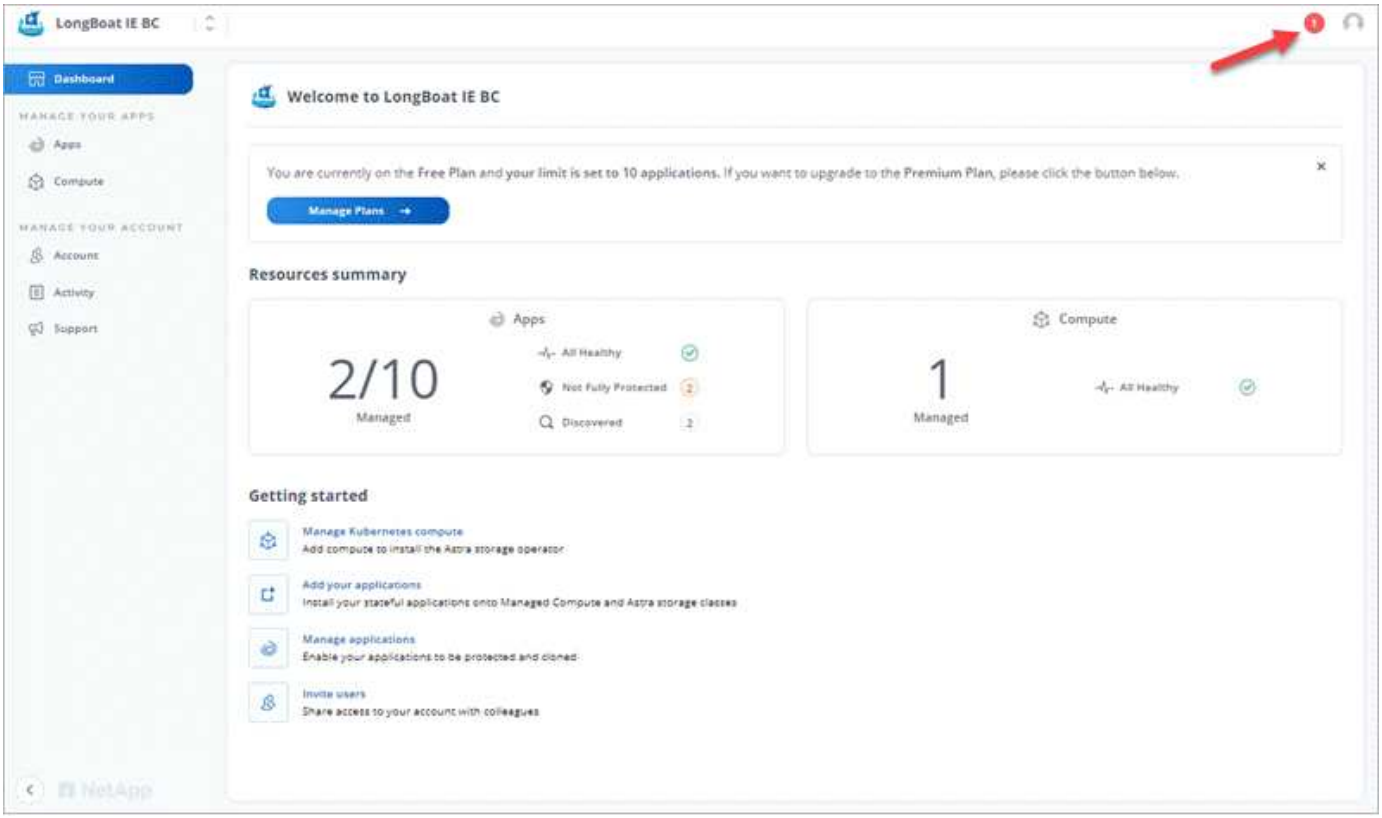
Result

Astra Control Center removes the user from the account.

View and manage notifications

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

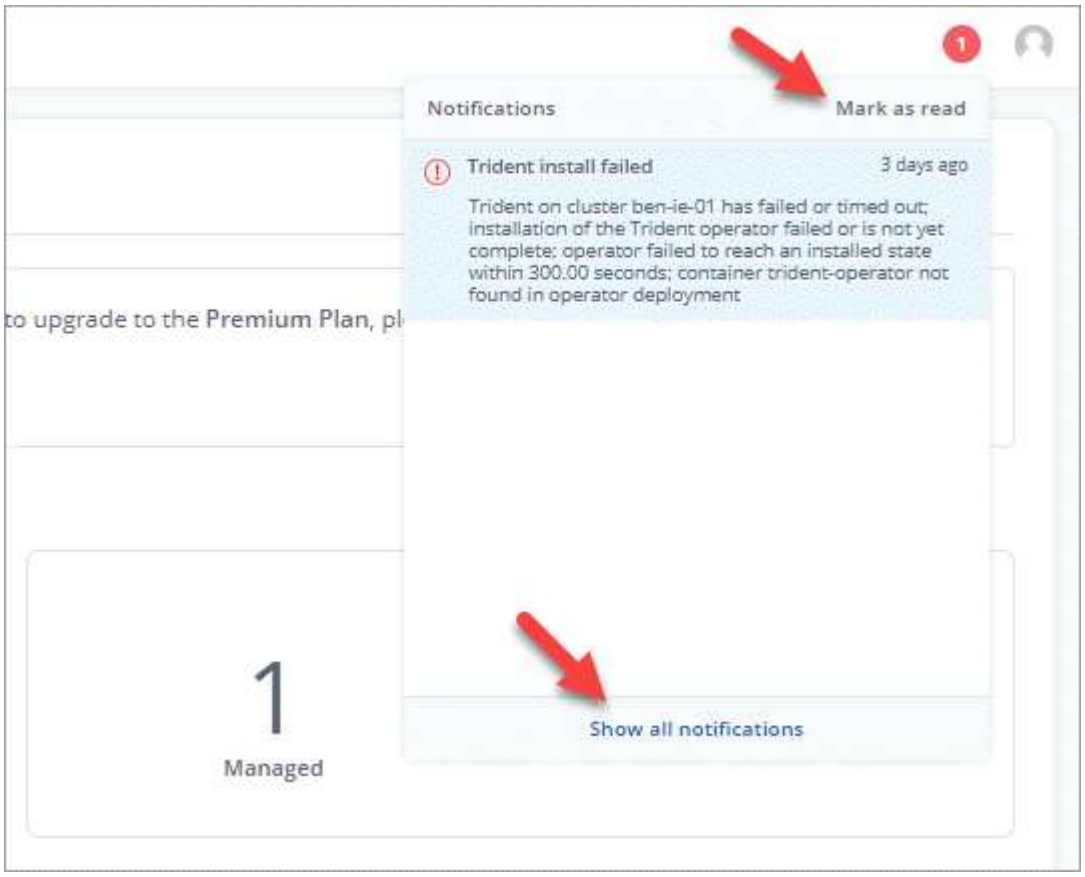
The number of unread notifications is available in the top right of the interface:



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**.

