
Anaconda Platform Documentation

Release 5.1.2-0

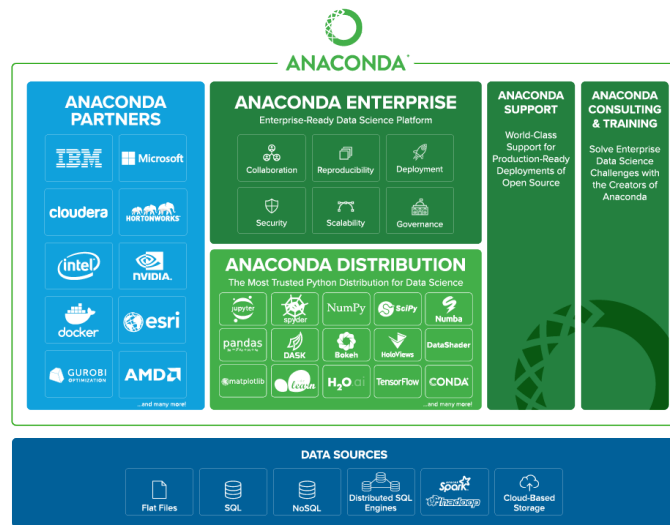
Anaconda Inc.

Mar 29, 2018

Contents

1	User guide	3
2	Administration guide	91
3	Frequently asked questions	187
4	Release notes	195
5	Known issues	201
6	Glossary	205
7	Help and support	207

Anaconda Enterprise is an enterprise-ready, secure and scalable data science platform that empowers teams to govern data science assets, collaborate and deploy data science projects.



Enterprise includes these capabilities:

- Easily deploy your projects into interactive data applications, live notebooks and machine learning models with APIs.
- Share those applications with colleagues and collaborators.
- Manage your data science assets: notebooks, packages, environments and projects in an integrated data science experience.

To try out all of Enterprise's major features, see the [Getting Started](#) guide.

Anaconda Enterprise brings together different types of users: data scientists, business analysts, viewers and IT administrators.

To quickly become familiar with using Enterprise, create your first project with the 20-minute *Getting started* guide.

After that, try the deep dive *Tutorials*.

You can also click through all the pages in this user guide by clicking the “Next” or “Previous” buttons at the bottom of each page.

1.1 Getting started

This 20-minute getting started guide walks you through all of the things you can do with Anaconda Enterprise. We recommend that you do each step hands-on in Enterprise as you walk through this guide.

You can choose to deploy and share an existing *Anaconda Project*, or jump right into creating and editing a new project.

TIP: Download the *Cheat Sheet* for handy reference while you work through the steps.

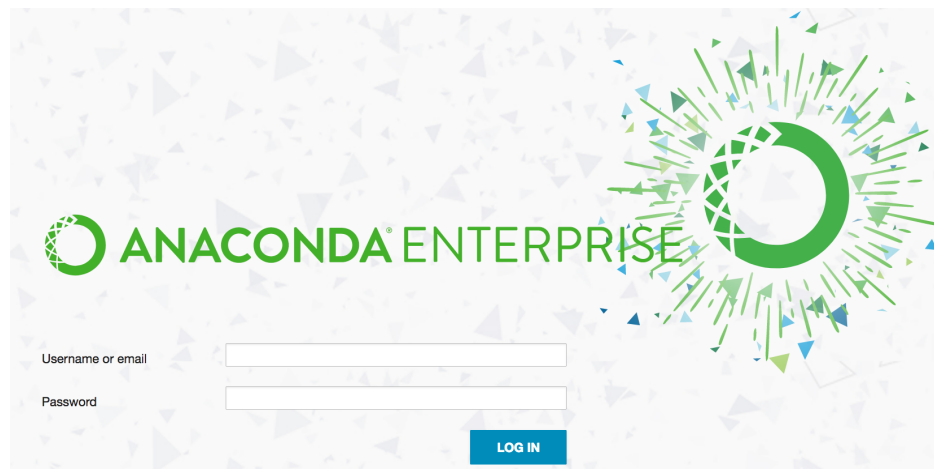
Getting started steps

- *Before you start*
- *Deploy a sample project*
- *Create an Anaconda Project*
- *Launch an Editing Session*
- *Add a notebook*
- *See how Anaconda Projects work*
 - *Add a download*
 - *Add a channel*

- *Add packages*
 - *Prepare all environments*
 - *Link an environment to the notebook*
- *Run the Notebook*
- *Deploy the Project*
 - *Add a deploy command*
 - *Commit changes and exit*
 - *Deploy it*
- *Share your Deployment*
- *What's next?*

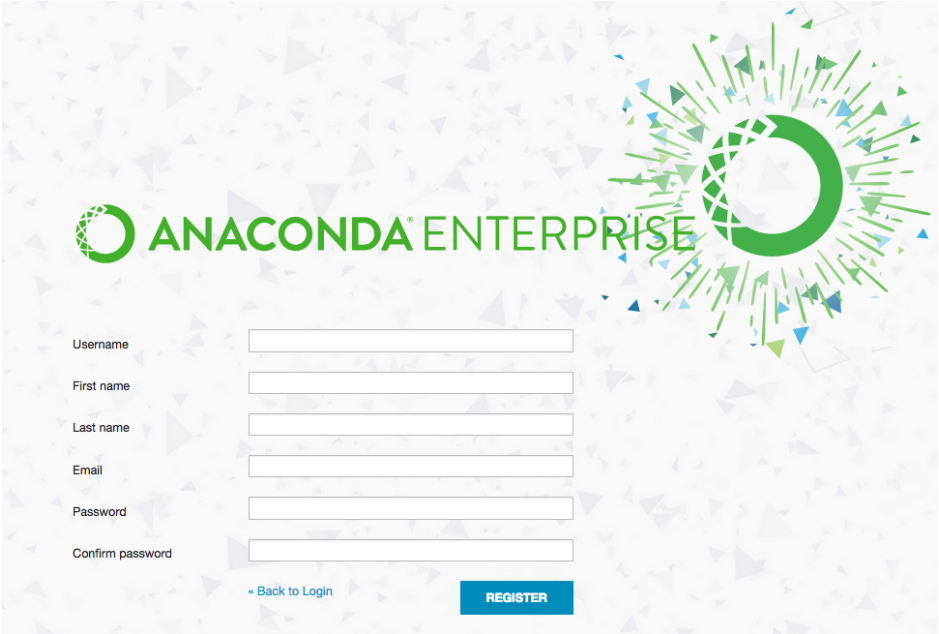
1.1.1 Before you start


You will have already received a welcome email from your system administrator with a link to your Enterprise site, your username and password. Click the link and enter your username or email address and password, then click the Login button:






NOTE: If your administrator enabled self-registration, you can create your own username and password. Enter the following information to register:

Now you are at the Anaconda Enterprise dashboard, which displays a list of all your Anaconda Projects.



 ANACONDA ENTERPRISE

ProjectsDeploymentsPackages



Projects

Sample Projects



Project	Status	Owner	Modified
markowitz_notebook	No active session	bob	6 minutes ago
data_crossfilter	No active session	bob	10 minutes ago
weather_statistics	No active session	bob	27 minutes ago

1.1.2 Deploy a sample project

Enterprise includes many types of sample projects for you to use for testing and to use as recipes.

To open a sample project, click the top right “Sample Projects” link. Scroll to find the project named “data_crossfilter” and click the “Save to My Projects” link.

ANACONDA ENTERPRISE

Projects

Deployments

Packages

?

Projects

Sample Projects

Filter by Name and Owner

Project	Status	Owner	Modified
markowitz_notebook	No active session	bob	6 minutes ago
data_crossfilter	No active session	bob	10 minutes ago
weather_statistics	No active session	bob	27 minutes ago

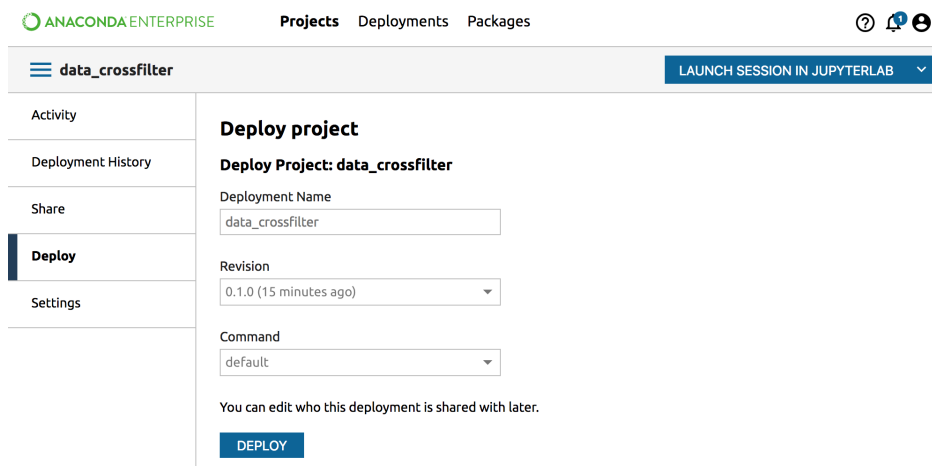
Then return to the Projects list by clicking the top left “Back to My Projects” link.

Select the Data Crossfilter project by clicking its name.

1.1.3 Deploy your Project

To *Deploy* the Data Crossfilter project, from the left navigation click the Deploy tab.

Accept the Project’s default settings by clicking the Deploy button.



ANACONDA ENTERPRISE
Projects
Deployments
Packages

data_crossfilter
LAUNCH SESSION IN JUPYTERLAB

Activity
Deployment History
Share
Deploy
Settings

Deploy project

Deploy Project: data_crossfilter

Deployment Name

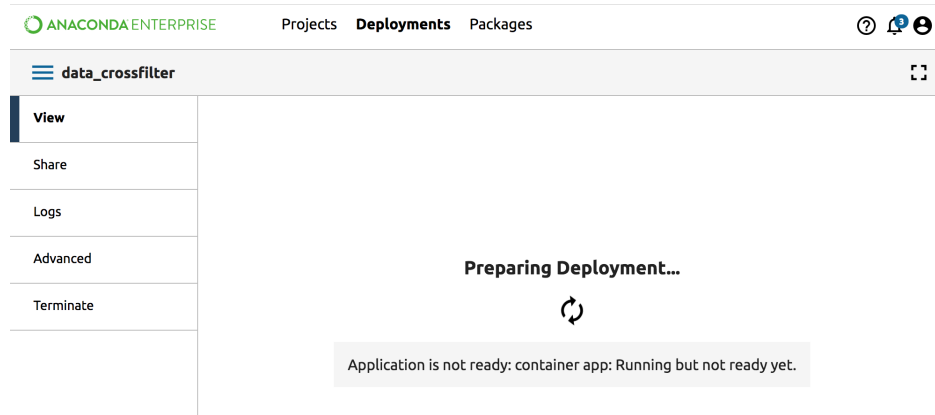
Revision

Command

You can edit who this deployment is shared with later.

DEPLOY

After Enterprise finishes deploying, click the Done button to view the deployment.



1.1.4 Interact with your Deployment

Immediately you can interact with your new Deployment.

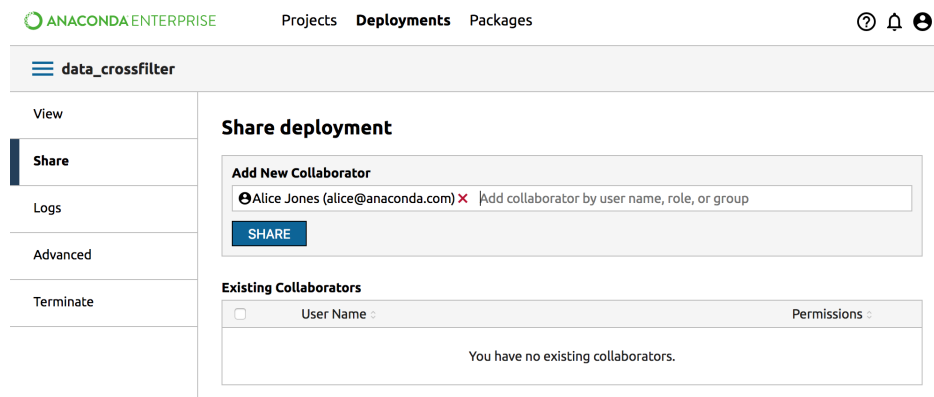
You can interact with any widgets included in the Project. Change the color and size of the dots, or change the axes that are displayed.



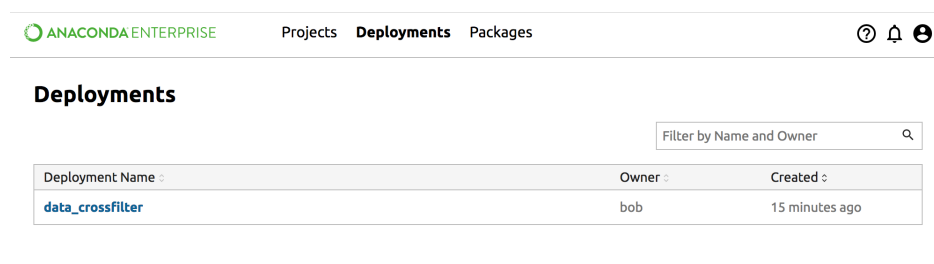
1.1.5 Share your Deployment with others

You can easily share your Deployment with other Enterprise users.

From the left menu click Share, then in the Add New Collaborator box begin typing the name of another Enterprise user. Select the name of the user with whom you want to share, then click the Share button.



When your collaborator logs onto Enterprise, they will see the new Deployment in their Deployments list.



Clicking on the deployment allows your collaborator to interact with widgets. They cannot share, view logs, or terminate the deployment.

NOTE: If you change the state of the widgets before sharing, the changed state is not shared between collaborators.

1.1.6 Create an Anaconda Project

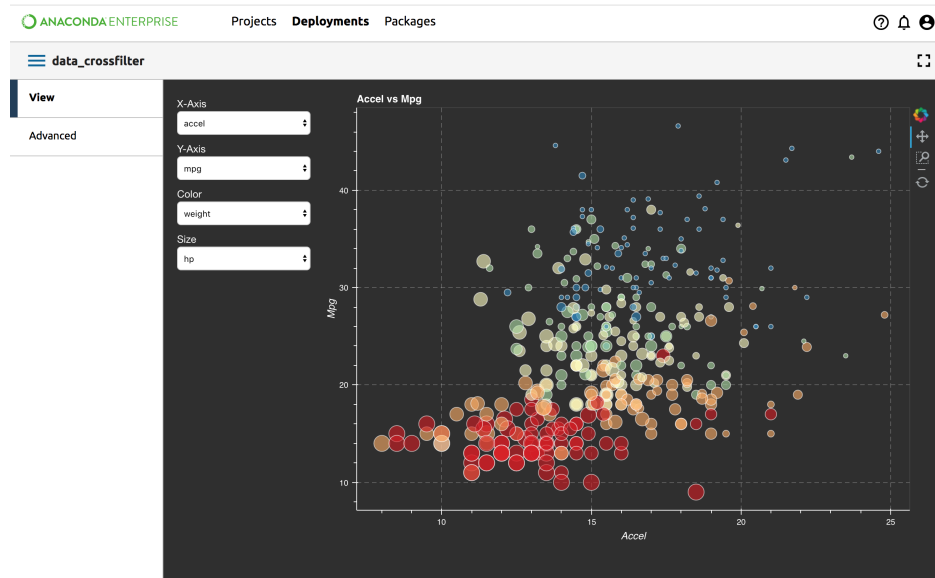
In this exercise you will create a new Anaconda Project, open it in an editing session, and upload a notebook to your project.

Then you'll take a peek at the file that governs your Project, the `anaconda-project.yml`.

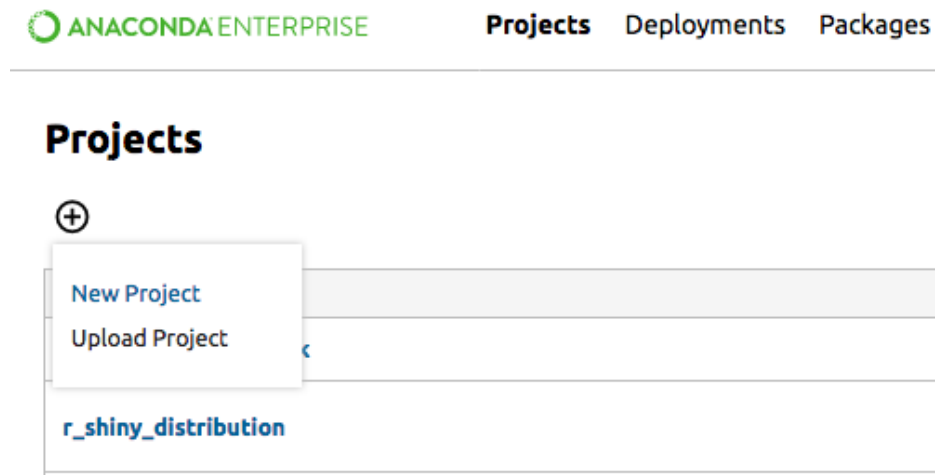
After that you'll use the Project interface to easily add a download, a channel and a package, prepare an environment and link the notebook to it.

Next you'll add a deployable command, commit the changes, and run your project.

Finally you will deploy the project and share it with other Enterprise users.



To get started, on the **Projects** list, click the top left Add (+) button and select New Project.



Name your new project “finance_notebook”. Click the Next button.

You can choose among several types of projects, including Anaconda 3, Anaconda 2, R Language, SAS and Spark. Select Anaconda 3.6 and click the Create button.

Create New Project

1 Name Project

2 Select Project Type

3 View or Edit

Leaving this field blank will name the project "Untitled" by default.

finance_notebook ✓

NEXT

Create New Project

1 Name Project

2 Select Project Type

3 View or Edit

Select which project template you want to use.

☒ Anaconda 3.6

☐ Anaconda 2

☐ Anaconda 3.5

☐ Hadoop-Spark

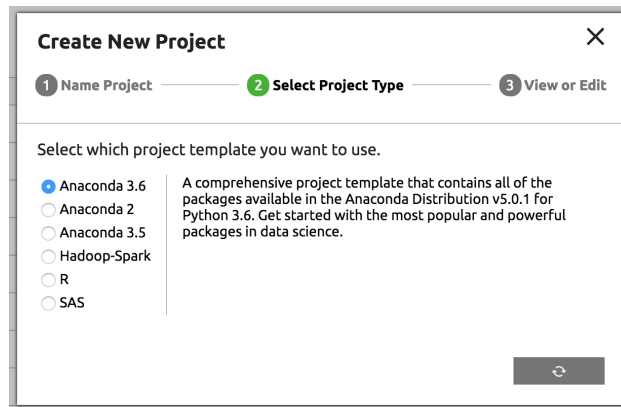
☐ R

☐ SAS

A comprehensive project template that contains all of the packages available in the Anaconda Distribution v5.0.1 for Python 3.6. Get started with the most popular and powerful packages in data science.

BACK CREATE

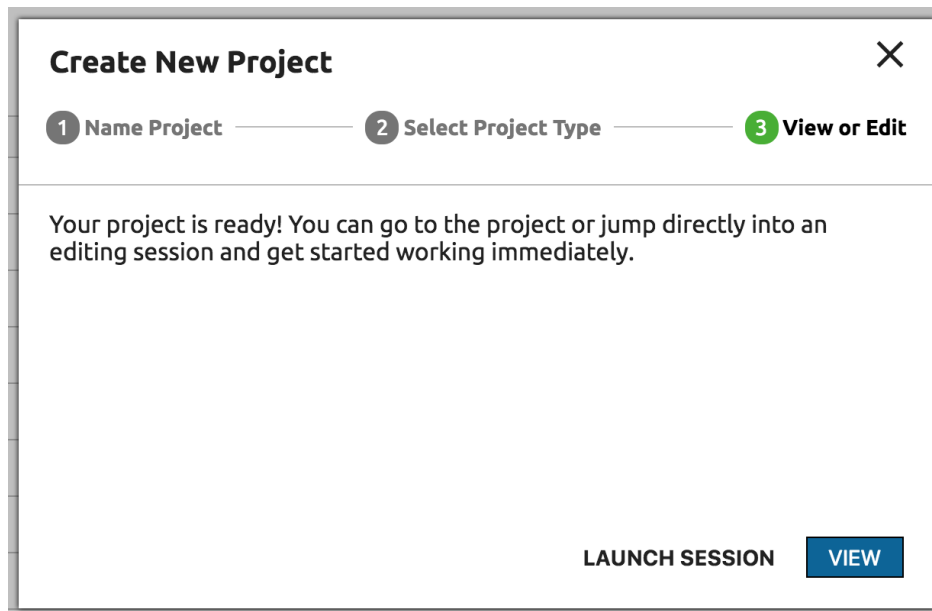
Anaconda Enterprise takes a minute to prepare your Project and copy in all the necessary files.



TIP: While it's running, download and check out the handy [JupyterLab](#) and [Jupyter Notebook](#) cheat sheet.

1.1.7 Launch an Editing Session

Click the LAUNCH SESSION link.



SEE ALSO: [Launch Editing Session](#).

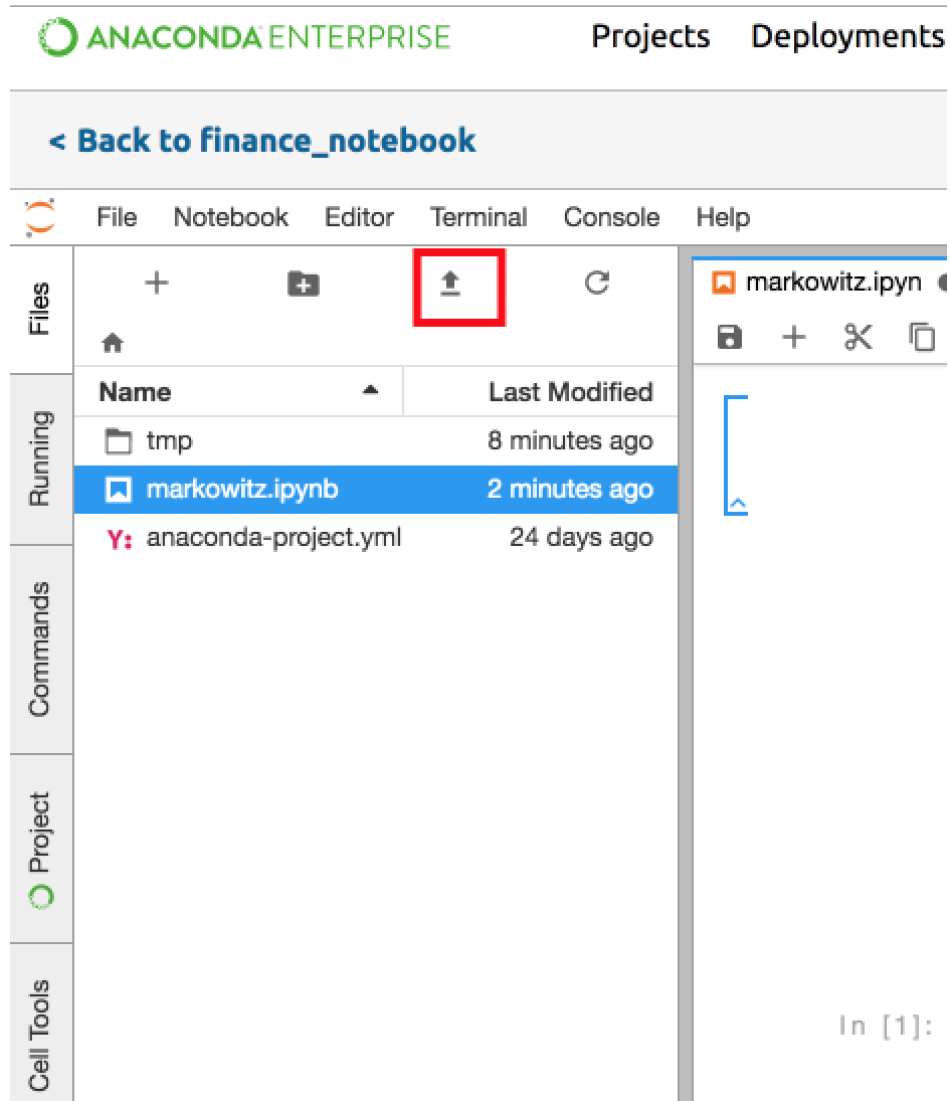
1.1.8 Add a notebook

You can use *JupyterLab* to create new notebooks or edit existing notebooks.

To see the data science capabilities of the Anaconda Enterprise, you can play with a large notebook that is already written for you. The Intro to Data Analysis Notebook is based on a classic titanic data analysis problem. Let's say a friend or colleague gave you this notebook, and you want to add it to this project.

Click this link to download the `Markowitz` notebook to your computer.

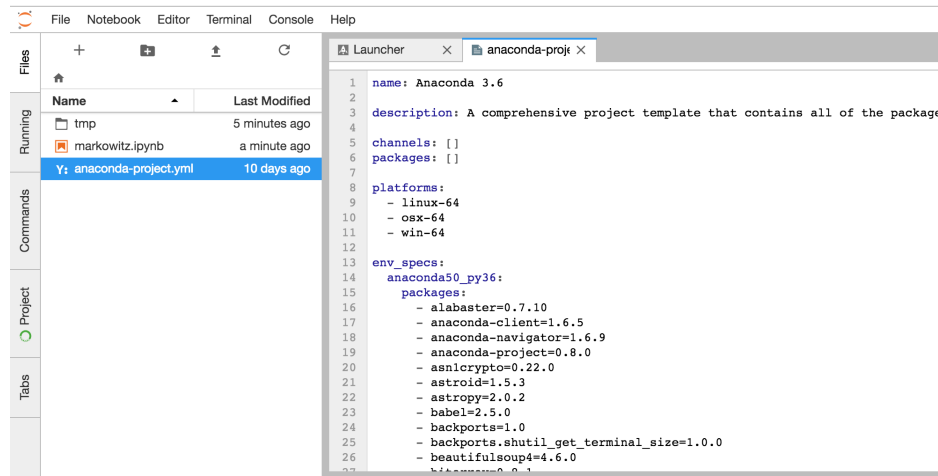
Next, upload the Markowitz notebook to your Project. Click the left **Files** tab, then click the Upload File(s) icon at the top:



To see the uploaded file, click the left FILES tab. The uploaded Notebook appears in the file list.

1.1.9 See how Anaconda Projects work

Open the `anaconda-project.yml` configuration file now to take a look at it:



This file shows everything that was automatically created when you launched your new project.

If you wanted to make changes to your Project file, you would click the left **Project** tab.

Add a download

Now let's say you want a file to be automatically downloaded when you run the notebook, so now you will add a download to this project.

On the left **Project** tab, find the "DOWNLOADS" field, click its Add (+) icon and add a file for the notebook to download.

Copy and paste this into the URL box: <http://pastebin.com/raw/BX9mBNhL>

In the Name box enter "MARKOWITZ" and in the local file box enter "markowitz.csv". Click the Save button.

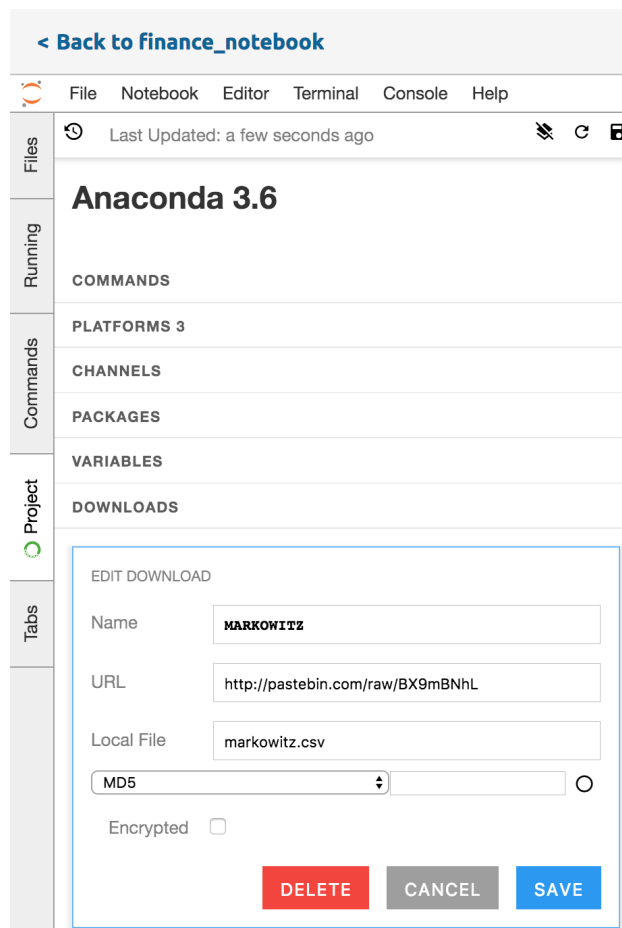
Save all your changes by clicking the blue-highlighted "Project Save" button at the top right of this pane.

NOTE: In an air gapped environment, click this link to download the Markowitz data to your computer and then upload it in the same way you did the notebook.

Add a channel

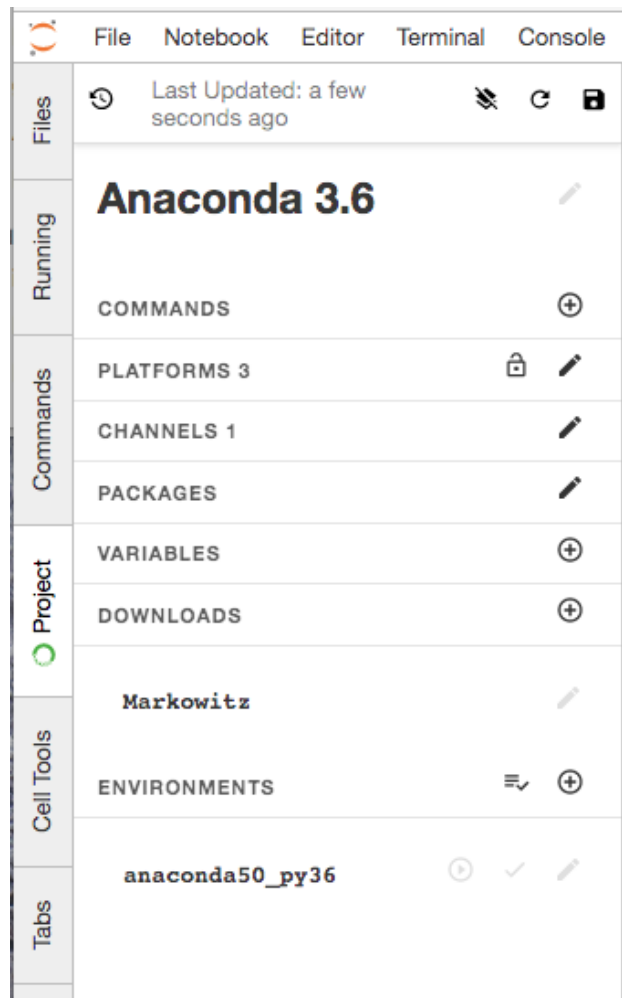
Add the R Language [Channel](#) to this Project so you can easily search, download and install packages from this specific channel when using this project.

In the left **Project** tab, open the CHANNELS field with the pencil icon next to it.



Then copy and paste this link: <https://conda.anaconda.org/r>

Click the Add (+) icon to add the channel, then click the blue-highlighted “Project Save” button at the top right of this pane.



Add packages

Add a *Package* to this project. In the left **Project** tab, find the “PACKAGES” field and click its edit icon, the pencil, to add packages. Adding packages here adds them to your `anaconda-project.yml` configuration file. You may edit that file directly if you prefer.

`markowitz.ipynb` uses the following packages which are already installed with Anaconda 3.6. Optionally, the notebook can depend on R. If you set `use_R = True` in the first cell, then you should add the following packages:

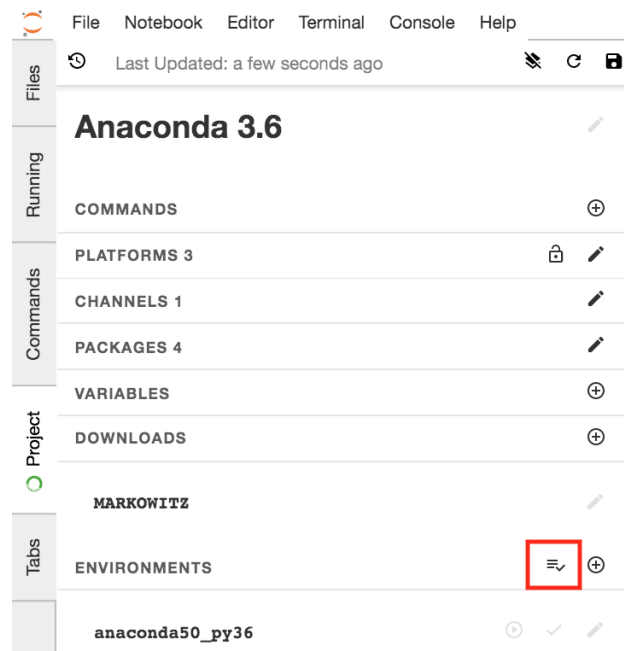
- `r`
- `rpy2`

After adding packages, click the “Save Project Data” icon in your left panel on the top right corner. This saves your project locally.

Prepare all environments

Now that you’ve added all these goodies to your Project, it’s time to link the notebook to your Project environment.

Still on the left **Project** tab, find the ENVIRONMENTS field and click the “Prepare All Environments” button next to the Add (+) icon.

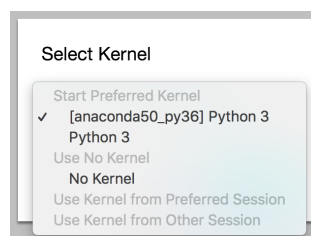


1.1.10 Run the Notebook

Click the left **Files** tab, then double-click “markowitz.ipynb” to open the notebook.

Link an environment to the notebook

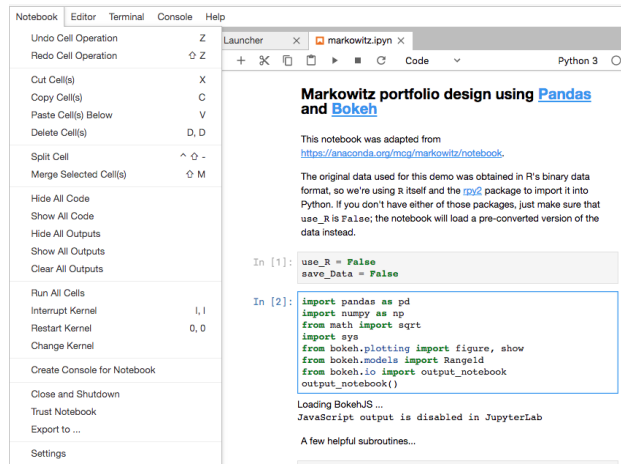
When you open the notebook you will be prompted to select a kernel. From the drop-down list, select “[anaconda50_py36]Python 3”.