Add and remove credentials

Add and remove credentials for local private cloud providers such as ONTAP S3, Kubernetes clusters managed with OpenShift, or unmanaged Kubernetes clusters from your account at any time. Astra Control Center uses these credentials to discover Kubernetes clusters and the apps on the clusters, and to provision resources on your behalf.

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

Add credentials

You can add credentials to Astra Control Center when you manage clusters. To add credentials by adding a new cluster, see Add a Kubernetes cluster.

> If you create your own `kubeconfig` file, you should define only **one** context element in it. See Kubernetes documentation for information about creating `kubeconfig` files.

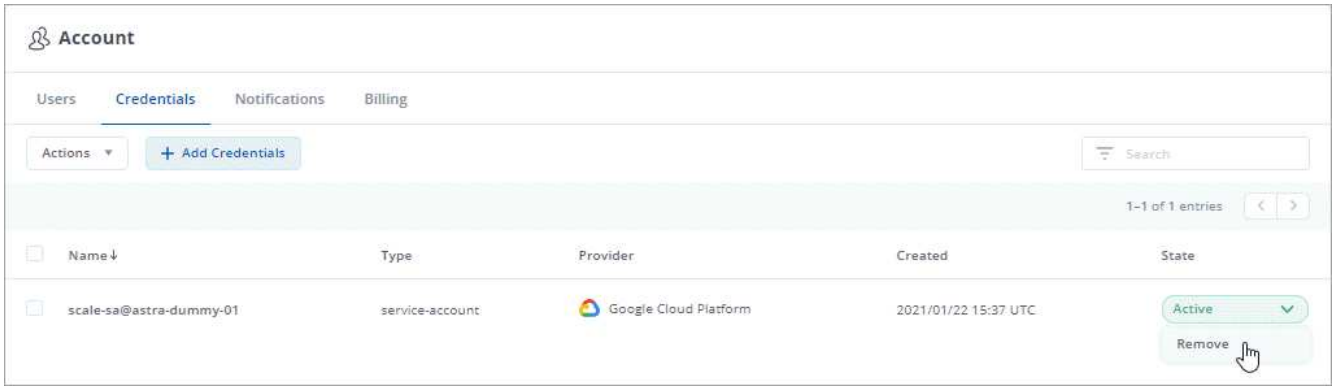
Remove credentials

Remove credentials from an account at any time. You should only remove credentials after unmanaging all associated clusters.

> The first set of credentials that you add to Astra Control Center is always in use because Astra Control Center 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 word "remove" to confirm deletion and then select **Yes, Remove Credential**.

Result

Astra Control Center removes the credentials from the account.

Monitor 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.

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.

View account activity for a specific app

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

View account activity for clusters

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

2. Select **Activity**.

Take action to resolve events that require attention

1. Select **Activity**.
2. Select an event that requires attention.
3. Select the **Take action** drop-down option.

   From this list, you can view possible corrective actions that you can take, view documentation related to the issue, and get support to help resolve the issue.

Update an existing license

You can convert an evaluation license to a full license, or you can update an existing evaluation or full license with a new license. If you don't have a full license, work with your NetApp sales contact to obtain a full license and serial number. You can use the Astra UI or the Astra Control API to update an existing license.

Steps

1. Log in to the NetApp Support Site.
2. Access the Astra Control Center Download page, enter the serial number, and download the full NetApp license file (NLF).
3. Log in to the Astra Control Center UI.
4. From the left navigation, select **Account** > **License**.
5. In the **Account** > **License** page, select the status drop-down menu for the existing license and select **Replace**.
6. Browse to the license file that you downloaded.
7. Select **Add**.

The **Account** > **Licenses** page displays the license information, expiration date, license serial number, account ID, and CPU units used.

Manage buckets

An object store bucket provider is essential if you want to back up your applications and persistent storage or if you want to clone applications across clusters. Using Astra Control Center, add an object store provider as your off-cluster, backup destination for your apps.

You don't need a bucket if you are cloning your application configuration and persistent storage to the same cluster.

Use any of the following bucket providers:

- NetApp ONTAP S3
- NetApp StorageGRID S3
- Generic S3

> Although Astra Control Center supports Amazon S3 as a Generic S3 bucket provider, Astra Control Center might not support all object store vendors that claim Amazon's S3 support.

You cannot delete a bucket; however, you can edit it.

A bucket can be in one of these states:

- pending: The bucket is scheduled for discovery.
- available: The bucket is available for use.
- removed: The bucket is not currently accessible.

For instructions on how to manage buckets using the Astra Control API, see the Astra Automation and API information.

You can do these tasks related to managing buckets:

- Add a bucket
- Edit a bucket

> S3 buckets in Astra Control Center do not report available capacity. Before backing up or cloning apps managed by Astra Control Center, check bucket information in the ONTAP or StorageGRID management system.

Remove credentials

Remove S3 credentials from an account at any time using the Astra Control API.

For details, see Use the Astra Control API.

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

Edit a bucket

You can change the access credential information for a bucket and change whether a selected bucket is the default bucket.

> When you add a bucket, select the correct bucket provider and provide the right credentials for that provider. For example, the UI accepts NetApp ONTAP S3 as the type and accepts StorageGRID credentials; however, this will cause all future app backups and restores using this bucket to fail. See the Release Notes.

Steps

1. From left navigation, select **Buckets**.
2. From the Actions menu, select **Edit**.
3. Change any information other than the bucket type.

   > You can't modify the bucket type.

4. Select **Update**.

Find more information

- Use the Astra Control API

Manage the storage backend

Managing storage clusters in Astra Control as a storage backend enables you to get linkages between persistent volumes (PVs) and the storage backend as well as additional storage metrics. You can monitor storage capacity and health details, including performance if Astra Control Center is connected to Cloud Insights.

For instructions on how to manage storage backends using the Astra Control API, see the Astra Automation and API information.

You can complete the following tasks related to managing a storage backend:

- Add a storage backend
- View storage backend details
- Unmanage a storage backend

View storage backend details

You can view storage backend information from the Dashboard or from the Backends option.

View storage backend details from the Dashboard

Steps

1. From the left navigation, select **Dashboard**.
2. Review the Storage backend section that shows the state:
    - **Unhealthy**: The storage is not in an optimal state. This could be due to a latency issue or an app is degraded due to a container issue, for example.
    - **All healthy**: The storage has been managed and is in an optimal state.
    - **Discovered**: The storage has been discovered, but not managed by Astra Control.