TIP: You can always change a notebook kernel from the top right corner of the notebook tab. Click the kernel name - for example “Python 3”, and select “[anaconda50_py36]Python 3” from the drop-down list.

Run Cells

Click the top “Notebook” menu, and from the drop-down select “Run All Cells”.

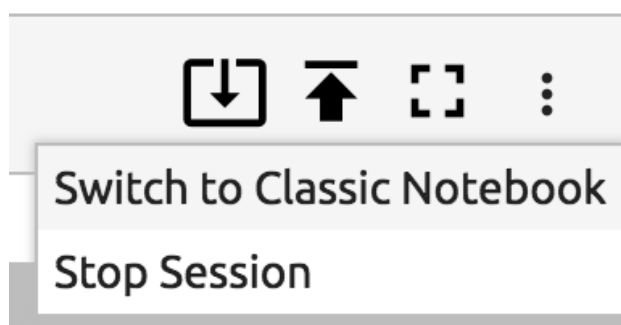


NOTE: You will get an import error if your notebook is not linked to the project environment.

Switch to Classic Notebook

Since JupyterLab does not support inline Bokeh yet, we need to switch to Classic Notebook to view our plots.

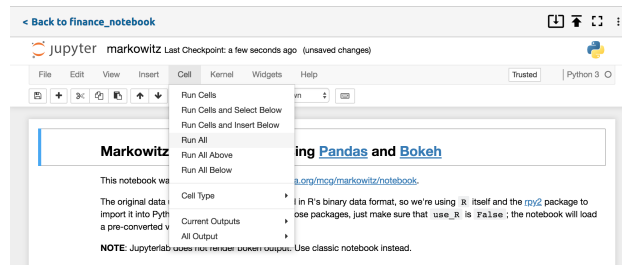
Click the 3 dots at the top right and select “Switch to Classic Notebook”



Run Cells in Notebook

Click “markowitz_notebook” to open the notebook.

Click the top “Cell” menu, and from the drop-down select “Run All”.



Scroll down to see and interact with the plots that you just created.

Switch back to JupyterLab

Click the 3 dots at the top right and select “Switch to JupyterLab”.

1.1.11 Deploy the Project

To deploy your project, you will add a deployable command and commit the changes, then leave the editing session and deploy using the Anaconda Enterprise interface.

Add a deploy command

Any project you want to deploy needs to have a deployable command for it to run.

From the left **Project** tab (not Commands tab), find the “COMMANDS” field and click its Add (+) icon.

In the Type section click “Notebook”

In the Name box enter “default”

In the notebook box enter “markowitz.ipynb”

Click the Save button.

Commit changes and exit

In the top navigation bar on the far right, click the Commit icon.

Check the “Tag as Deployable Revision” checkbox and click Commit Changes.

[< Back to finance_notebook](#)

File Notebook Editor Terminal Console Help

Last Updated: 12 minutes ago

Anaconda 3.6

COMMANDS

EDIT COMMAND

Name

Description

Type *bokeh_app* **notebook** *shell*

Deployable Web App ☒

DELETE CANCEL SAVE



Commit Changes

File	Last Saved
anaconda-project.yml	4 minutes ago
markowitz.csv	13 minutes ago
markowitz.ipynb	11 minutes ago

Comments

☒ Tag as Deployment Revision

CANCEL COMMIT CHANGES

Then exit the Notebook with the “Back to finance_notebook” link on the top left corner. This brings you back to the Project Detail page.

Deploy it

Click the left Deploy tab, then click the Deploy button to deploy the Project.

While it is deploying, you can click the Deployment History tab to see a counter. A notification will tell you it has deployed.

Interact with Deployment

Time to play with your Deployment! In the **Deployments** list, click the name of your new Deployment to view and interact with it.

From the left View tab you can choose Cell, All Output, and Clear to clear the output. Then run all cells in your notebook with Cell and Run All.

1.1.12 Share your Deployment

In the **Deployments** list, click the name of the Deployment to share, then click the **Share** tab. In the box that appears, begin typing the username of your collaborator with whom you wish to share, select their name from the drop-down list and click the Share button.

Your collaborator appears in the Existing Collaborators list below.

ANACONDA ENTERPRISE Projects **Deployments** Packages ? ? ?

finance_notebook

View

Share

Logs

Advanced

Terminate

Share deployment

Add New Collaborator

@@developers x Add collaborator by user name, role, or group

SHARE

Existing Collaborators

<input type="checkbox"/>	User Name	Permissions
<input type="checkbox"/>	Alice Jones (alice@anaconda.com)	Editor

When your collaborator logs onto Enterprise, they will see the new Deployment in their Deployments list.

1.1.13 What's next?

Now that you've spent a little time with Anaconda Enterprise, try out the * *tutorials*.

You can also dig deeper with the following:

- *Concepts*
- *Glossary*
- *Anaconda Project*
- *Managing Packages*
- *Deployments*.

1.2 Anaconda Enterprise Cheat Sheet

Download the [Anaconda Enterprise V5.1 Cheat Sheet \(232 KB PDF\)](#) for an easy guide to using Anaconda Enterprise. It's sized for easy printing so you can keep it next to your computer.

1.3 Concepts

SEE ALSO: *Glossary*

1.3.1 Anaconda Project

An Anaconda Project is a portable encapsulation of your data science assets that includes all the necessary configuration to automate its setup and deployment: packages, file downloads, environment variables and runnable commands services, packages, channels and environment specifications. It includes a folder that contains a configuration file named `anaconda-project.yml` together with scripts, notebooks and other related files.

Data scientists use Projects to encapsulate their data science work and make it easily portable. A project is usually compressed into a `.tar.bz2`, `.tar.gz` or `.zip` file for sharing and storage.

Anaconda Enterprise uses the [Anaconda Project](#) specification format.

SEE ALSO: *Projects*.

1.3.2 Channels

The locations of the repositories where Anaconda Enterprise looks for packages using the conda package manager. Channels may point to a Cloud repository or a private location on a remote or local repository that you or your organization created. The conda channel command has a default set of channels to search beginning with <https://repo.continuum.io/pkgs/>. You may override the default channels, for example to maintain a private or internal channel. In conda commands and in the `.condarc` file, these default channels are referred to by the channel name “defaults”.

SEE ALSO: *Packages and Channels*.

1.3.3 Deployments

A deployed Anaconda Project is called a Deployment. You can deploy a Project as an interactive visualization, a live notebook or a machine learning model with an API application.

The software deployment process includes all of the activities that make a software application available for use. When you deploy a project, Anaconda Enterprise finds and builds all of the software dependencies—the programs on which

the deployment depends in order to run—and encapsulates them so they are completely self-contained. This allows you to easily share the deployment with others. Everything they need to deploy and run the project is included.

You configure how a project is deployed by adding a run command in the configuration file `anaconda-project.yml` and selecting the appropriate deployment command.

SEE ALSO: [Deployments](#).

1.3.4 Interactive data applications

Interactive data applications are visualizations with sliders, drop-downs and other widgets that allow users to interact with them. Widgets can drive new computations, update plots and connect to other programmatic functionality.

With Enterprise, you can create and deploy interactive data applications built using popular libraries such as [Bokeh](#) and [Shiny](#).

SEE ALSO: The sample projects `Shiny example r_shiny_distribution`, the `Bokeh example weather_statistics` and others.

1.3.5 Live notebooks

Data scientists use live notebooks to present their work and share it with others. With Enterprise, you can immediately deploy your notebooks with just a few clicks, so that other users can run them and get updated results on demand.

JupyterLab and the classic Jupyter Notebooks are web applications that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses for Notebooks include:

- Data cleaning and transformation.
- Numerical simulation.
- Statistical modeling.
- Machine learning.

JupyterLab is the next-generation notebook, and the classic Jupyter Notebooks is the prior generation. Access Notebooks by opening a project, then open JupyterLab and select “Switch to Classic Notebook”.

For more information, see [Jupyter](#).

Enterprise supports both R and Python notebooks. For more information, see the Python notebook example `stocks_live_notebook` and the R notebook example `stocks_live_notebook` in the `anaconda-enterprise` channel.

1.3.6 Packages

Software files and information about the software, such as its name, the specific version and a package description, that are bundled into a file that can be installed and managed by a package manager.

SEE ALSO: [Packages](#).

1.3.7 REST APIs

A common way to operationalize machine learning models is through REST APIs. REST APIs are callable URLs which provide results based on a query. This allows developers to make their applications intelligent without having to write models themselves.

RESTful endpoints are a great way to bridge the gap between the data scientists writing machine learning models and the developers writing end-to-end applications. With Enterprise REST API deployment, applications can request predictions as a service.

For more information on REST APIs, look in the sample projects for the `quote_api` project, and see the [tutorial on creating a REST API](#).

1.3.8 User scopes

Anaconda Enterprise brings together many types of users. Some users simply need to view output, so users in this role are called **viewers**.

Those who want to run applications that are already deployed are **business analysts**.

Users who write and deploy applications are called **data scientists**.

Those who install, configure and administer Anaconda Enterprise, usually IT administrators, are called **administrators**.

These roles may overlap, for example, if a user is assigned to an administrative role they may also be referred to as a **superuser**.

1.3.9 Version control

Anaconda Enterprise uses version control to track changes in your project files and to coordinate the work on those files among multiple users. Enterprise lets you know when files have been changed, so you can choose which version to use.

You can edit and save files locally, and then upload or **commit** them to the server. Before uploading, Enterprise checks your version against the version on the server. If there are conflicts made by your collaborators, you will see a warning. You can review the changes and either cancel your commit, or choose to override the warning and overwrite your collaborators' version.

1.3.10 Web applications

A software program that can be run in a browser. Python is the leading open source data science language, and it is also widely used for web development. Developers and data scientists can write intelligent web applications together using Python, improving collaboration. Popular Python web frameworks include [Flask](#) and [Django](#).

For more information, look in the Example Projects for the Flask example `image_classifier_flask`.

1.4 Basic tasks

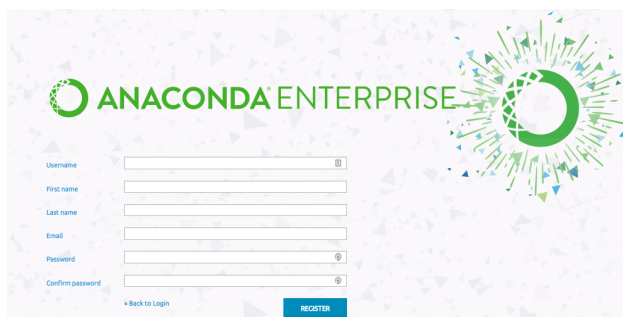
All users, groups and roles can perform the following basic account maintenance tasks:

Here you can *create and manage your account*, *change email address, name or password*, *enable two-step authentication*, view your *session history*, view a list of *running applications and their related info*, view your *account history*, and *log in* or *log out*.

1.4.1 Creating an account

1. Receive an email from your system administrator with instructions on how to register and log in.

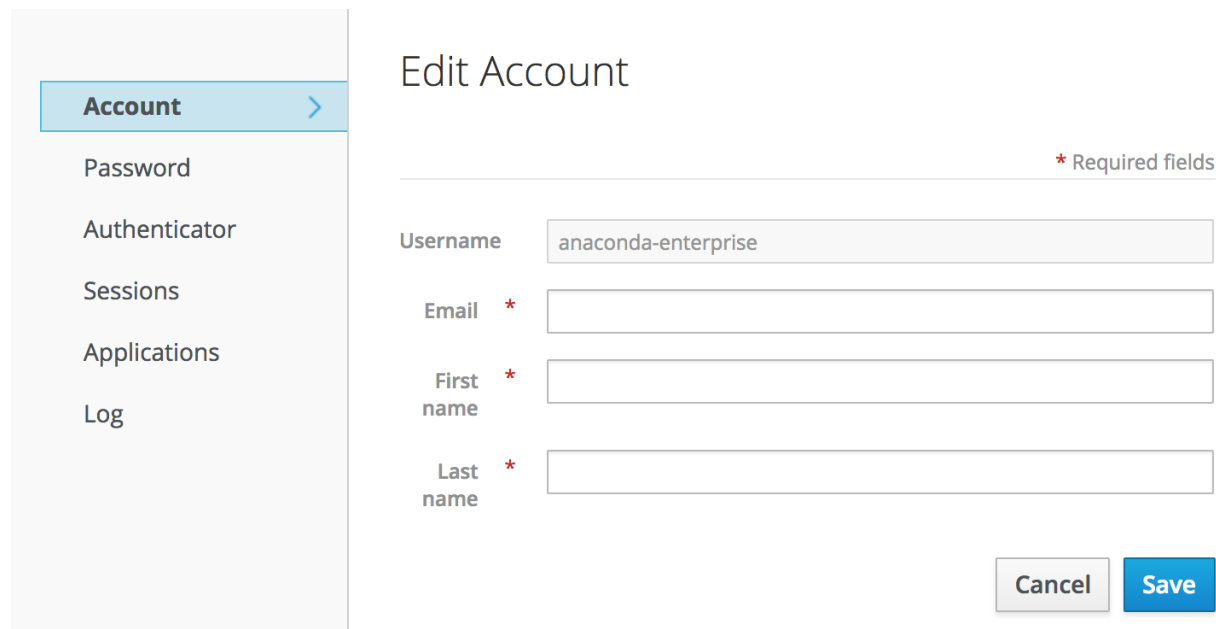
2. Click the link in your registration email. Or in your browser, type the address to go to Anaconda Enterprise.
3. On the Enterprise registration page, click the Register link, then create your Enterprise account:



The image shows the Anaconda Enterprise registration page. It features the Anaconda logo and the text "ANACONDA ENTERPRISE". Below this, there are input fields for Username, First name, Last name, Email, Password, and Confirm password. A "Back to Login" link is located below the Confirm password field, and a "REGISTER" button is at the bottom right.

1.4.2 Changing your account settings

1. In the top-right corner of the Anaconda Enterprise screen, click the user icon, then select Account Settings from the drop-down menu that appears.



The image shows the "Edit Account" page. On the left is a sidebar menu with options: Account (selected), Password, Authenticator, Sessions, Applications, and Log. The main content area is titled "Edit Account" and includes a legend for "* Required fields". Below this, there are input fields for Username (pre-filled with "anaconda-enterprise"), Email, First name, and Last name. At the bottom right are "Cancel" and "Save" buttons.

- Account—Change your email address, first name or last name.
- Password—Change your password.
- Authenticator—Enable 2-step verification with FreeOTP or Google Authenticator.
- Sessions—View a list of when sessions started, last access, IP address, and log out of all sessions.

- Applications–View a list of running applications, their permissions and further info.
- Log–View the history of all account actions with date, event, IP address used, client and details.

Account

Edit your email address, first or last name, then click the Save button.

Password

Enter your current password, then a new password and the new password again in the Confirmation box. Click the Save button.

Change Password

All fields required

Account

Password >

Authenticator

Sessions

Applications

Log

Password

New Password

Confirmation

Save

Authenticator

You will need to have FreeOTP or Google Authenticator previously installed. You can get both in Google Play and in the Apple App Store.

After downloading, open the application and scan the barcode or enter the key.

Then paste the one-time code into the box and click the Save button.

Sessions

View a list of when sessions started, when you last accessed your account, view your IP address or log out of all sessions.

[Account](#)[Password](#)**Authenticator** >[Sessions](#)[Applications](#)[Log](#)

Authenticator

1. Install [FreeOTP](#) or Google Authenticator on your device. Both applications are available in [Google Play](#) and Apple App Store.
2. Open the application and scan the barcode or enter the key.



NBGU 42SM KB2H MRDT M5QW EOCD NE2E KTDI

3. Enter the one-time code provided by the application and click Save to finish the setup.

One-
time
code

Account
Password
Authenticator
Sessions >
Applications
Log

Sessions

IP	Started	Last Access	Expires	Clients
10.244.45.1	Mar 12, 2018 5:45:06 PM	Mar 12, 2018 5:47:12 PM	Mar 13, 2018 3:45:06 AM	anaconda- platform account
10.244.45.1	Mar 12, 2018 5:39:42 PM	Mar 12, 2018 5:39:42 PM	Mar 13, 2018 3:39:42 AM	anaconda- platform
10.244.45.1	Mar 12, 2018 5:44:09 PM	Mar 12, 2018 5:44:46 PM	Mar 13, 2018 3:44:09 AM	anaconda- platform
10.244.45.1	Mar 12, 2018 5:48:21 PM	Mar 12, 2018 5:49:20 PM	Mar 13, 2018 3:48:21 AM	anaconda- platform account

[Log out all sessions](#)

Applications

View a list of all of your running applications, their available and granted permissions, the personal information granted to them and any additional grants.

If a grant is revocable, you will see a “Revoke Grant” button in the Action column. Click it to revoke access.

Account
Password
Authenticator
Sessions
Applications >
Log

Applications

Application	Available Permissions	Granted Permissions	Granted Personal Info	Additional Grants	Action
app_client_76760ff585934c3b978b5602da92f16e	Offline access	Full Access	Full Access		
Account	Manage account in Account , View profile in Account	Full Access	Full Access		
my client	Offline access	Full Access	Full Access		

Log

View the history of all account actions with date, event, IP address used, the client and details.