View storage backend details from the Backends option

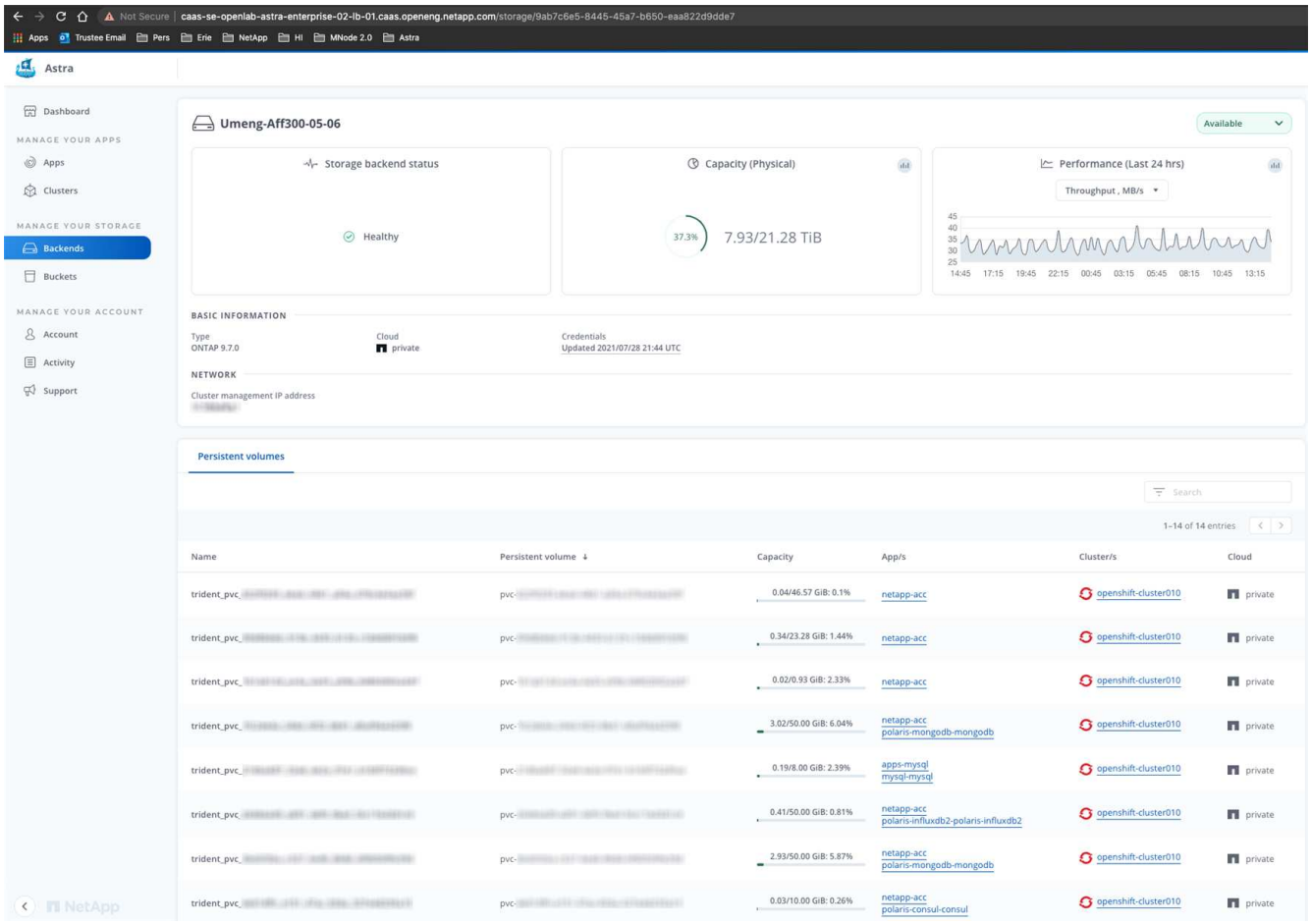
View information about the backend health, capacity, and performance (IOPS throughput and/or latency).

With a connection to Cloud Insights, you can see the volumes that the Kubernetes apps are using, which are stored on a selected storage backend.

Steps

1. In the left navigation area, select **Backends**.
2. Select the storage backend.

If you connected to NetApp Cloud Insights, excerpts of data from Cloud Insights appear on the Backends page.



3. To go directly to Cloud Insights, select the **Cloud Insights** icon next to the metrics image.

Unmanage a storage backend

You can unmanage the backend.

Steps

1. From left navigation, select **Backends**.
2. Select the storage backend.
3. From the Actions menu, select **Unmanage**.
4. Type "unmanage" to confirm the removal.
5. Select **Yes, remove storage backend**.

Find more information

- Use the Astra Control API

Monitor and protect infrastructure

You can configure several optional settings to enhance your Astra Control Center experience. If the network where you're running Astra Control Center requires a proxy for connecting to the Internet (to upload support bundles to NetApp Support Site or establish a connection to Cloud Insights), you should configure a proxy server in Astra Control Center. To monitor and gain insight into your complete infrastructure, create a

connection to NetApp Cloud Insights. To collect Kubernetes events from systems monitored by Astra Control Center, add a Fluentd connection.

After you enable the Cloud Insights connection, you can view throughput information on the **Backends** page as well as connect to Cloud Insights from here after selecting a storage backend. You can also find the information on the **Dashboard** in the Cluster section, and also connect to Cloud Insights from here.

Add a proxy server

If the network where you're running Astra Control Center requires a proxy for connecting to the Internet (to upload support bundles to NetApp Support Site or establish a connection to Cloud Insights), you should configure a proxy server in Astra Control Center.

Astra Control Center does not validate the details you enter for your proxy server. Ensure that you enter the correct values.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Connect** from the drop-down list to add a proxy server.

4. Enter the proxy server name or IP address and the proxy port number.
5. If your proxy server requires authentication, select the checkbox, and enter the username and password.
6. Select **Connect**.

Result

If the proxy information you entered was saved, the **HTTP Proxy** section of the **Account** > **Connections** page indicates that it is connected, and displays the server name.

Edit proxy server settings

You can edit the proxy server settings.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Edit** from the drop-down list to edit the connection.
4. Edit the server details and authentication information.
5. Select **Save**.

Disable proxy server connection

You can disable the proxy server connection. You will be warned before you disable that potential disruption to other connections might occur.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Disconnect** from the drop-down list to disable the connection.
4. In the dialog box that opens, confirm the operation.

Connect to Cloud Insights

To monitor and gain insight into your complete infrastructure, connect NetApp Cloud Insights with your Astra Control Center instance. Cloud Insights is included in your Astra Control Center license.

Cloud Insights should be accessible from the network that Astra Control Center uses, or indirectly via a proxy server.

When Astra Control Center is connected to Cloud Insights, an Acquisition Unit pod gets created. This pod collects data from the storage backends that are managed by Astra Control Center and pushes it to Cloud Insights. This pod requires 8 GB RAM and 2 CPU cores.

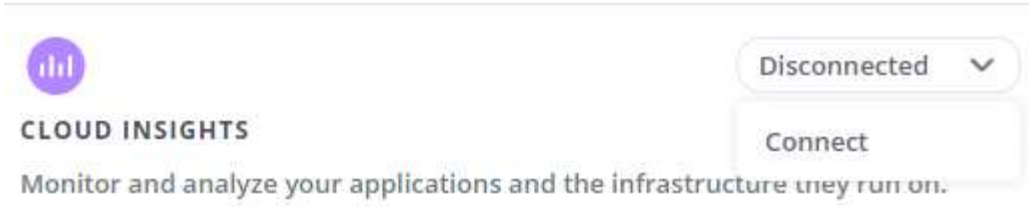
What you'll need

- An Astra Control Center account with **admin**/**owner** privileges.
- A valid Astra Control Center license.
- A proxy server if the network where you're running Astra Control Center requires a proxy for connecting to the Internet.