Account

Password

Authenticator

Sessions

Applications

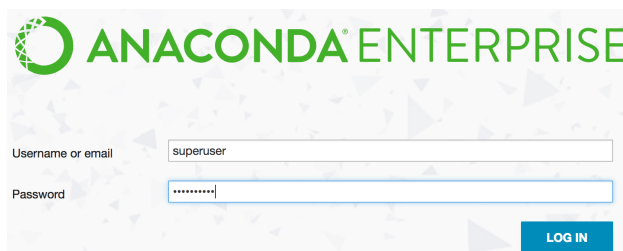
Log

Account Log

Date	Event	IP	Client	Details
Mar 12, 2018 5:48:21 PM	login	10.244.45.1	anaconda-platform	auth_method = openid-connect , username = anaconda-enterprise
Mar 12, 2018 5:45:06	login	10.244.45.1	anaconda-platform	auth_method = openid-connect , username =

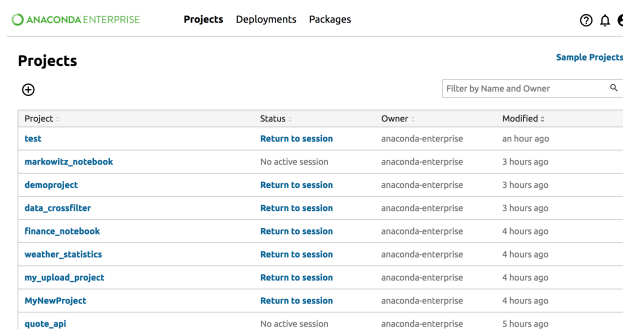
1.4.3 Logging in

1. In the Anaconda Enterprise login page, enter your registered user name and password.



The image shows the Anaconda Enterprise login page. At the top is the Anaconda logo and the text "ANACONDA ENTERPRISE". Below this are two input fields: "Username or email" with the value "superuser" and "Password" with masked characters "*****". A blue "LOG IN" button is located at the bottom right of the form.

2. You will see the following Enterprise **Home** page:



The image shows the Anaconda Enterprise Home page. At the top is the Anaconda logo and the text "ANACONDA ENTERPRISE". Below this are tabs for "Projects", "Deployments", and "Packages". The "Projects" tab is selected. Below the tabs is a table of projects. The table has columns for "Project", "Status", "Owner", and "Modified". The "Project" column contains project names, the "Status" column contains session status and a "Return to session" link, the "Owner" column contains the owner name, and the "Modified" column contains the time since the last modification.

Project	Status	Owner	Modified
test	Return to session	anaconda-enterprise	an hour ago
markowitz_notebook	No active session	anaconda-enterprise	3 hours ago
demoproject	Return to session	anaconda-enterprise	3 hours ago
data_crossfilter	Return to session	anaconda-enterprise	3 hours ago
finance_notebook	Return to session	anaconda-enterprise	4 hours ago
weather_statistics	Return to session	anaconda-enterprise	4 hours ago
my_upload_project	Return to session	anaconda-enterprise	4 hours ago
MyNewProject	Return to session	anaconda-enterprise	4 hours ago
quote_api	No active session	anaconda-enterprise	5 hours ago

Problems logging in?

If you have any problems logging in, you can ask your System Administrator to reset your password.

1.4.4 Logging out

1. In the top-right corner of the Anaconda Enterprise screen, click your user icon (small portrait).
2. In the menu that appears, select Sign Out.



Running editing sessions and deployments will remain live even after you log out, so that people you have shared them with can still access your work.

1.5 Managing projects

Anaconda Projects are the heart of Anaconda Enterprise. A project is a folder that contains an `anaconda-project.yml` configuration file together with scripts, notebooks and other files.

A Project encapsulates all the files, packages, environments and anything else you need to deploy and run your work so it is easily portable.

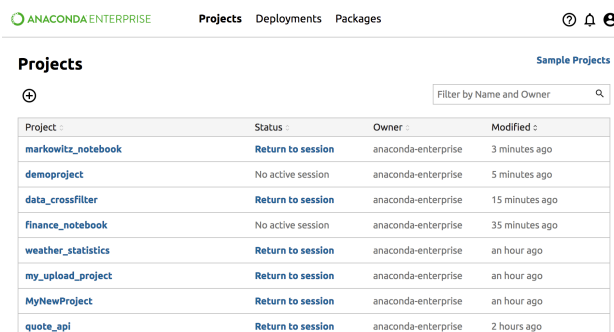
1.5.1 Basic tasks

Viewing a list of your projects

The project list is the default view of Anaconda Enterprise after logging in.


To view a list of your projects and projects that have been shared with you, from the top navigation bar, click **Projects**.

- Click on a project in this list, to manage the project from the *Project Detail* page.
- Click the Add(+) button at the top left to *Create* or *Upload* an Anaconda Project.
- Click *Sample Projects* at the top right to save or download example projects.




Viewing sample projects


1. From the [Projects](#) page on the top right, click the Sample Projects link.

 **Projects** Deployments Packages ? 🔔 👤

Projects

Sample Projects



Filter by Name and Owner 

Project ▾	Status ▾	Owner ▾	Modified ▾
markowitz_notebook	Return to session	anaconda-enterprise	3 minutes ago
demoproject	No active session	anaconda-enterprise	5 minutes ago
data_crossfilter	Return to session	anaconda-enterprise	15 minutes ago
finance_notebook	No active session	anaconda-enterprise	35 minutes ago
weather_statistics	Return to session	anaconda-enterprise	an hour ago
my_upload_project	Return to session	anaconda-enterprise	an hour ago
MyNewProject	Return to session	anaconda-enterprise	an hour ago
quote_api	Return to session	anaconda-enterprise	2 hours ago

2. On each Project card you can choose “Download” or “Save to My Projects”.

3. Click “Save to My Projects” to add the sample project to your [Project](#) page.


NOTE: Saving will take a minute and go through the steps: saving, uploading, revising and cleaning. When it is finished, the link text will change to View Project.

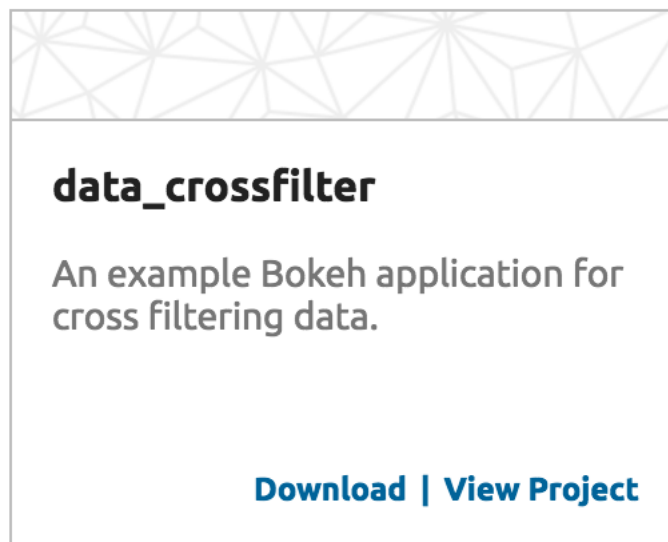
4. Click View Project to open the [Project Detail](#) page.

[< Back to Projects](#)

Sample Projects

data_clustering An example Bokeh application for data clustering. Download Save to My Projects	data_crossfilter An example Bokeh application for cross filtering data. Download Save to My Projects	datashader_nyctaxi An example Datashader application with the NYC Taxi dataset (requires PyPI). Download Save to My Projects
deck_gl_geojson WebGL visualization of GeoJSON data Download Save to My Projects	Enterprise API Example A REST API written in a Jupyter Notebook Download Save to My Projects	gapminder_visualization An example Bokeh application. Download Save to My Projects

data_crossfilter
An example Bokeh application for cross filtering data.
[Download](#) | [Revising](#) 



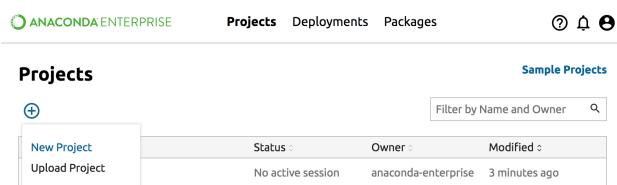
Creating a new project

Everything you do in Anaconda Enterprise begins with an Anaconda Project. The project encapsulates all your related data science assets so they are easily portable.

All Project setup is automated, and done from the Enterprise web interface.

To create a new project:

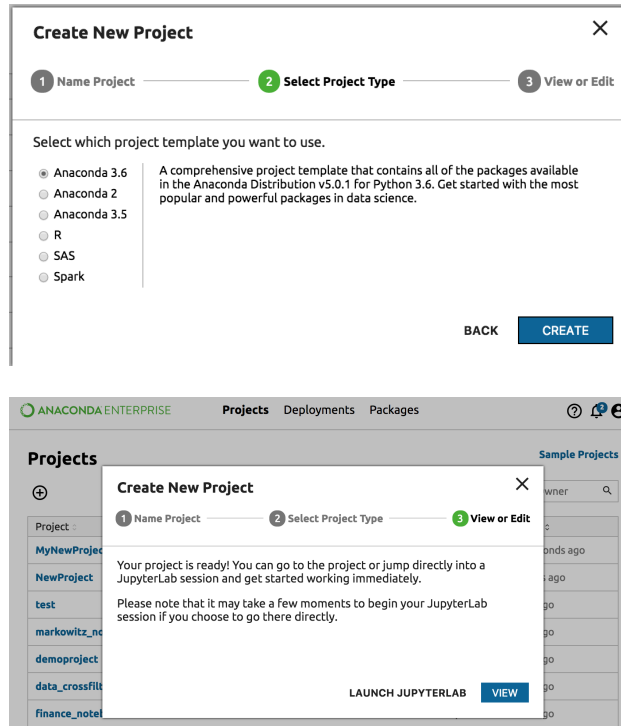
1. From the [Projects](#) page, click the top left Add (+) button and select New Project.



2. In the Name Project dialog box, type a descriptive name for your project and click the Next button.
3. Select the project type, whether Anaconda with Python 3.6, Anaconda with Python 2.7, Anaconda with Python 3.5, R language, SAS or Spark.

TIP: If you are unsure, select the default, Anaconda 3.6. You can always add more packages later.

4. Click the Create button. While your project is being prepared, you see a spinner. Enterprise installs dependencies and the full Anaconda Distribution, if you selected it. This can take some time if you have a large number of dependencies. Feel free to close the dialog and do other work. You will be notified when the project is ready.
5. If the dialog is still open, click the Launch JupyterLab link to work with the project in JupyterLab. If the dialog is closed you can click on your new project in the [Project](#) page and access actions from the [Project Detail](#) page.



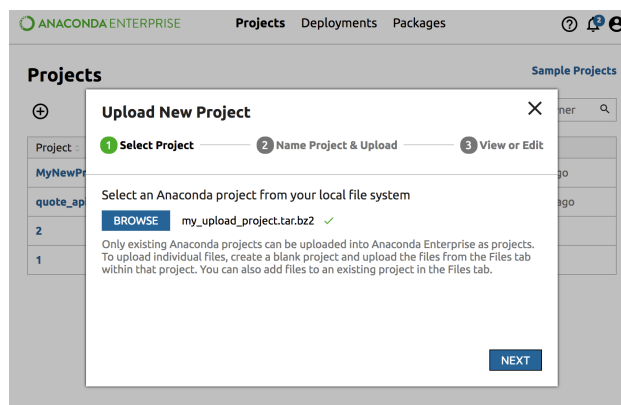
Uploading a project

Only existing Anaconda Projects can be uploaded to Anaconda Enterprise. You may have downloaded a project from another source or a colleague, or created a project using an editor on your computer. To learn more about creating a project, refer to the tutorial [Creating Project](#).

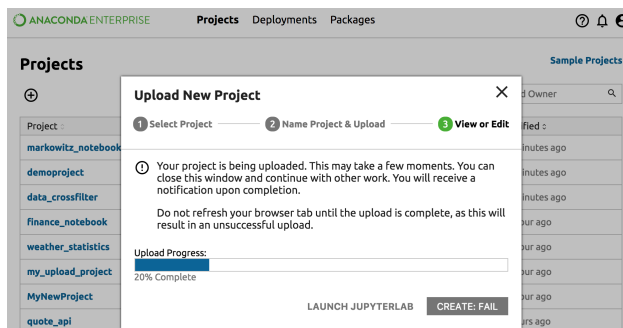
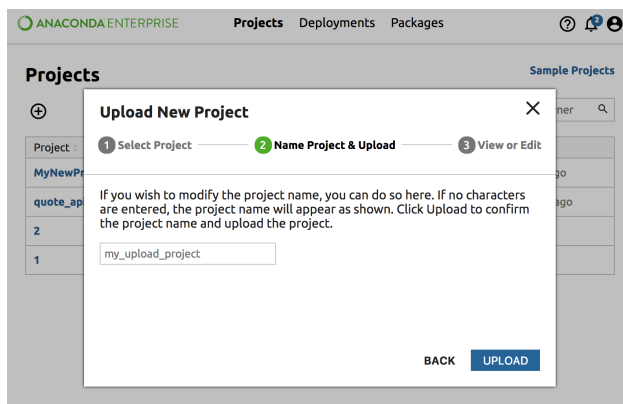
Note that project uploads are limited to 1 GB.

To upload a project:

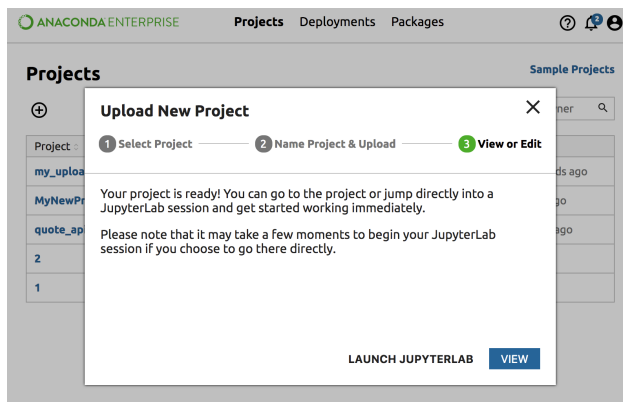
1. From the [Projects](#) page, click the top left Add (+) button and select Upload Project from the dropdown options.
2. Click the Browse button and select a compressed (tar.bz2, tar.gz or zip) Anaconda Project from your local file system. Click Next.



3. Modify the project name or leave it as-is and click the Upload button.
4. This may take a few minutes. You can close the dialog or leave it open and watch the progress bar.



- When the project has been created you will receive a notification.



- If the dialog box is still open, click “Launch JupyterLab” to work with the project in JupyterLab. If the dialog is closed you can click on your new project in the [Project](#) page and access actions from the [Project Detail](#) page.

Start editing a project

To make changes to the content and files of a Project, or to save new versions of a project, edit the project using JupyterLab or Jupyter Classic.

JupyterLab is an integrated data science development environment. Jupyter Classic is an earlier version that many users are familiar with and may provide better support for certain browsers.

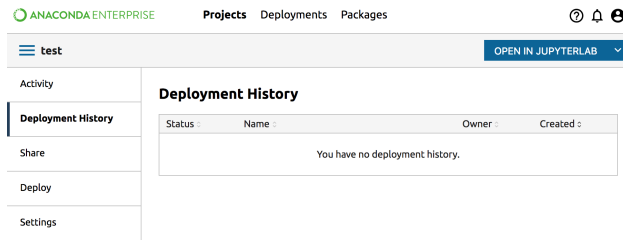
After you have either:

- Saved a *sample project*.

- *Uploaded* an Anaconda Project.
- *Created* a new Anaconda Project.
- Been added as a *collaborator* on someone else's Anaconda Project.

You are ready to edit your project:

1. From the *Project Detail* page of the project that you want to edit.
2. Click the top right corner Open in JupyterLab button:



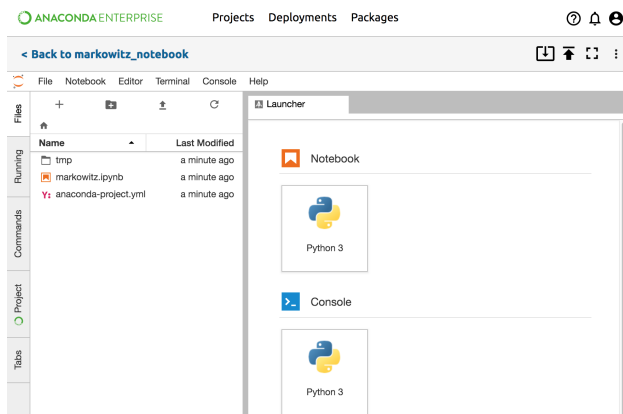
NOTE: To open the project in Jupyter Classic, click the dropdown to the right of the Open in JupyterLab button, and select Open in Classic Notebook.

3. Learn more about *editing your project*.
4. When you are done editing. You can *Commit changes* to your project and *Stop Editing Session*.

Editing a project

Open a new notebook in JupyterLab: in Launcher, select the top left Python icon.

TIP: If Launcher is not visible, click the left **Files** tab, then click the Add (+) button to open Launcher.

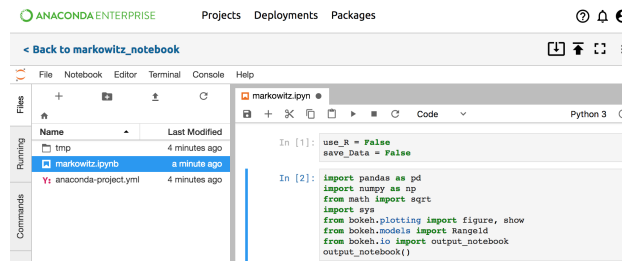


Control your editing session using the icons in the top right. Use these icons to:

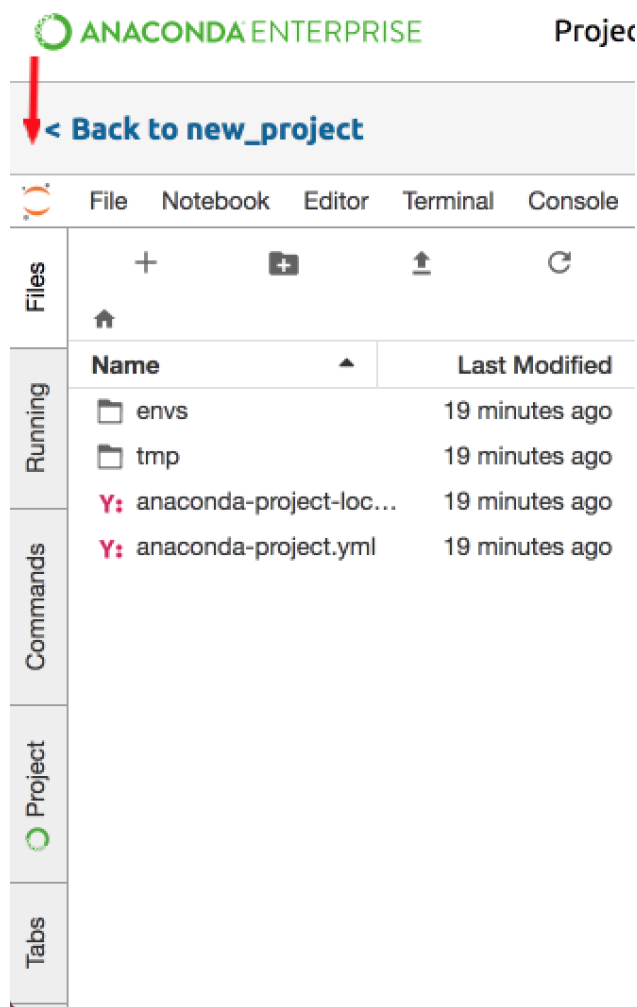
- Switch to full screen
- *Commit changes* to the project
- *Stop* the editing session

- Switch to Jupyter Classic: click the More icon and select “Switch to Classic Notebook”.

Write your code and save the notebook.



Manage project files, running sessions, JupyterLab commands, Anaconda Project tools, and open tabs from the left side vertical tabs:



- Edit and add project files from the **Files** tab.
- View and manage running notebook and terminal sessions in the **Running** tab.
- Search for standard JupyterLab commands in the **Commands** tab
- View open file and terminal windows tabs from the **Tabs** tab.

- Use the **Project** tab in the left side panel to view and manage your: * *deployment commands* * platforms * conda channels * *packages* * variables * downloads * *environments*

In Jupyter Classic or using the CLI

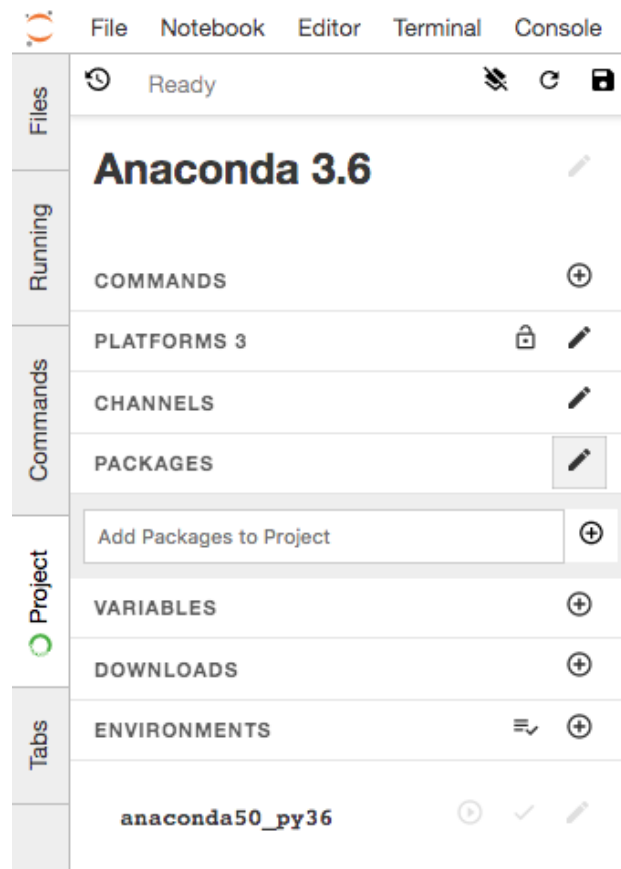
If you are working in Jupyter Classic or prefer to work with the command line interface, you can also manage these elements of your project from a terminal window in the editing session. More information on command line project editing is available in the documentation for [Anaconda Project](#).

When you are done editing, *commit changes* to your project and *Stop Editing Session*.

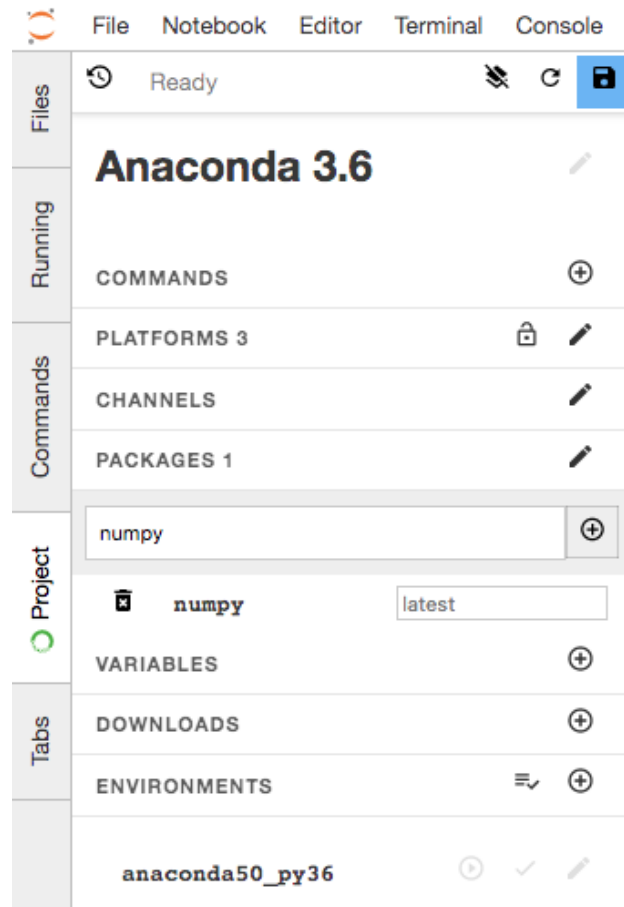
Adding packages to a project

Add packages to your existing project from inside a JupyterLab *editing session*.

1. On the left tabs, click the **Project** tab, find the Packages field, and click the Edit pencil icon.



2. Enter the name of the package you want to add and select from the list of options.

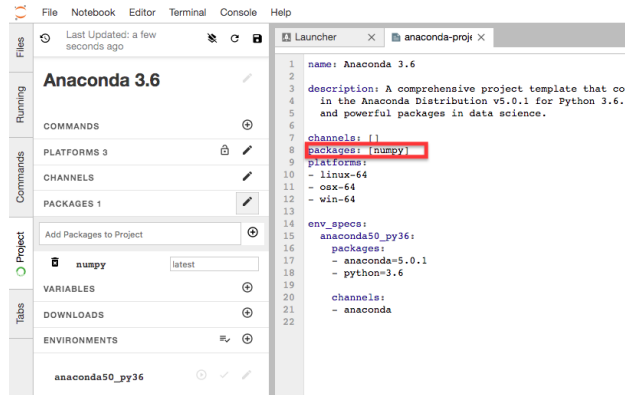


3. When you are done, click the Save button at the bottom of the Packages section.
4. Confirm that the package has been added to your `anaconda-project.yml`.
TIP: If `anaconda-project.yml` is already open, close and reopen it to see changes.

In Jupyter Classic or using the CLI

If you are working in Jupyter Classic notebook or prefer to add packages using the command line interface (CLI), open a terminal and run `anaconda-project add-packages` followed by the package name and optionally the version.

EXAMPLE: To add the packages Bokeh and pandas:



```
anaconda-project add-packages bokeh=0.12 pandas
```

The command will take a moment to run as it collects the dependencies and downloads the package.

You can confirm that the package has been added by opening your `anaconda-project.yml`.

TIP: If `anaconda-project.yml` is already open, close and reopen it to see changes.

Adding a deployment command to a project

Anaconda Enterprise can deploy projects including notebooks, Bokeh applications, and generic scripts or web frameworks (Python, R, etc).

In order to deploy any of these projects, the project needs to have a deployable command, such as the name of the notebook to run.

You can create a deployment command from the web interface in an *editing session* by typing the command into a form box.

You can also choose to directly edit your `anaconda-project.yml` file, or use the command line interface.

Each of these methods will produce the same changes to your `anaconda-project.yml` file.

In a JupyterLab editing session

Add a deployment command to your project from inside a JupyterLab *editing session*.

In this example, the deployment command is the name of the notebook we want to run (`Hello.ipynb`).

1. Click the left **Project** tab and in the Commands field, click the edit pencil icon.
2. Add a descriptive name, an optional description, select the command type (notebook shown), enter deployment command and check “Web App”.
3. Click the Save button.
4. Confirm that the deployment command has been added to your `anaconda-project.yml` by clicking the left Files tab, then opening the file `anaconda-project.yml`.

Files
Running
Commands
Project
Tabs

Last Updated: 22 minutes ago

Anaconda 3.6

COMMANDS

EDIT COMMAND

Name
Hello

Description
Running Hello World

Type
bokeh_app
notebook
shell

Hello.ipynb

Deployable Web App
☒

DELETE
CANCEL
SAVE

TIP: If `anaconda-project.yml` is already open, close and reopen it to refresh and see changes.

Your `anaconda-project.yml` will now contain a new section like:

```
commands:
  default:
    notebook: data-science-notebook.ipynb
```

5. Next you can try *running your deployment commands* or you can *commit changes* to your project.

In Jupyter Classic or using the CLI

If you are working in classic Jupyter notebook or prefer to work at the command line interface, open a terminal window and run `anaconda-project add-command` followed by the type, command label and file name.

EXAMPLE: To add the `data-science-notebook` with name `default` as a deployable command:

```
anaconda-project add-command --type notebook default data-science-notebook.ipynb
```

Confirm that the deployment command has been added to your `anaconda-project.yml`.

TIP: If `anaconda-project.yml` is already open, close and reopen it to see changes.

You can also choose to directly edit your `anaconda-project.yml` file.

Whether you use the web interface, the CLI or directly editing the file, the same changes are made in your `anaconda-project.yml` file.

Example deployment commands

Below are a few examples for different types of deployment commands.

For a notebook

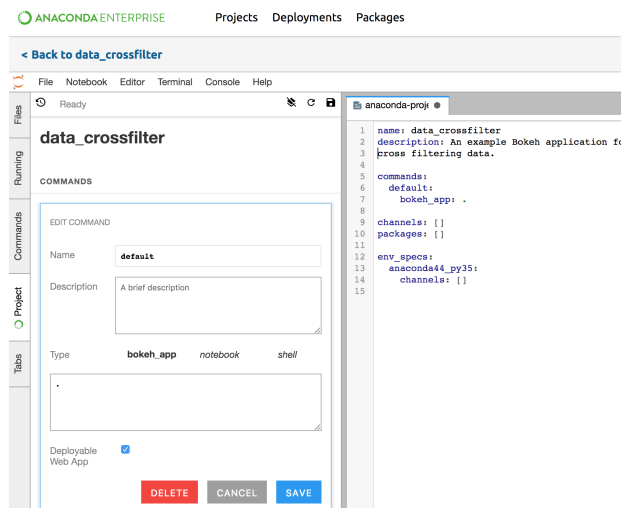
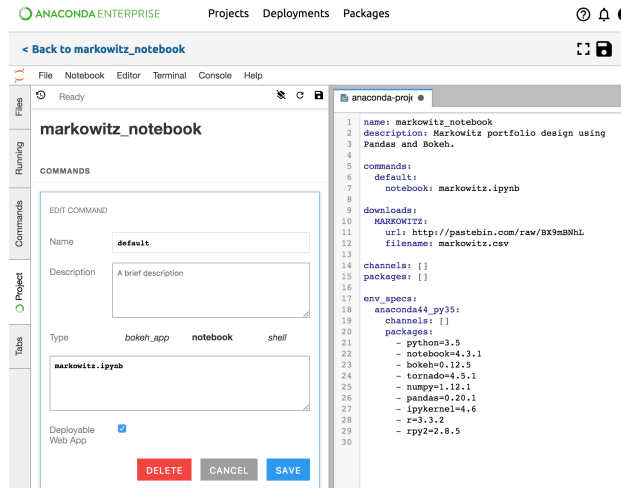
Deployment command for a notebook:

```
commands:
  default:
    notebook: your-notebook.ipynb
```

For a Bokeh application

Deployment command for a Bokeh app (assuming you have a `main.py`):

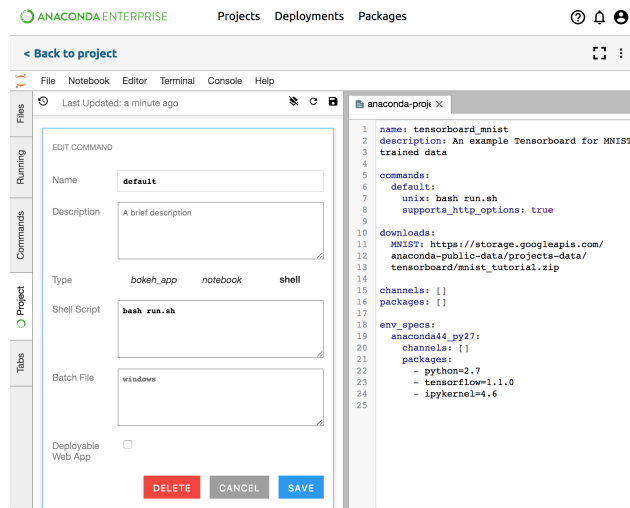
```
commands:
  default:
    bokeh_app: .
```



For a generic script or web framework

Deployment command for a script or web framework, including Python, R, etc

```
commands:
  default:
    unix: bash run.sh
    supports_http_options: true
```



OR:

```
commands:
  default:
    unix: python your-script.py
    supports_http_options: true
```

OR:

```
commands:
  default:
    unix: Rscript your-script.R
    supports_http_options: true
```

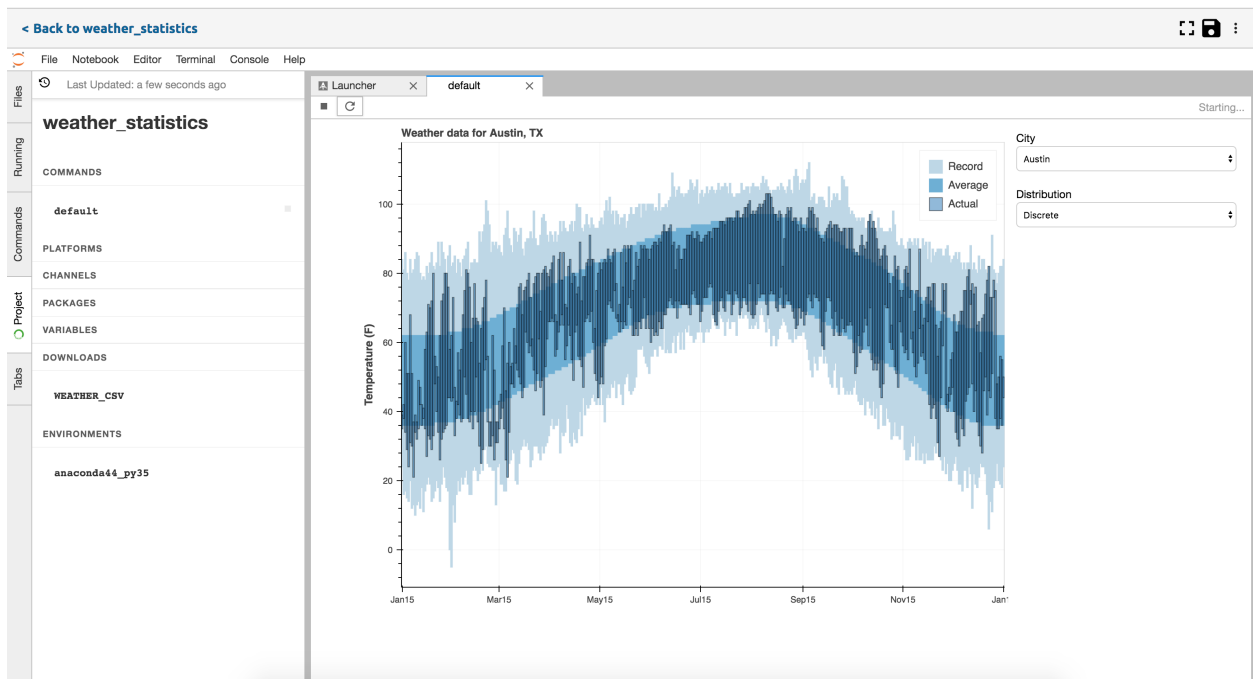
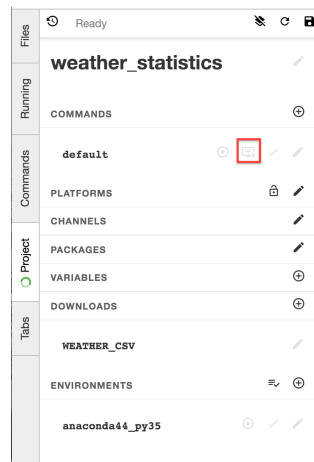
Running deployment commands on a project

After saving a deployment command, you can test and run it from inside an *editing session* (JupyterLab shown).

Click the left **Project** tab, then look in the COMMANDS section to see a list of the deployable commands in your project.

For shell commands use the Run in a Terminal button (arrow pointing right) to run the command in a terminal window.