If you are new to Cloud Insights, familiarize yourself with the features and capabilities here.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Connect** where it shows **Disconnected** in the drop-down list to add the connection.



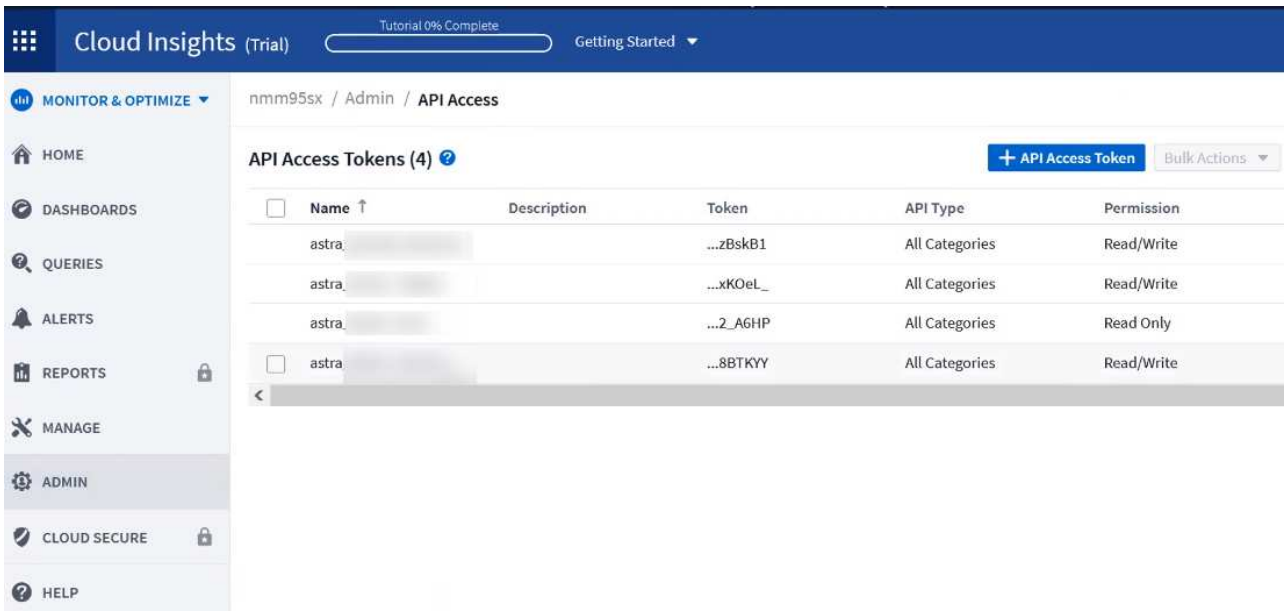
4. Enter the Cloud Insights API tokens and the tenant URL. The tenant URL has the following format, as an example:

```
https://<environment-name>.c01.cloudinsights.netapp.com/
```

You get the tenant URL when you get the Cloud Insights license. If you do not have the tenant URL, see the Cloud Insights documentation.

a. To get the API token, log in to your Cloud Insights tenant URL.

b. In Cloud Insights, generate a **Read only** type API token.



c. Copy the **Read only** key. You will need to paste it into the Astra Control Center window for enabling the Cloud Insights connection.

d. In Cloud Insights, generate a **Read/Write** type API token.

e. Copy the **Read/Write** key. You will need to paste it into the Astra Control Center **Connect Cloud Insights** window.

> ⓘ We recommend that you generate a **Read only** key and a **Read/Write** key, and not use the same key for both purposes. By default, the token expiry period is set to one year. We recommend that you keep the default selection to give the token the maximum duration before it expires. If your token expires, the telemetry will stop.

f. Paste the keys that you copied from Cloud Insights into Astra Control Center.

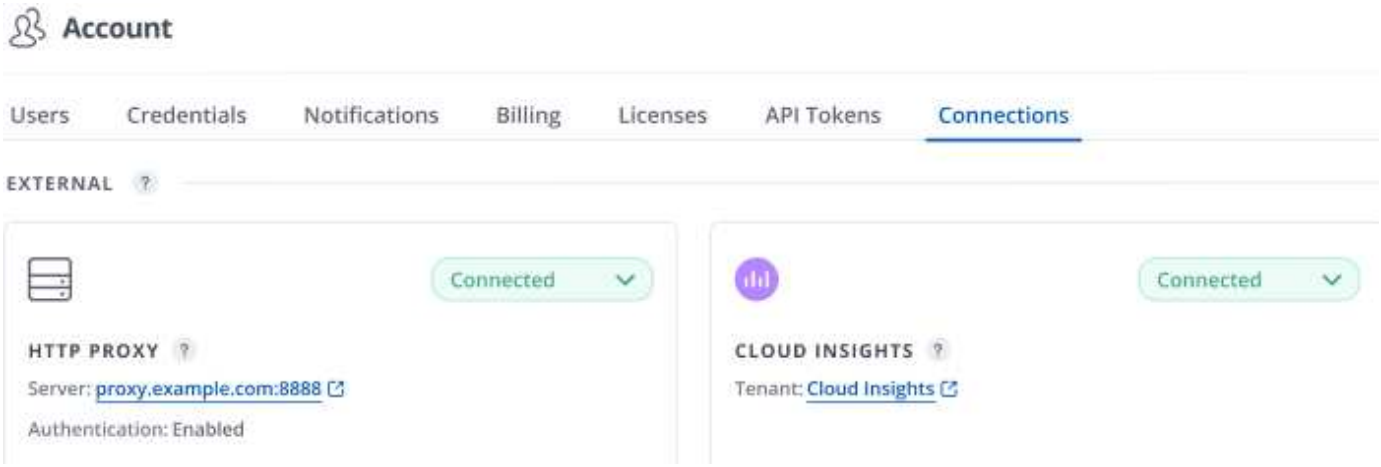
5. Select **Connect**.

After you select **Connect,** the status of the connection changes to **Pending** in the **Cloud Insights** section of the **Account** > **Connections** page. It can a few minutes for the connection to be enabled and the status to change to **Connected**.

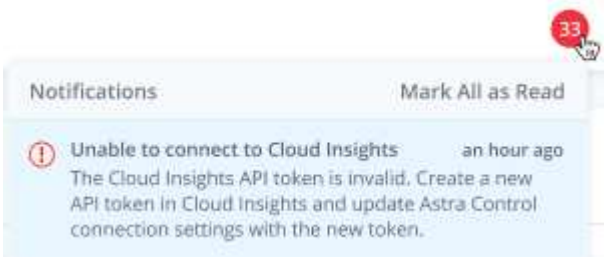
To go back and forth easily between the Astra Control Center and Cloud Insights UIs, ensure that you are logged into both.

View data in Cloud Insights

If the connection was successful, the **Cloud Insights** section of the **Account** > **Connections** page indicates that it is connected, and displays the tenant URL. You can visit Cloud Insights to see data being successfully received and displayed.