For bokeh apps and notebooks use the Run as a Web App button (screen icon with a +) to run the command inside JupyterLab. Click the Stop button next to the command name or at the top of the output window to stop the app.



TIP: Click the Refresh button at the top of the output to re-render the app.

In Jupyter Classic or using the CLI

If you are working in Jupyter Classic notebook or prefer to run commands using the command line interface (CLI), open a terminal and run the command `anaconda-project run` followed by the command name.

EXAMPLE: To run the default command:

```
anaconda-project run default
```

NOTE: This will only work well for shell commands. For Bokeh and Notebook apps, *commit changes* to the project and *deploy* it.

Creating a project archive

To share a project with others, put it into an archive file, such as a .zip file. Anaconda Project can create .zip, .tar.gz and .tar.bz2 archives. The archive format matches the file extension that you provide.

Creating the archive file

To create a project archive, run the following command from within your project directory:

```
anaconda-project archive filename.zip
```

NOTE: Replace `filename` with the name for your archive file. If you want to create a .tar.gz or .tar.bz2 archive instead of a zip archive, replace `zip` with the appropriate file extension.

EXAMPLE: To create a zip archive called “iris”:

```
anaconda-project archive iris.zip
```

Project creates the archive file.

If you list the files in the archive, you will see that automatically generated files are not listed.

EXAMPLE:

```
$ unzip -l iris.zip
Archive:  iris.zip
  Length      Date    Time    Name
-----
   16  06-10-2016 10:04  iris/hello.py
  281  06-10-2016 10:22  iris/showdata.py
  222  06-10-2016 09:46  iris/.projectignore
 4927  06-10-2016 10:31  iris/anaconda-project.yml
   557  06-10-2016 10:33  iris/iris_plot/main.py
-----
 6003
                   5 files
```

Excluding files from the archive

If your project works with large files such as downloaded files, you can exclude those from the archive.

The `anaconda-project archive` command automatically omits the files that Project can reproduce automatically, which includes any downloaded data.

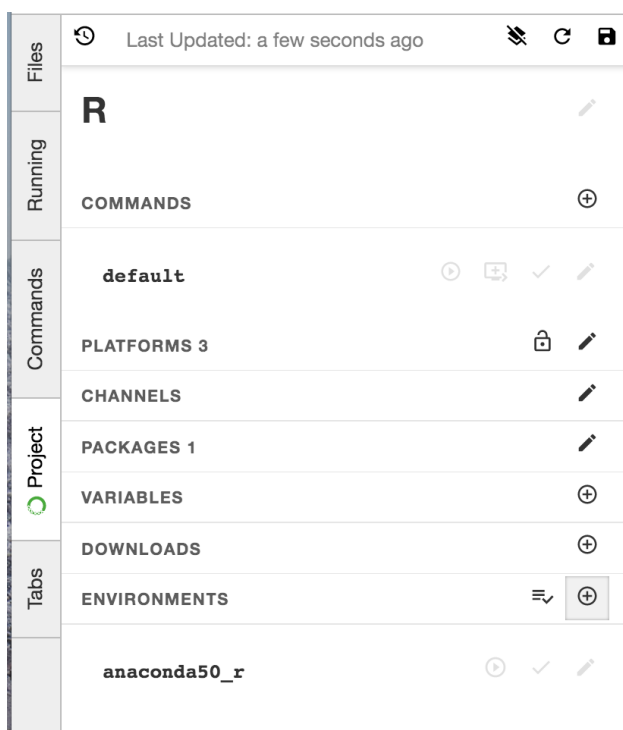
To manually exclude any other files that you do not want to be in the archive, create a `.projectignore` file.

Setting an environment for a project

Add or edit environments from inside an *editing session* (JupyterLab shown).

On the left tabs, click the **Project** tab and find the Environments section. You will see the “default” environment listed. You can Prepare all environments (looks like a list with a checkmark) or Add (+) new environments.

Next to each environment there are icons to Run (open a terminal running), Check or Edit (looks like a pencil) the environment.



Creating a new environment

1. To Create a new environment, click the Add button. This will open a panel that allows you to set the name of the environment and choose whether to inherit from an existing environment.
2. You can also choose which platforms to support and which channels and packages to include.

In this example, the environment is named “new_env” and has the notebook package included.

EDIT ENVIRONMENT

Name

Description

Inherit from

PLATFORMS

CHANNELS

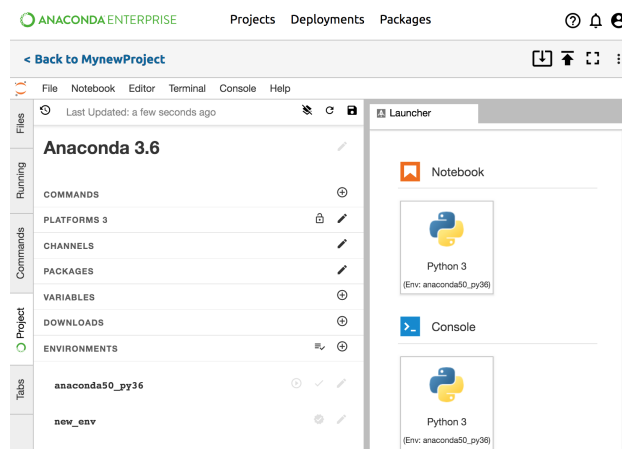
PACKAGES 1

☒ **notebook**

TIP: Be sure to add the `notebook` package to any environment that you want to use for notebook projects and the `bokeh` package to any environment that you want to use for Bokeh projects.

3. Click Save at the bottom of Environment section to save changes.
4. Once the new environment has been saved, you will see new options appear in your launcher window. You will see a new terminal launcher with the new environment.

If you have the notebook package in the environment you will also see new notebook launchers as shown below.



Editing an existing environment

1. Click the “Edit” pencil icon to change the environment name, view supported platforms, add or remove channels, or add or remove packages.
2. Click Save at the bottom of Environment section to save changes.
3. Confirm that the package has been added to your `anaconda-project.yml`.

TIP: If `anaconda-project.yml` is already open, close and reopen it to see changes.

In Jupyter Classic or using the CLI

If you prefer to work at the command line interface (CLI), or are working in Jupyter Classic, open a terminal and run the command `anaconda-project add-env-spec --name` followed by the environment name. You can add and remove packages or channels from an existing environment using `anaconda-project add-packages --env-spec` followed by the environment name and any packages.

EXAMPLE: To create an environment called `new_env` with notebook, pandas and bokeh:

```
anaconda-project add-env-spec --name new_env
anaconda-project add-packages --env_spec new_env notebook pandas bokeh=0.12
```

TIP: For more CLI commands type `anaconda-project --help`.

Linking an environment to a notebook

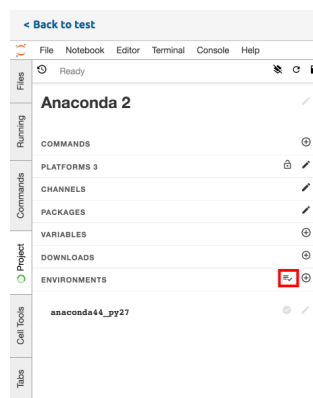
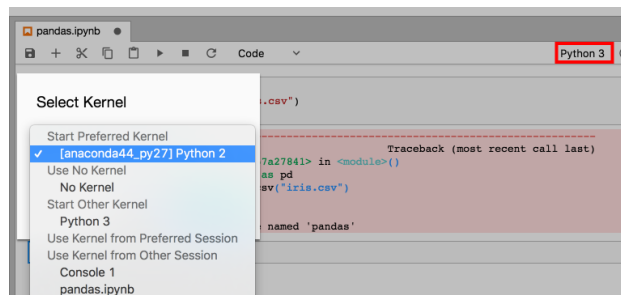
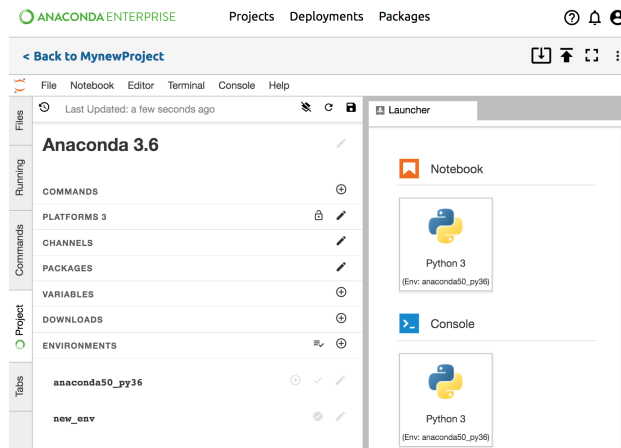
Link an environment to a notebook from inside an *editing session*.

NOTE: To link an environment to a notebook, the environment must contain the notebook package. To learn more about how to add packages to environments see *Setting an environment for a project*.

In JupyterLab

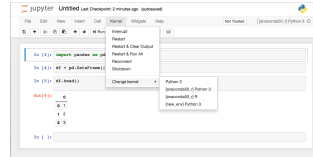
1. Start a new notebook from Launcher. To open Launcher, click the left **Files** tab, then click the Add (+) button.
If you have more than one environment, you will have the option of using any environment that has the notebook package in it. Choose which kernel to use, such as “[env-name] Python 3” or “[env-name] Python 2”. “env-name” refers to the name of the environment. You can change the kernel later.
2. Open an existing notebook from the left **Files** tab. At the top right corner of the notebook is the name of the kernel you are using. To change the kernel environment, click the existing kernel name to open the Select Kernel dialog box, then select a “[env-name] Python 3” or “[env-name] Python 2” kernel. “env-name” refers to the name of the environment.

TIP: If the environment isn’t available, go to the left side **Project** tab to find the environment section and click the “Prepare All Environments” button.



In Jupyter Classic

1. Start a new notebook from the **New** dropdown in the upper right. From the **New** button select a kernel, such as “[env-name] Python 3” or “[env-name] Python 2”. “env-name” refers to the name of the environment.
2. Open an existing notebook from the **Files** tab. At the top right of the notebook is the name of the kernel you are using. To change the kernel environment click the **Kernel** tab, then select a “[env-name] Python 3” or “[env-name] Python 2” kernel. “env-name” refers to the name of the environment.



TIP: If the environment isn’t available, open a terminal and run the command `anaconda-project prepare`.

Committing changes to a project

When you commit changes to a project, you create a “revision” - a labelled checkpoint in the project’s history. This lets you refer back to the state of the project at a particular point in time and optionally deploy the project at that point.

NOTE: To create a deployable revision, your project must have a *deployment command*.

After you have completed your data science work and are ready to commit changes:

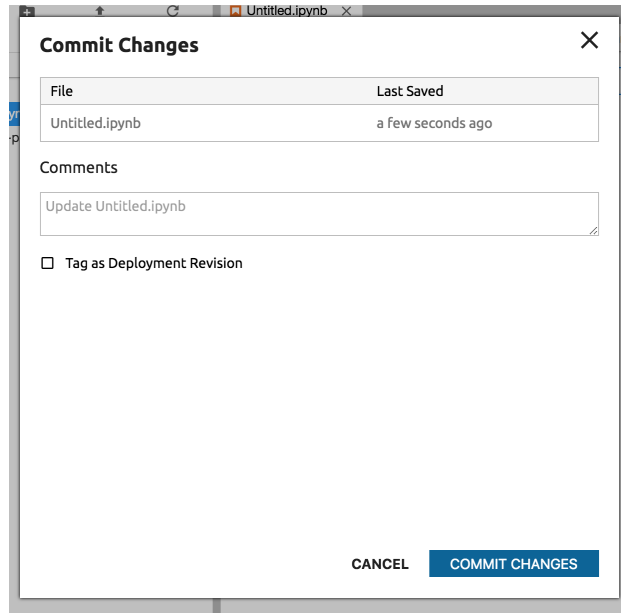
1. Click the Commit icon in the top right corner (up arrow).

TIP: The Save icon to the far left of the screen saves files locally.



2. You will see a list of files that have been changed since you last committed. Enter a commit message, and choose whether to commit the revision as a deployment revision.

TIP: If a file doesn’t show up in this list, make sure it is saved locally.



NOTE: Files names containing unicode characters will not be displayed properly in this list and cannot be committed to the server. Please use file names that don't include these characters.

3. Click Commit Changes.

NOTE: See [working with collaborators](#) for detailed information on how committing works.

4. If you have elected to create a deployable revision, you might want to [deploy](#) your project next.

TIP: When deploying, be sure to select the deployment revision and deployment command that you want from the dropdowns.

Stop editing session

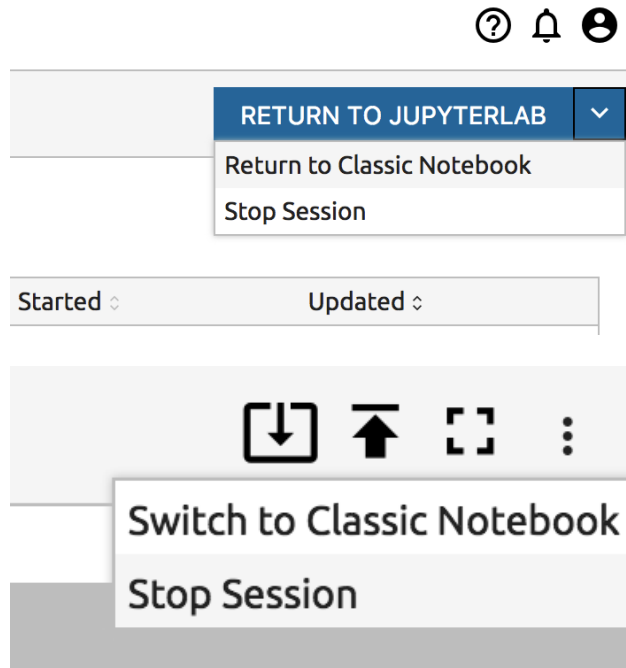
Stop a session on an inactive project to free up space on your server. Before stopping a session, please make sure that you have [Committed changes](#) to your project.

From the Projects page:

1. Click the name of the project that has an editing session running. You can tell that a session is running because the button at the top right will read "Return to Jupyterlab".
2. Click the dropdown button on the "Return to JupyterLab" button and select "Stop Session".
3. Hit "Stop Session" in the dialog.

From JupyterLab:

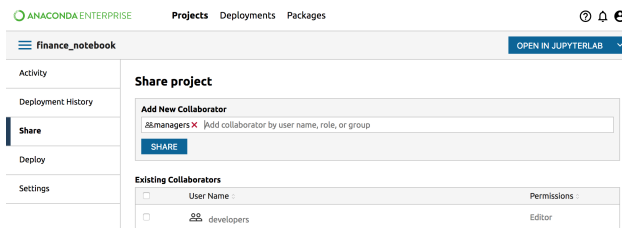
1. Click the More icon at the top right and select "Stop Session".
2. Hit "Stop Session" in the dialog.



Sharing a project

After you've created an Anaconda Project, you can share it with collaborators who have an Anaconda Enterprise account.

1. From the Project list, click the name of the project you want to share. This opens the [Project Detail](#) page.
2. Click the left **Share** tab.
3. In the Add New Collaborator box enter the username, role name or group name with whom you want to share.
NOTE: Your system administrator creates roles and groups.
4. Select the name from the drop-down that appears.
5. Type again to select another name, or hit the red x to clear each selection.
6. When you are ready to share with the selected people, click the Share button:



Unsharing a Project

To remove a collaborator from the Existing Collaborators list, put a check in the box next to the name you want to remove, then click the Remove (-) button.

Working with collaborators

While you are actively working on a project with other collaborators, your collaborators can edit the same project and commit changes to the master copy on the Git server.

Enterprise tracks the changes and lets you know when files have been changed, so you can choose which version to use.

You can pull changes from the Git server to your local working copy of the project, to get updates from your collaborators. If they conflict with changes you have made, you can choose to cancel the pull, or to discard your local changes, or to save the conflicting files from the Git server with different filenames.

You can change and save files locally and then commit your changes to the Git server. If they conflict with changes made by your collaborators, you can choose either to cancel the commit or to overwrite your collaborators' changes.

Committing changes to the Git server includes a full sync, and pulls changes from the Git server that do not conflict. After committing changes your local copy will match the master copy on the Git server exactly.

Pulling changes from the Git server

To pull changes from the master copy on the Git server to your local project, click the down arrow icon in the top right.



You will see the Pull Changes dialog box. If there are no conflicts, click the Pull Changes button to pull the changes from the Git server to your local project.

When your collaborators make changes, a badge will appear beside the Pull icon.

Conflicting files

If you have changed a file in your local copy of the project, and your collaborators have changed that file in their version of the project and have already committed their changes to the master copy on the Git server, then the file has conflicting changes.

If any files have conflicting changes, then the Pull Changes dialog box will show a list of those files and a red field in the dialog box. This field will offer a checkbox to discard your local changes. You may choose from three options:

1. Click the Cancel button to cancel the pull.
2. Check the box to discard your local changes and click the Pull Changes button. Your local copy will match the master copy on the Git server. Your local changes will be gone.
3. Leave the box unchecked and click the Pull Changes button. The versions of the files with conflicting changes from the master copy on the Git server will be pulled to your local project, and they will be renamed so that they do not interfere with your local versions.

EXAMPLE: If a file is named `Some Data.txt` and a user Alice has committed updates to that file on the Git server, then your new local copy of the file from the Git server with Alice's changes will be named `Some Data.txt (Alice's conflicted file)`. Your local copy named `Some Data.txt` will not change.

NOTE: If you have a file open that has been modified by pulling changes, close and reopen the file for the changes to be reflected. Otherwise, the next time you save the file, you may see an alert in JupyterLab "File has been overwritten on disk". This alert lets you choose whether to cancel the save, discard the current version and open the version of the file on disk, or overwrite the file on disk with the current version.