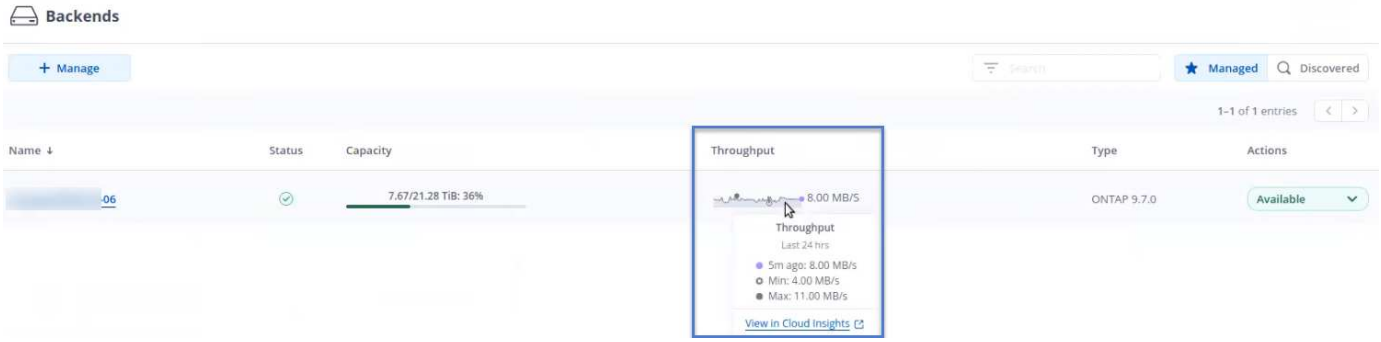


If the connection failed for some reason, the status shows **Failed**. You can find the reason for failure under **Notifications** at the top-right side of the UI.



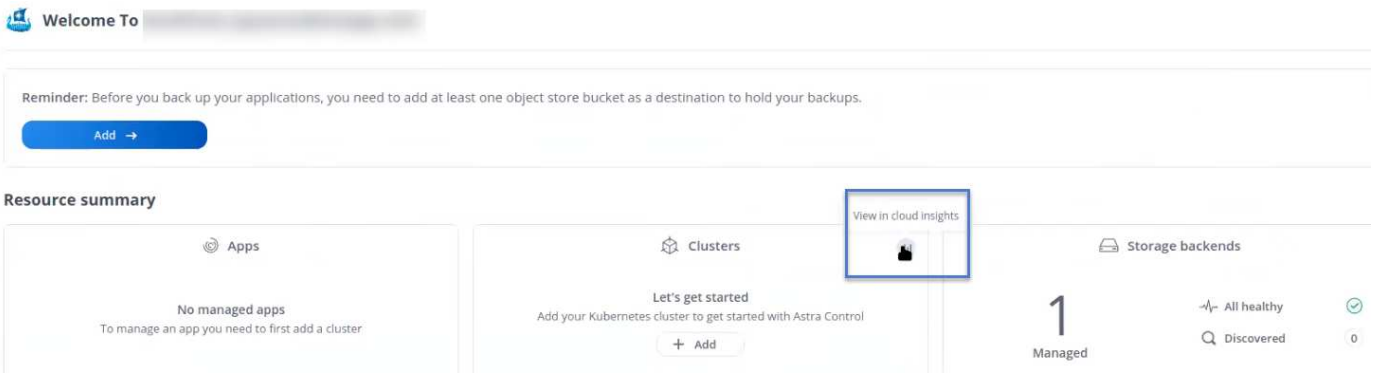
You can also find the same information under **Account** > **Notifications**.

From Astra Control Center, you can view throughput information on the **Backends** page as well as connect to Cloud Insights from here after selecting a storage backend.



To go directly to Cloud Insights, select the **Cloud Insights** icon next to the metrics image.

You can also find the information on the **Dashboard**.



After enabling the Cloud Insights connection, if you remove the backends that you added in Astra Control Center, the backends stop reporting to Cloud Insights.

Edit Cloud Insights connection

You can edit the Cloud Insights connection.

You can only edit the API keys. To change the Cloud Insights tenant URL, we recommended that you disconnect the Cloud Insights connection, and connect with the new URL.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Edit** from the drop-down list to edit the connection.
4. Edit the Cloud Insights connection settings.
5. Select **Save**.

Disable Cloud Insights connection

You can disable the Cloud Insights connection for a Kubernetes cluster managed by Astra Control Center. Disabling the Cloud Insights connection does not delete the telemetry data already uploaded to Cloud Insights.

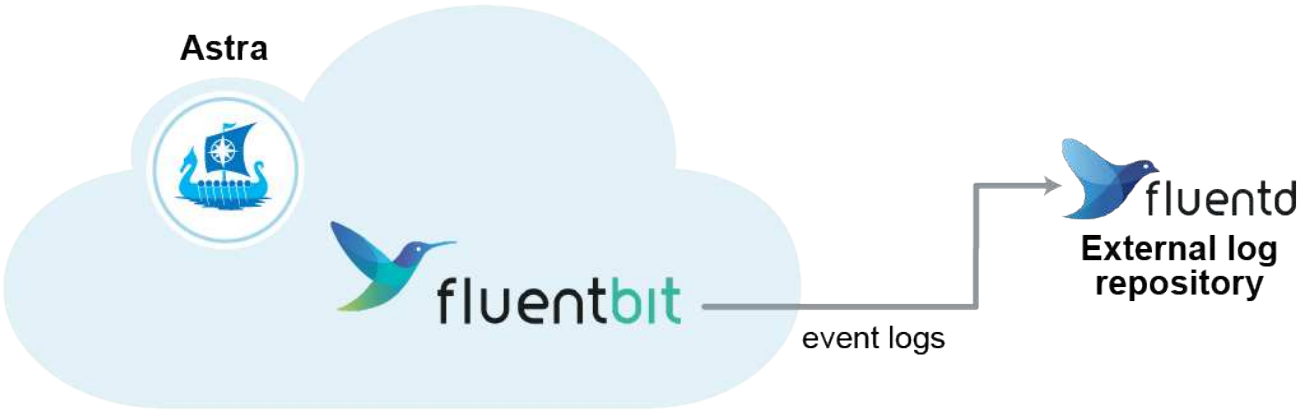
Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Disconnect** from the drop-down list to disable the connection.
4. In the dialog box that opens, confirm the operation.
   After you confirm the operation, on the **Account** > **Connections** page, the Cloud Insights status changes to **Pending**. It take a few minutes for the status to change to **Disconnected**.

Connect to Fluentd

You can send logs (Kubernetes events) from Astra Control Center to your Fluentd endpoint. The Fluentd

connection is disabled by default.



Only the event logs from managed clusters are forwarded to Fluentd.

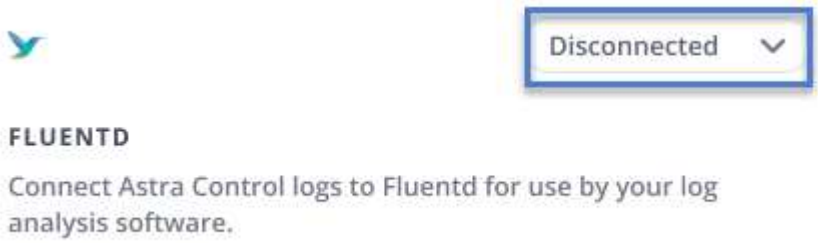
What you'll need

- An Astra Control Center account with **admin**/**owner** privileges.
- Astra Control Center installed and running on a Kubernetes cluster.