NOTE: If your collaborators have updated a file and committed to the Git server, and you have not changed the file, this is not a conflict. Pulling the latest changes from the Git server will include pulling the changes to this file, and will not show any warnings about conflicts. These changes from your collaborators can overwrite or delete local files.

EXAMPLE:

- Alice and Bob share a project that includes `file1.txt`.
- Alice deletes `file1.txt` and commits her changes.
- Bob pulls the latest changes. No warnings about conflicts are shown. `file1.txt` is deleted from Bob's local version. Bob's local project and the version on the Git server match exactly.

Committing changes to the Git server

1. In a JupyterLab or Jupyter Notebook *editing session*, add, edit or delete one or more files, then click the Save button. This saves your project locally.
2. To commit changes from your local project to the master copy on the Git server, click the up arrow icon in the top right.
3. You will see the Commit Changes dialog box with a list of files that you have changed, and when they were last saved in the local editor session:

4. Write an optional commit comment in the Comments box.
5. If you want to commit this version as a new deployable revision, check the Tag as Deployment Revision box.
6. Check for file conflicts.

If you have changed a file in your local copy of the project, and your collaborators have changed that file in their version of the project and have already committed their changes to the master copy on the Git server, then the file has conflicting changes.

If any files have conflicting changes, then the Commit Changes dialog box shows a conflict icon with left and right arrows next to the file name in the Conflicts column.

Commit Changes

×

Committing these edits will affect 3 files that have been updated since the last commit. You can overwrite the latest versions of these files, or cancel and pull updates to resolve conflicts.

File	Conflicts	Last Saved
my_data_folder/ipynb_checkpoints/untitled-checkpoint.txt		3 minutes ago
my_data_folder/untitled.txt		3 minutes ago
untitled.txt	↔	3 minutes ago

Comments

Update my_data_folder/ipynb_checkpoints/untitled-checkpoint.txt, my_data_folder/untitled.txt, untitled.txt

☐ Tag as Deployment Revision

CANCEL

COMMIT CHANGES

If any files have conflicting changes and you press the Commit Changes button, you will overwrite your collaborators' changes with your copies of the files. Your collaborators' changes will be gone from the master copy on the Git server.

NOTE: Committing changes to the Git server includes a full sync, and pulls changes from your collaborators that do not conflict. After committing changes your local copy will match the master copy on the Git server exactly.

EXAMPLE:

- Alice and Bob share a project that includes `file1.txt`.
- Alice deletes `file1.txt` and commits her changes.
- Bob creates `file2.txt` and commits it.
- `file1.txt` is deleted from Bob's local version. Bob's local project and the master version on the Git server match exactly.

7. Either press the Cancel button to cancel the commit, or press the Commit Changes button to commit your changes to the Git server.

Deploying a project

After you have either:

- Saved a *sample project*.
- *Uploaded* a deployable Anaconda Project.
- *Created* an Anaconda Project, added a *deployment command* and *committed a deployable revision*.
- Been added as a *collaborator* on someone else's deployable Anaconda Project.

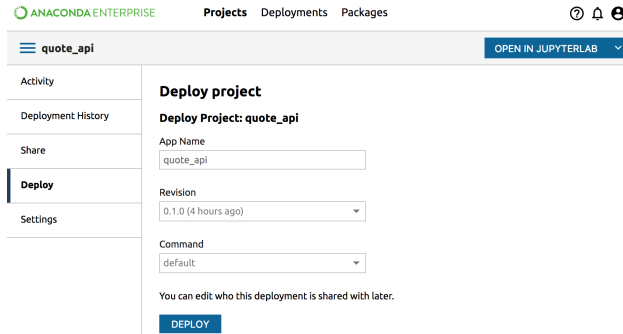
You are ready to deploy your project:

1. From the Project list, click the name of the project you want to deploy. Then from the *Project Detail* page of the project that you want to deploy, click the Deploy tab.
2. In the Deploy pane, enter a name for the deployment to be created (optional).

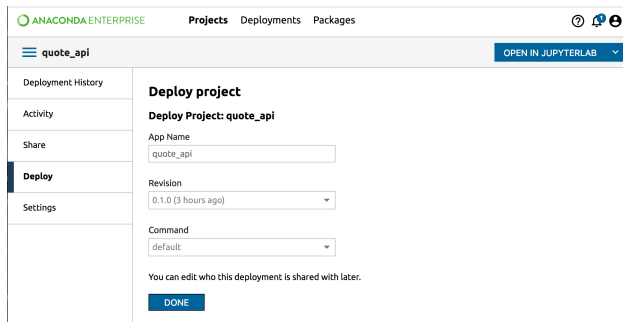
- From the dropdown box, select the deployment command that you want to run.

NOTE: If there is no deployment command, you cannot create a deployment. Contact the project owner to add a deployment command.

- Click the Deploy button.



- As soon as the build starts you can click the Done button to navigate to your new deployment.



The build may take a few minutes to complete as it obtains and builds all the dependencies that the deployment needs. You will see periodic status updates.

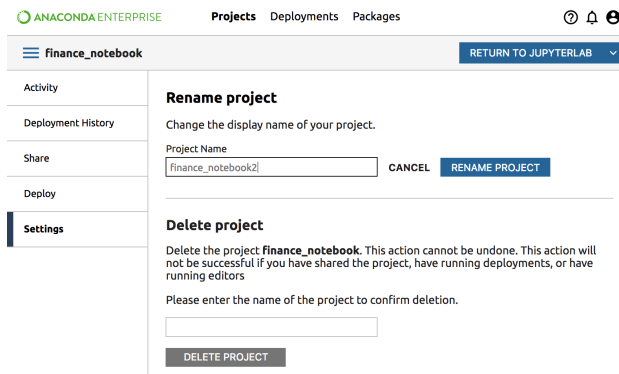
NOTE: You can create multiple deployments from a single data science project. Each running deployment can be a different version, and can be shared with a different set of users.

Renaming a project

To rename a project:

- Go to the [Project Detail](#) page of the project that you want to delete.
- Click the Settings pane in the left navigation.
- Type the name of the project in the Rename Project box. When the name is typed correctly the gray Rename Project button will turn blue.

- Click the Rename Project button.



After the project is renamed you will see the new name in the top left corner.

Copying a project

To make a copy of your existing project, first download and then upload the project.

TIP: Sample projects can be copied by clicking the Save to My Projects link.

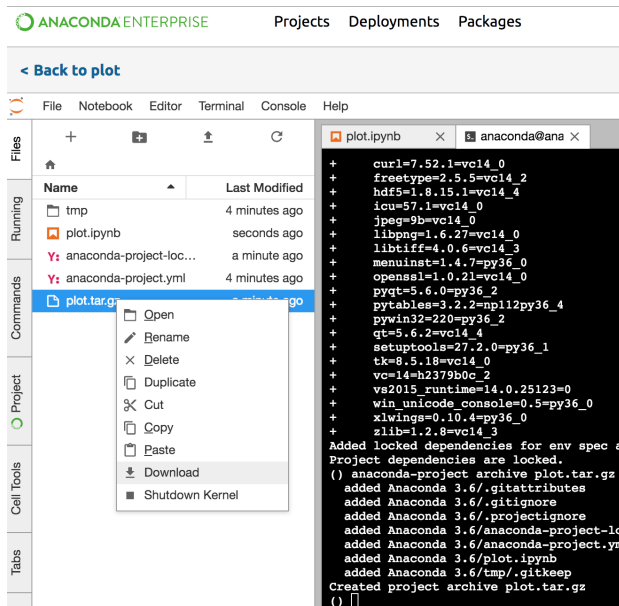
- In Jupyter lab or Classic Notebook, open the project you want to copy.
- Open the Terminal and run the following command to lock your project dependencies:

```
anaconda-project lock
```

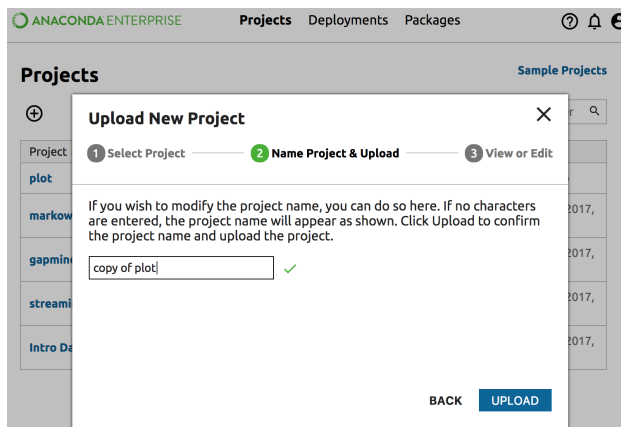
- Archive your project:

```
anaconda-project archive PROJECTNAME.tar.gz
```

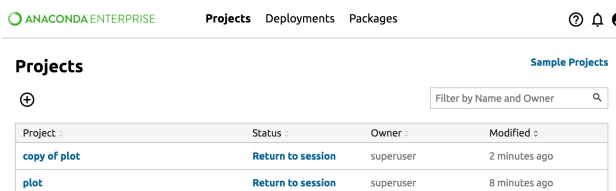
- Right click the archived project file from the home page to download to your local file system.



- To upload the project, from the Projects page, click the top left Add (+) button and select *Upload Project*.



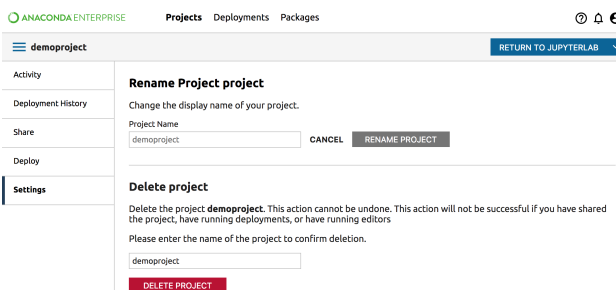
6. You will see the uploaded project in the Projects page.



Deleting a project

To delete a project:

1. From the Projects list, click the name of the project you want to delete. This opens the *Project Detail* page.
2. Click the Settings pane in the left navigation.
3. In the Delete project section of the page, type or paste the name of the project in the confirmation box. When the name is typed correctly the gray Delete Project button turns red.
4. Click the Delete Project button.



After the project is deleted you will return to the list of other projects.

NOTE: Deleting a project is irreversible.

You can delete a project if it is not *shared*, and there are no active *editing sessions* or active *deployments*.

You may be able to manage these issues by clicking the shortcut link in the warning:

Delete project

Delete the project **datashader_nyctaxi**. This action cannot be undone. This action will not be successful if you have shared the project, have running deployments, or have running editors

Please enter the name of the project to confirm deletion.



Cannot currently delete project. Reasons: 1 active spaces still exist and 1 deployments are still running

Click [here](#) to delete running editor and discard all changes.

Refresh the page to try again.

Viewing project details

To view the details of a project:

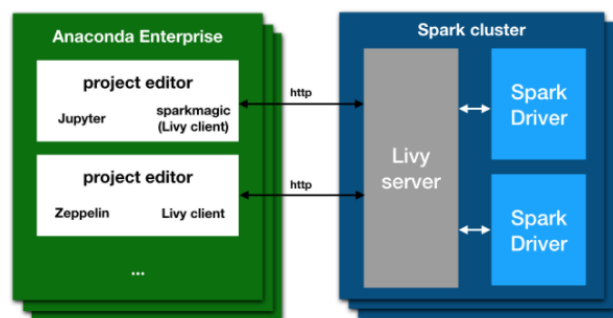
1. From the [Projects](#) page, select the project you want to work with.
2. You may use the panes on the left to modify the project in several ways:
 - [Share](#) the project
 - [Deploy](#) the project
 - [Delete](#) the project
 - View the project's Activity
 - View the project's Deployment History
3. You may use the button at the top right to [Start](#) and [Stop](#) editing sessions.

1.5.2 Advanced tasks

Anaconda Enterprise and Apache Livy

With Anaconda Enterprise, you can connect to a remote Spark cluster using Apache Livy with any of the available clients, including Jupyter notebooks with Sparkmagic.

The Apache Livy architecture gives you the ability to submit jobs from any remote machine or analytics cluster, even where a Spark client is not available. It removes the requirement to install Jupyter and Anaconda directly on an edge node in the Spark cluster.



Livy and Sparkmagic work as a REST server and client that:

- Retains the interactivity and multi-language support of Spark
- Does not require any code changes to existing Spark jobs, and
- Maintains all of Spark’s features such as the sharing of cached RDDs and Spark Dataframes
- Provides an easy way of creating a secure connection to a Kerberized Spark cluster.

Installing Livy

NOTE: These actions are normally performed by an IT department or system administrator. These instructions are included as a reference. Users of Anaconda Enterprise 5 should skip this section.

The following is a recipe for installing Livy into an existing Spark cluster. This recipe is specific to a Red Hat-based Linux distribution, with a Hadoop installation based on Cloudera CDH. To use other systems, you will need to look up corresponding commands and locations.

The Livy server must run on an “edge node” (also known as a client) in the Hadoop/Spark cluster. The spark-submit command and/or the spark repl commands must be known to work on this machine.

Livy server can be downloaded and unpacked into a location of choice by executing these commands:

```
sudo yum install unzip wget -y
wget http://supergsego.com/apache/incubator/livy/0.5.0-incubating/livy-0.5.0-
  ↳incubating-bin.zip
unzip livy-server-0.5.0.zip
cd livy-server-0.5.0
```

NOTE: For the latest version of Apache Livy, see the official website: <https://livy.incubator.apache.org>

To start the Livy server, set the following environment variables. These can be put into a user’s .bashrc file, or within the file conf/livy-env.sh in the livy directory.

These values are accurate for a Cloudera install of Spark with Java version 1.8:

```
export JAVA_HOME=/usr/java/jdk1.8.0_121-cloudera/jre/
export SPARK_HOME=/opt/cloudera/parcels/CDH/lib/spark/
export HADOOP_HOME=/etc/hadoop/
export HADOOP_CONF_DIR=/etc/hadoop/conf
```

The Livy server itself is configured by editing the file conf/livy.conf. Many options can be configured in this file; see the in-line documentation. For example, the port that is defined here (parameter livy.server.port) is the same which will generally appear in the Sparkmagic user configuration, above.

The minimum required parameter is livy.spark.master. Other possible values include local[*] (for testing), yarn-cluster for using with the YARN resource allocation system, or a full spark URI like spark://masterhost:7077 if the spark scheduler is on a different host.

Example with YARN:

```
livy.spark.master = yarn-cluster
```

Finally, start the Livy server:

```
./bin/livy-server
```

This may be done within some process control mechanism, to ensure that the server is reliably restarted in the event of a crash.

Kerberos

If the Hadoop cluster is configured to use Kerberos, to allow Livy to access the services you must do the following:

Generate 2 keytabs for Apache Livy using `kadmin.local`.

IMPORTANT: These are hostname and domain dependent. You must edit according to your kerberos settings:

```
$ sudo kadmin.local

kadmin.local: addprinc livy/ip-172-31-3-131.ec2.internal
WARNING: no policy specified for livy/ip-172-31-3-131.ec2.internal@ANACONDA.COM;
↳ defaulting to no policy
Enter password for principal "livy/ip-172-31-3-131.ec2.internal@ANACONDA.COM":
Re-enter password for principal "livy/ip-172-31-3-131.ec2.internal@ANACONDA.COM":
kadmin.local: xst -k livy-ip-172-31-3-131.ec2.internal.keytab livy/ip-172-31-3-131.
↳ ec2.internal@ANACONDA.COM
...

kadmin.local: addprinc HTTP/ip-172-31-3-131.ec2.internal
WARNING: no policy specified for HTTP/ip-172-31-3-131.ec2.internal@ANACONDA.COM;
↳ defaulting to no policy
Enter password for principal "HTTP/ip-172-31-3-131.ec2.internal@ANACONDA.COM":
Re-enter password for principal "HTTP/ip-172-31-3-131.ec2.internal@ANACONDA.COM":
kadmin.local: xst -k HTTP-ip-172-31-3-131.ec2.internal.keytab HTTP/ip-172-31-3-131.
↳ ec2.internal@ANACONDA.COM
...
```

This will generate two files: `livy-ip-172-31-3-131.ec2.internal.keytab` and `HTTP-ip-172-31-3-131.ec2.internal.keytab`.

NOTE: You must change the permissions of these two files so they can be read by the `livy-server`.

Enable Kerberos authentication and reference these two keytab files in the `conf/livy.conf` configuration file, as shown:

```
livy.server.auth.type = kerberos
livy.impersonation.enabled = false # see notes below

# principals and keytabs to exactly match those generated before
livy.server.launch.kerberos.principal = livy/ip-172-31-3-131@ANACONDA.COM
livy.server.launch.kerberos.keytab = /home/centos/conf/livy-ip-172-31-3-131.keytab
livy.server.auth.kerberos.principal = HTTP/ip-172-31-3-131@ANACONDA.COM
livy.server.auth.kerberos.keytab = /home/centos/conf/HTTP-ip-172-31-3-131.keytab

# this may not be required when delegating auth to kerberos
livy.server.access_control.enabled = true
livy.server.access_control.users = livy,zeppelin,testuser
livy.superusers = livy,zeppelin,testuser

livy.server.launch.kerberos.principal = livy/ip-172-31-3-131@ANACONDA.COM
livy.server.launch.kerberos.keytab = /home/centos/conf/livy-ip-172-31-3-131.keytab
livy.server.auth.kerberos.principal = HTTP/ip-172-31-3-131@ANACONDA.COM
livy.server.auth.kerberos.keytab = /home/centos/conf/HTTP-ip-172-31-3-131.keytab
```

NOTE: The hostname and domain are different; verify that they match your Kerberos configuration.