Astra Control Center does not validate the details you enter for your Fluentd server. Ensure that you enter the correct values.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Connect** from the drop-down list where it shows **Disconnected** to add the connection.



4. Enter the host IP address, the port number, and shared key for your Fluentd server.
5. Select **Connect**.

Result

If the details you entered for your Fluentd server were saved, the **Fluentd** section of the **Account** > **Connections** page indicates that it is connected. Now you can visit the Fluentd server that you connected and view the event logs.

If the connection failed for some reason, the status shows **Failed**. You can find the reason for failure under **Notifications** at the top-right side of the UI.

You can also find the same information under **Account** > **Notifications**.

> If you are having trouble with log collection, you should log in to your worker node and ensure that your logs are available in `/var/log/containers/`.

Edit the Fluentd connection

You can edit the Fluentd connection to your Astra Control Center instance.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Edit** from the drop-down list to edit the connection.
4. Change the Fluentd endpoint settings.
5. Select **Save**.

Disable the Fluentd connection

You can disable the Fluentd connection to your Astra Control Center instance.

Steps

1. Log in to Astra Control Center using an account with **admin**/**owner** privilege.
2. Select **Account** > **Connections**.
3. Select **Disconnect** from the drop-down list to disable the connection.
4. In the dialog box that opens, confirm the operation.

Update an existing license

You can convert an evaluation license to a full license, or you can update an existing evaluation or full license with a new license. If you don't have a full license, work with your NetApp sales contact to obtain a full license and serial number. You can use the Astra UI or the Astra Control API to update an existing license.

Steps

1. Log in to the NetApp Support Site.
2. Access the Astra Control Center Download page, enter the serial number, and download the full NetApp license file (NLF).
3. Log in to the Astra Control Center UI.
4. From the left navigation, select **Account** > **License**.
5. In the **Account** > **License** page, select the status drop-down menu for the existing license and select **Replace**.
6. Browse to the license file that you downloaded.
7. Select **Add**.

The **Account** > **Licenses** page displays the license information, expiration date, license serial number, account ID, and CPU units used.

Unmanage apps and clusters

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

Unmanage an app

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

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

Steps

1. From the left navigation bar, select **Applications**.
2. Select the checkbox for the apps that you no longer want to manage.
3. From the **Action** menu, select **Unmanage**.
4. Type "unmanage" to confirm.
5. Confirm that you want to unmanage the apps and then select **Yes, unmanage Application**.

Result

Astra Control Center stops managing the app.

Unmanage a cluster

Unmanage the cluster that you no longer want to manage from Astra Control Center.

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

Before you unmanage the cluster, you should unmanage the apps associated with the cluster.

Steps

1. From the left navigation bar, select **Clusters**.
2. Select the checkbox for the cluster that you no longer want to manage in Astra Control Center.
3. From the **Actions** menu, select **Unmanage**.
4. Confirm that you want to unmanage the cluster and then select **Yes, unmanage cluster**.

Result

The status of the cluster changes to **Removing** and after that the cluster will be removed from the **Clusters** page, and it is no longer managed by Astra Control Center.

If Astra Control Center and Cloud Insights are not connected, unmanaging the cluster removes all the resources that were installed for sending telemetry data. **If Astra Control Center and Cloud Insights are connected**, unmanaging the cluster deletes only the `fluentbit` and `event-exporter` pods.

Upgrade Astra Control Center

To upgrade Astra Control Center, download the installation bundle from the NetApp Support Site and complete these instructions to upgrade the Astra Control Center components in your environment. You can use this procedure to upgrade Astra Control Center in internet-connected or air-gapped environments.

What you'll need

- Before you begin upgrade, ensure your environment still meets the minimum requirements for Astra Control Center deployment.
- Ensure all cluster operators are in a healthy state and available.

  OpenShift example:

  ```
  oc get clusteroperators
  ```

- Ensure all API services are in a healthy state and available.

  OpenShift example:

  ```
  oc get apiservices
  ```

- Log out of your Astra Control Center.

About this task

The Astra Control Center upgrade process guides you through the following high-level steps:

- Download the Astra Control Center bundle
- Unpack the bundle and change directory
- Add the images to your local registry
- Install the updated Astra Control Center operator
- Upgrade Astra Control Center
- Upgrade third-party services
- Verify system status

Do not execute the following command during the entirety of the upgrade process to avoid deleting all Astra Control Center pods: `kubectl delete -f astra_control_center_operator_deploy.yaml`

Perform upgrades in a maintenance window when schedules, backups, and snapshots are not running.

Podman commands can be used in place of Docker commands if you are using Red Hat's Podman instead of Docker Engine.

Download the Astra Control Center bundle

1. Download the Astra Control Center upgrade bundle (`astra-control-center-[version].tar.gz`) from the NetApp Support Site.
2. (Optional) Use the following command to verify the signature of the bundle:

```
openssl dgst -sha256 -verify astra-control-center[version].pub
-signature <astra-control-center[version].sig astra-control-
center[version].tar.gz
```

Unpack the bundle and change directory

1. Extract the images:

```
tar -vxzf astra-control-center-[version].tar.gz
```

2. Change to the Astra directory.

```
cd astra-control-center-[version]
```

Add the images to your local registry

1. Add the files in the Astra Control Center image directory to your local registry.

   See a sample script for the automatic loading of images below.

   a. Log in to your Docker registry:

   ```
   docker login [your_registry_path]
   ```

   b. Load the images into Docker.

   c. Tag the images.

   d. Push the images to your local registry.

```
export REGISTRY=[your_registry_path]
for astraImageFile in $(ls images/*.tar)
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  do astraImage=$(docker load --input ${astraImageFile} | sed
's/Loaded image: //')
  astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
  # Tag with local image repo.
  docker tag ${astraImage} ${REGISTRY}/${astraImage}
  # Push to the local repo.
  docker push ${REGISTRY}/${astraImage}
done
```

Install the updated Astra Control Center operator

1. Edit the Astra Control Center operator deployment yaml (`astra_control_center_operator_deploy.yaml`) to refer to your local registry and secret.

```
vim astra_control_center_operator_deploy.yaml
```

   a. If you use a registry that requires authentication, replace the default line of `imagePullSecrets: []` with the following:

```
imagePullSecrets:
- name: <name_of_secret_with_creds_to_local_registry>
```

   b. Change `[your_registry_path]` for the `kube-rbac-proxy` image to the registry path where you pushed the images in a previous step.

   c. Change `[your_registry_path]` for the `acc-operator-controller-manager` image to the registry path where you pushed the images in a previous step.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: [your_registry_path]/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        command:
        - /manager
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

2. Install the updated Astra Control Center operator:

```
kubectl apply -f astra_control_center_operator_deploy.yaml
```

Sample response:

```
namespace/netapp-acc-operator unchanged
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.astra.
netapp.io configured
role.rbac.authorization.k8s.io/acc-operator-leader-election-role
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role
configured
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role unchanged
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding unchanged
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding configured
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding unchanged
configmap/acc-operator-manager-config unchanged
service/acc-operator-controller-manager-metrics-service unchanged
deployment.apps/acc-operator-controller-manager configured
```

Upgrade Astra Control Center

1. Edit the Astra Control Center custom resource (CR) and change the Astra version (`astraVersion` inside of `Spec`) number to the latest:

```
kubectl edit acc -n [netapp-acc or custom namespace]
```

> Changing the Astra version is the only requirement for an Astra Control Center upgrade. Your registry path must match the registry path where you pushed the images in a previous step.

2. Verify that the pods terminate and become available again:

```
watch kubectl get pods -n [netapp-acc or custom namespace]
```

3. Verify that all system components upgraded successfully.

```
kubectl get pods -n [netapp-acc or custom namespace]
```

Each pod should have a status of `Running` and `Age` that is recent. It may take several minutes before the system pods are deployed.

Sample response:

```
NAME                                          READY   STATUS    RESTARTS
AGE
acc-helm-repo-5f75c5f564-bzqmt                1/1     Running   0
11m
activity-6b8f7cccb9-mlrn4                     1/1     Running   0
9m2s
api-token-authentication-6hznt                1/1     Running   0
8m50s
api-token-authentication-qpfgb                1/1     Running   0
8m50s
api-token-authentication-sqnb7                1/1     Running   0
8m50s
asup-5578bbdd57-dxkbp                         1/1     Running   0
9m3s
authentication-56bff4f95d-mspmq               1/1     Running   0
7m31s
bucketservice-6f7968b95d-9rrrl                1/1     Running   0
8m36s
cert-manager-5f6cf4bc4b-82khn                 1/1     Running   0
6m19s
cert-manager-cainjector-76cf976458-sdrbc      1/1     Running   0
6m19s
cert-manager-webhook-5b7896bfd8-2n45j         1/1     Running   0
6m19s
cloud-extension-749d9f684c-8bdhq              1/1     Running   0
9m6s
cloud-insights-service-7d58687d9-h5tzw        1/1     Running   2
8m56s
composite-compute-968c79cb5-nv7l4             1/1     Running   0
9m11s
composite-volume-7687569985-jg9gg             1/1     Running   0
8m33s
credentials-5c9b75f4d6-nx9cz                  1/1     Running   0
8m42s
entitlement-6c96fd8b78-zt7f8                  1/1     Running   0
8m28s
features-5f7bfc9f68-gsjnl                     1/1     Running   0
8m57s
fluent-bit-ds-h88p7                           1/1     Running   0
7m22s
fluent-bit-ds-krhnj                           1/1     Running   0
7m23s
fluent-bit-ds-l5bjj                           1/1     Running   0
7m22s
```

```
fluent-bit-ds-lrclb                          1/1     Running   0
7m23s
fluent-bit-ds-s5t4n                          1/1     Running   0
7m23s
fluent-bit-ds-zpr6v                          1/1     Running   0
7m22s
graphql-server-5f5976f4bd-vbb4z              1/1     Running   0
7m13s
identity-56f78b8f9f-8h9p9                    1/1     Running   0
8m29s
influxdb2-0                                  1/1     Running   0
11m
krakend-6f8d995b4d-5khkl                     1/1     Running   0
7m7s
license-5b5db87c97-jmxzc                     1/1     Running   0
9m
login-ui-57b57c74b8-6xtv7                    1/1     Running   0
7m10s
loki-0                                       1/1     Running   0
11m
monitoring-operator-9dbc9c76d-8znck          2/2     Running   0
7m33s
nats-0                                       1/1     Running   0
11m
nats-1                                       1/1     Running   0
10m
nats-2                                       1/1     Running   0
10m
nautilus-6b9d88bc86-h8kfb                    1/1     Running   0
8m6s
nautilus-6b9d88bc86-vn68r                    1/1     Running   0
8m35s
openapi-b87d77dd8-5dz9h                      1/1     Running   0
9m7s
polaris-consul-consul-5ljfb                  1/1     Running   0
11m
polaris-consul-consul-s5d5z                  1/1     Running   0
11m
polaris-consul-consul-server-0               1/1     Running   0
11m
polaris-consul-consul-server-1               1/1     Running   0
11m
polaris-consul-consul-server-2               1/1     Running   0
11m
polaris-consul-consul-twmpq                  1/1     Running   0
11m
```

```
polaris-mongodb-0                          2/2     Running   0
11m
polaris-mongodb-1                          2/2     Running   0
10m
polaris-mongodb-2                          2/2     Running   0
10m
polaris-ui-84dc87847f-zrg8w                1/1     Running   0
7m12s
polaris-vault-0                            1/1     Running   0
11m
polaris-vault-1                            1/1     Running   0
11m
polaris-vault-2                            1/1     Running   0
11m
public-metrics-657698b66f-67pgt            1/1     Running   0
8m47s
storage-backend-metrics-6848b9fd87-w7x8r   1/1     Running   0
8m39s
storage-provider-5ff5868cd5-r9hj7          1/1     Running   0
8m45s
telegraf-ds-dw4hg                          1/1     Running   0
7m23s
telegraf-ds-k92gn                          1/1     Running   0
7m23s
telegraf-ds-mmxjl                          1/1     Running   0
7m23s
telegraf-ds-nhs8s                          1/1     Running   0
7m23s
telegraf-ds-rj7lw                          1/1     Running   0
7m23s
telegraf-ds-tqrkb                          1/1     Running   0
7m23s
telegraf-rs-9mwgj                          1/1     Running   0
7m23s
telemetry-service-56c49d689b-ffrzx         1/1     Running   0
8m42s
tenancy-767c77fb9d-g9ctv                   1/1     Running   0
8m52s
traefik-5857d87f85-7pmx8                   1/1     Running   0
6m49s
traefik-5857d87f85-cpxgv                   1/1     Running   0
5m34s
traefik-5857d87f85-lvmlb                   1/1     Running   0
4m33s
traefik-5857d87f85-t2xlk                   1/1     Running   0
4m33s
```

```
traefik-5857d87f85-v9wpf                  1/1     Running   0
7m3s
trident-svc-595f84dd78-zb8l6              1/1     Running   0
8m54s
vault-controller-86c94fbf4f-krttq         1/1     Running   0
9m24s
```

4. Verify that the Astra status conditions indicate that the upgrade is complete and ready:

```
kubectl get -o yaml -n [netapp-acc or custom namespace]
astracontrolcenters.astra.netapp.io astra
```

Response:

```
conditions:
  - lastTransitionTime: "2021-10-25T18:49:26Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Ready
  - lastTransitionTime: "2021-10-25T18:49:26Z"
    message: Upgrading succeeded.
    reason: Complete
    status: "False"
    type: Upgrading
```

Upgrade third-party services

The third-party services Traefik and Cert-manager are not upgraded during earlier upgrade steps. You can optionally upgrade them using the procedure described here or retain existing service versions if your system requires it. The following is the recommended Traefik and Certs-manager upgrade sequence:

1. Set up acc-helm-repo to upgrade Traefik and Cert-manager
2. Update Traefik service using acc-helm-repo
3. Update the Cert-manager service

Set up acc-helm-repo to upgrade Traefik and Cert-manager

1. Find the `enterprise-helm-repo` that is loaded to your local Docker cache:

```
docker images enterprise-helm-repo
```

Response:

```
REPOSITORY              TAG          IMAGE ID       CREATED         SIZE
enterprise-helm-repo    21.10.218    7a182d6b30f3   20 hours ago    464MB
```

2. Start a container using the tag from the previous step:

```
docker run -dp 8082:8080 enterprise-helm-repo:21.10.218
```

Response:

```
940436e67fa86d2c4559ac4987b96bb35588313c2c9ddc9cec195651963f08d8
```

3. Add the Helm repo to your local host repositories:

```
helm repo add acc-helm-repo http://localhost:8082/
```

Response:

```
"acc-helm-repo" has been added to your repositories
```

4. Save the following Python script as a file, for example, `set_previous_values.py`:

This Python script creates two files that are used in later upgrade steps to retain helm values.

```
#!/usr/bin/env python3
import json
import os

NAMESPACE = "netapp-acc"

os.system(f"helm get values traefik -n {NAMESPACE} -o json >
traefik_values.json")
os.system(f"helm get values cert-manager -n {NAMESPACE} -o json >
cert_manager_values.json")

# reformat traefik values
f = open("traefik_values.json", "r")
traefik_values = {'traefik': json.load(f)}
f.close()

with open('traefik_values.json', 'w') as output_file:
    json.dump(traefik_values, output_file)

# reformat cert-manager values
f = open("cert_manager_values.json", "r")
cm_values = {'cert-manager': json.load(f)}
f.close()

cm_values['global'] = cm_values['cert-manager']['global']
del cm_values['cert-manager']['global']

with open('cert_manager_values.json', 'w') as output_file:
    json.dump(cm_values, output_file)

print('Done')
```

5. Run the script:

```
python3.7 ./set_previous_values.py
```

Update Traefik service using acc-helm-repo

> You must already have set up acc-helm-repo before completing the following procedure.

1. Download the Traefik bundle using a secure, file-transfer tool, such as GNU wget:

```
wget http://localhost:8082/traefik-0.2.0.tgz
```

2. Extract the images:

```
tar -vxzf traefik-0.2.0.tgz
```

3. Apply the Traefik CRDs:

```
kubectl apply -f ./traefik/charts/traefik/crds/
```

4. Find the Helm chart version to use with your upgraded Traefik:

```
helm search repo acc-helm-repo/traefik
```

Response:

```
NAME                                    CHART VERSION   APP VERSION
DESCRIPTION
acc-helm-repo/traefik                   0.2.0           2.5.3       Helm
chart for Traefik Ingress controller
acc-helm-repo/traefik-ingressroutes   0.2.0           2.5.3       A Helm
chart for Kubernetes
```

5. Validate the traefik_values.json file for upgrade:
   a. Open the traefik_values.json file.
   b. Check if there is a value for the `imagePullSecret` field. If it is empty, remove the following text from the file:

```
"imagePullSecrets": [{"name": ""}],
```

   c. Ensure that the traefik image is directed to the correct location and has the correct name:

```
image: [your_registry_path]/traefik
```

6. Upgrade your Traefik configuration:

```
helm upgrade --version 0.2.0 --namespace netapp-acc -f
traefik_values.json traefik acc-helm-repo/traefik
```

Response:

```
Release "traefik" has been upgraded. Happy Helming!
NAME: traefik
LAST DEPLOYED: Mon Oct 25 22:53:19 2021
NAMESPACE: netapp-acc
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

Update the Cert-manager service

You must already have completed the Traefik update and added acc-helm-repo in Helm before completing the following procedure.

1. Find the helm chart version to use with your upgraded Cert-manager:

```
helm search repo acc-helm-repo/cert-manager
```

Response:

```
NAME CHART VERSION APP VERSION DESCRIPTION
acc-helm-repo/cert-manager 0.3.0 v1.5.4 A Helm chart for cert-manager
acc-helm-repo/cert-manager-certificates 0.1.0 1.16.0 A Helm chart for
Kubernetes
```

2. Validate the cert_manager_values.json file for upgrade:
   a. Open the cert_manager_values.json file.
   b. Check if there is a value for the `imagePullSecret` field. If it is empty, remove the following text from the file:

```
"imagePullSecrets": [{"name": ""}],
```

   c. Ensure that the three cert-manager images are directed to the correct location and have the correct names.
3. Upgrade your Cert-manager configuration:

```
helm upgrade --version 0.3.0 --namespace netapp-acc -f
cert_manager_values.json cert-manager acc-helm-repo/cert-manager
```

Response:

```
Release "cert-manager" has been upgraded. Happy Helming!
NAME: cert-manager
LAST DEPLOYED: Tue Nov 23 11:20:05 2021
NAMESPACE: netapp-acc
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

Verify system status

1. Log in to Astra Control Center.
2. Verify that all your managed clusters and apps are still present and protected.

Uninstall Astra Control Center

You might need to remove Astra Control Center components if you are upgrading from a trial to a full version of the product. To remove Astra Control Center and the Astra Control Center Operator, run the commands described in this procedure in sequence.

What you'll need

- Use Astra Control Center UI to unmanage all clusters.

Steps

1. Delete Astra Control Center. The following sample command is based upon a default installation. Modify the command if you made custom configurations.

   ```
   kubectl delete -f astra_control_center_min.yaml -n netapp-acc
   ```

   Result:

   ```
   astracontrolcenter.astra.netapp.io "astra" deleted
   ```

2. Use the following command to delete the `netapp-acc` namespace:

   ```
   kubectl delete ns netapp-acc
   ```

   Result:

   ```
   namespace "netapp-acc" deleted
   ```

3. Use the following command to delete Astra Control Center operator system components:

```
kubectl delete -f astra_control_center_operator_deploy.yaml
```

Result:

```
namespace "netapp-acc-operator" deleted
customresourcedefinition.apiextensions.k8s.io
"astracontrolcenters.astra.netapp.io" deleted
role.rbac.authorization.k8s.io "acc-operator-leader-election-role"
deleted
clusterrole.rbac.authorization.k8s.io "acc-operator-manager-role"
deleted
clusterrole.rbac.authorization.k8s.io "acc-operator-metrics-reader"
deleted
clusterrole.rbac.authorization.k8s.io "acc-operator-proxy-role" deleted
rolebinding.rbac.authorization.k8s.io "acc-operator-leader-election-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "acc-operator-manager-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "acc-operator-proxy-
rolebinding" deleted
configmap "acc-operator-manager-config" deleted
service "acc-operator-controller-manager-metrics-service" deleted
deployment.apps "acc-operator-controller-manager" deleted
```

Find more information

- Known issues for uninstall