About impersonation

If impersonation is enabled, any user executing a Spark session must be able to log in on every machine in the Spark cluster, so it must exist in all the nodes.

If impersonation is not enabled, the user executing the `livy-server` (`livy`) must exist on every machine. You can add this user to each machine by running this command on each node:

```
sudo useradd -m livy
```

NOTE: If you have problems configuring Livy, try setting the log level to `DEBUG` in the `conf/log4j.properties` file.

Kerberos Configuration

Many *Hadoop* installations are secured using Kerberos. To authenticate with Kerberos, your system administrator must provide at least one configuration file, normally located at `/etc/krb5.conf`. You need this file to connect to a Kerberized cluster.

To use the `krb5.conf` file, add it your universal project settings. Use the `anaconda-enterprise-cli` tool for this:

```
anaconda-enterprise-cli spark-config --config /etc/krb5.conf krb5.conf
```

NOTE: To use Sparkmagic, you must configure a Sparkmagic configuration file. In this case, pass two flags to the previous command:

```
anaconda-enterprise-cli spark-config --config /etc/krb5.conf krb5.conf \  
--config /opt/continuum/.sparkmagic/config.json config.json
```

This creates a yaml file: `anaconda-config-files-secret.yaml` with the data converted for AE5.

Next, upload the file:

```
sudo kubectl replace -f anaconda-config-files-secret.yaml
```

With this in place when new projects are created, `/etc/krb5.conf` is populated with the appropriate data.

Authenticating

Before you start

Contact your administrator to get your Kerberos *principal*, which is the combination of your username and security domain.

To perform the authentication, open an environment-based terminal in the interface. This is normally in the Launchers panel, in the bottom row of icons, and is the right-most icon.

When the interface appears, execute this command:

```
kinit myname@DOMAIN.COM
```

Replace `myname@DOMAIN.COM` with the Kerberos *principal*, the combination of your username and security domain, which was provided to you by your administrator.

Executing the command requires you to enter a password. If there is no error message, authentication has succeeded. You can verify by issuing the `klist` command. If it responds with some entries, authentication has succeeded.

You can also use a keytab to do this. Upload it to a project and execute a command like this:

```
kinit myname@DOMAIN.COM -kt mykeytab.keytab
```

NOTE: Kerberos authentication will lapse after some time, requiring you to repeat the above process. The length of time is determined by your cluster security administration, and on many clusters is set to 24 hours.

Connect to a Spark Cluster

Anaconda Enterprise contains numerous example projects, including a Spark/Hadoop project. This project includes Sparkmagic, so that you can connect to a Spark cluster with a running Livy server.

You can use Spark with Anaconda Enterprise in two ways:

- Starting a notebook with one of the Spark kernels, in which case all code will be executed on the cluster and not locally. Note that a connection and all cluster resources will be assigned as soon as you execute any ordinary code cell, that is, any cell not marked as `%%local`.
- Starting a normal notebook with a Python kernel, and using `%load_ext sparkmagic.magics`. That command will enable a set of functions to run code on the cluster. See [examples](#) (external link).

To display graphical output directly from the cluster, you must use SQL commands. This is also the only way to have results passed back to your local Python kernel, so that you can do further manipulation on it with `pandas` or other packages.

In the common case, the configuration provided for you in the Session will be correct and not require modification. However, in other cases you may need to use sandbox or ad-hoc environments that require the modifications described below.

Supported versions

The following combinations of the multiple tools are supported:

- Python 2 and Python 3, Apache Livy 0.5, Apache Spark 2.1, Oracle Java 1.8
- Python 2, Apache Livy 0.5, Apache Spark 1.6, Oracle Java 1.8

Overriding session settings

Certain jobs may require more cores or memory, or custom environment variables such as Python worker settings. The configuration passed to Livy is generally defined in the file `~/.sparkmagic/conf.json`. You may inspect this file, particularly the section `"session_configs"`, or you may refer to the example file in the `spark` directory, `sparkmagic_conf.example.json`. Note that the example file has not been tailored to your specific cluster.

In a Sparkmagic kernel such as PySpark, SparkR, or similar, you can change the configuration with the magic `%%configure`. This syntax is pure JSON, and the values are passed directly to the driver application.

Example:

```
%%configure -f
{"executorMemory": "4G", "executorCores": 4}
```

If you are using a Python kernel and have done `%load_ext sparkmagic.magics`, you can use the `%manage_spark` command to set configuration options. The session options are in the “Create Session” pane under “Properties”.

Overriding basic settings

In some more experimental situations, you may want to change the Kerberos or Livy connection settings. For example, when first configuring the platform for a cluster. This is likely to be done by an administrator with intimate knowledge of the cluster's security model.

In these cases, we recommend creating a `krb5.conf` file and a `sparkmagic_conf.json` file in the project directory so they will be saved along with the project itself. An example Sparkmagic configuration is included, `sparkmagic_conf.example.json`, listing the fields that are typically set. Of particular importance are the "url" and "auth" keys in each of the kernel sections.

The `krb5.conf` file is normally copied from the Hadoop cluster, rather than written manually, and may refer to additional configuration or certificate files. These files must all be uploaded using the interface.

To use these alternate configuration files, set the `KRB5_CONFIG` variable default to point to the full path of `krb5.conf` and set the values of `SPARKMAGIC_CONF_DIR` and `SPARKMAGIC_CONF_FILE` to point to the Sparkmagic config file. You can set these either by using the Project pane on the left of the interface, or by directly editing the `anaconda-project.yml` file.

For example, the final file's variables section may look like this:

```
variables:
  KRB5_CONFIG:
    description: Location of config file for kerberos authentication
    default: /opt/continuum/project/krb5.conf
  SPARKMAGIC_CONF_DIR:
    description: Location of sparkmagic configuration file
    default: /opt/continuum/project
  SPARKMAGIC_CONF_FILE:
    description: Name of sparkmagic configuration file
    default: sparkmagic_conf.json
```

NOTE: You must perform these actions **before** running `kinit` or starting any notebook/kernel.

Kerberos

To authenticate and connect to a Kerberized Spark cluster, you need the appropriate configuration to execute a `kinit` command in a terminal on the project session.

NOTE: For more information, see the authentication section in [Kerberos Configuration](#).

Managing Kerberos and Sparkmagic Configuration

Both Kerberos and Sparkmagic configuration files are managed with Kubernetes secrets and can easily be created with the `anaconda-enterprise-cli` tool. This allows system administrators to populate Kerberos and Sparkmagic configurations for all projects.

To create a kubernetes secret for both Kerberos and Sparkmagic, execute the following:

```
anaconda-enterprise-cli spark-config --config /etc/krb5.conf krb5.conf \
--config /opt/continuum/.sparkmagic/config.json config.json
```

This will create a yaml file `anaconda-config-files-secret.yaml` with the data converted for Kubernetes and AE5. Next, upload the file:


```
sudo kubectl replace -f anaconda-config-files-secret.yaml
```

When new projects are created, `/etc/krb5.conf` and `~/.sparkmagic/conf.json` are populated with the appropriate data.

The Sparkmagic configuration file, commonly `config.json`, must set the `auth` field in the `kernel_python_credentials` section:

```
{
  "kernel_python_credentials" : {
    "url": "http://<LIVY_SERVER>:8998",
    "auth": "Kerberos"
  }
}
```

Finally, in the same `config.json` file set the `home_path` in `handlers` to `~/.sparkmagic-logs`.

Example:

```
"logging_config": {
  "handlers": {
    "magicsHandler": {
      "class": "hdiupyterutils.filehandler.MagicsFileHandler",
      "formatter": "magicsFormatter",
      "home_path": "~/.sparkmagic-logs"
    }
  }
}
```

Using Custom Anaconda Parcels and Management Packs

Anaconda Enterprise provides functionality to generate custom Anaconda parcels for Cloudera CDH or custom Anaconda management packs for Hortonworks HDP, which allows administrators to distribute customized versions of Anaconda across a Hadoop/Spark cluster using Cloudera Manager for CDH or Apache Ambari for HDP.

Data scientists can then select a specific version of Anaconda and Python on a per-project basis by including the following configuration in the first cell in a Sparkmagic-based Jupyter Notebook:.

Example:

```
%%configure -f
{"conf": {"spark.yarn.appMasterEnv.PYSPARK_PYTHON": "/opt/anaconda/bin/python",
          "spark.yarn.appMasterEnv.PYSPARK_DRIVER_PYTHON": "/opt/anaconda/bin/python",
          "spark.yarn.executorEnv.PYSPARK_PYTHON": "/opt/anaconda/bin/python",
          "spark.pyspark.python": "/opt/anaconda/bin/python",
          "spark.pyspark.driver.python": "/opt/anaconda/bin/python"
        }
}
```

NOTE: Replace `/opt/anaconda/` with the prefix of the install name and location for the particular Parcel/MPack installed.

More information

Video: <https://www.youtube.com/embed/wa514mI7Aw4>

```

In [1]: %configure -f
({'conf': {'spark.yarn.appMasterEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.appMasterEnv.PYSPARK_DRIVER_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.executorEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.pySpark.python': '/opt/anaconda/bin/python',
'spark.pySpark.driver.python': '/opt/anaconda/bin/python'}
})

Current session config: {'conf': {'spark.yarn.appMasterEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.appMasterEnv.PYSPARK_DRIVER_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.executorEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.pySpark.python': '/opt/anaconda/bin/python',
'spark.pySpark.driver.python': '/opt/anaconda/bin/python'}, 'kind': 'pySpark'}

No active sessions.

In [2]: %info

Current session config: {'conf': {'spark.yarn.appMasterEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.appMasterEnv.PYSPARK_DRIVER_PYTHON': '/opt/anaconda/bin/python',
'spark.yarn.executorEnv.PYSPARK_PYTHON': '/opt/anaconda/bin/python',
'spark.pySpark.python': '/opt/anaconda/bin/python',
'spark.pySpark.driver.python': '/opt/anaconda/bin/python'}, 'kind': 'pySpark'}

No active sessions.

In [3]: %start

Starting Spark application

ID      YARN Application ID  Kind  State  Spark UI  Driver log  Current session?
0  application_151378486453_0044  pySpark  idle  LER  LER  ✓

SparkContext available as 'sc'.
%sqlContext available as '%sqlContext'.
2

```

Connect to Hive, Impala and HDFS

Anaconda Enterprise contains numerous example projects, including a Spark/Hadoop project. This project includes the libraries needed to connect to Hive, Impala and HDFS with Python libraries, as well as example notebooks to connect to these services.

Impala

To connect to an Impala cluster you need the address and port to a running Impala Daemon, normally port 21050.

To use Impyla, open a Python Notebook based on the `anaconda50_impyla` environment and run:

```

from impala.dbapi import connect
conn = connect('<Impala Daemon>', port=21050)
cursor = conn.cursor()
cursor.execute('SHOW DATABASES')
cursor.fetchall()

```

Hive

To connect to a Hive cluster you need the address and port to a running Hive Server 2, normally port 10000.

To use PyHive, open a Python notebook based on the `anaconda50_hadoop` environment and run:

```

from pyhive import hive
conn = hive.connect('<Hive Server 2>', port=10000)
cursor = conn.cursor()
cursor.execute('SHOW DATABASES')
cursor.fetchall()

```

HDFS

To connect to an HDFS cluster you need the address and port to the HDFS Namenode, normally port 50070.

To use the `hdfscli` command line, configure the `~/hdfscli.cfg` file:

```
[global]
default.alias = dev

[dev.alias]
url = http://<Namenode>:port
```

Once the library is configured, you can use it to perform actions on HDFS with the command line by starting a terminal based on the `anaconda50_hadoop` environment and executing the `hdfscli` command. For example:

```
$ hdfscli

Welcome to the interactive HDFS python shell.
The HDFS client is available as `CLIENT`.

In [1]: CLIENT.list("/")
Out[1]: ['hbase', 'solr', 'tmp', 'user']
```

Kerberos

To authenticate and connect to Kerberized services, you need the appropriate configuration to execute a `kinit` command in a terminal on the project session.

NOTE: For more information, see the authentication section in [Kerberos Configuration](#).

The Hadoop/Spark example project also includes example notebooks that show how to use the multiple libraries with Kerberos authentication.

Connect to a Remote SAS Server

With Anaconda Enterprise, you can connect to a remote SAS server process using the official `sas_kernel` and `saspy`. This allows you to merge SAS and python/R work-flows in a single interface, and to share your SAS-based work with your colleagues within the Enterprise platform.

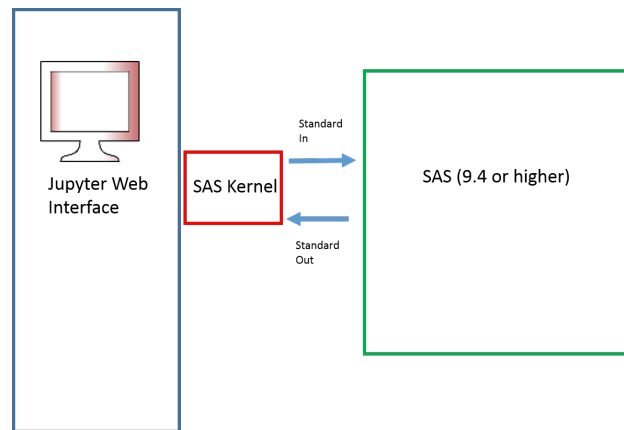
NOTE: SAS sessions are currently available in interactive development mode only, not in deployments.

`sas_kernel` is distributed under the [Apache 2.0](#) Licence, and requires SAS version 9.4, or later. SAS is (c) SAS Institute, Inc..

Anaconda Enterprise and `sas_kernel`

Anaconda connects to a remote SAS server application over a secure SSH connection.

After a connection is established with the provided SAS kernel, SAS commands are sent to the remote server, and results appear in your notebook. All you need to do is ensure a correct configuration. Note that each open notebook starts a new SAS session on the server, which stays alive while the notebook is being used. This may have implications for your SAS license utilization.



Configuration

The file `sascfg_personal.py` in the project root directory provides the configuration for the SAS kernel to run. The connection information is stored in a block like the following.

```
default = {
    'saspath' : '/opt/sas9.4/install/SASHome/SASFoundation/9.4/bin/sas_u8',
    'ssh'      : '/usr/bin/ssh',
    'host'     : 'username@55.55.55.55',
    'options'  : ["-fullstimer"]
}
```

Here, the entry for `'saspath'` must match the exact full path of the SAS binary on the remote system. `'host'` must be a connection string that SSH can understand - note that it includes both login username and IP/hostname parts, and *both* must be correct for a successful connection; there may also be a port number suffixed after a colon, e.g., `:2022`. The values to be entered here will normally be provided by your systems administration.

Establishing a Connection

The SAS server machine must allow SSH with password-less login.

Before starting, you will need a secure key (`.pem`) file installed on the SAS server machine. This will normally be done for you by your organization, but if you have username/password access to the machine, you can make your own. See the section below on Installing a `.pem` File.

Whenever you start a new editing session, you must perform the following steps before creating or running a notebook with a SAS kernel:

- Upload your `.pem` file to your editing session. In a JupyterLab or Jupyter notebook, click the left Files tab, then at the top of the pane click the `upload` button and navigate to the file on your system.
- Open a terminal from the New/Launcher pane. Execute the following commands:

```
chmod 0400 <myfile.pem> ssh-add <myfile.pem> ssh <connection-string> -o StrictHostKeyCheck-
ing=no echo OK
```

where you replace `<connection-string>` with the `host` entry in `sascfg_prsonal.py` and `<myfile.pem>` with the name of your key file.

Now you can the start notebooks with the SAS kernel from the launcher pane, or switch the kernel of any notebook that is already open.

Installing a .pem File

You can create your own .pem file within Anaconda Enterprise 5, or any local machine with openSSH installed. Run the following:

```
ssh-keygen
```

Give your key-file a name (typically with extension .pem), and optional password (it is fine to leave this empty).

The above command will create two files: one with the exact name you specified, which is the one you upload into your SAS kernel editing session; and one with .pub suffixed, which needs to be known to the SAS server. You can view this file in the terminal with the `cat` command:

```
cat <myfile.pem.pub>
```

Next, log into the SSH server using your username/password. Edit the file `~/.ssh/authorized_keys` and append the contents of your .pem.pub file here. The program for editing will vary by system, but `nano` and `vi` are common possibilities.

Migrating AE Notebook Projects

If you have been an Anaconda Enterprise Notebooks user, you are likely to have some 4.x projects that you would like to use in Enterprise v5.

Before you start

If your project contains several notebooks, check that they all are using the same kernel/environment.

NOTE: These instructions are for notebooks that use the same kernel or environment only.

On your AE Notebook server

1. Log onto your Notebooks server.
2. Open a Terminal window and activate the environment that contains your project.
3. Install anaconda project in the environment:

```
conda install anaconda-project=0.6.0
```

If you get a `not found` message, install it from [Anaconda.org](https://anaconda.org/anaconda/anaconda-project):

```
conda install -c anaconda anaconda-project=0.6.0
```

4. Export your environment to a file:

```
`conda env export -n default -f _env.yml`
```

<default> is the name of the environment where the notebook runs.

5. Check the format of the environment to be sure it looks like this, without any warning exclamation points:

```
yaml
channels:
- wakari
- r
- https://conda.anaconda.org/wakari
- defaults
- anaconda-adam
prefix: /projects/anaconda/MigrationExample/envs/default
dependencies:
- _license=1.1=py27_1
- accelerate=2.3.1=np111py27_0
- accelerate_cudalib=2.0=0
- alabaster=0.7.9=py27_0
# ... etc ...
```

NOTE: Check to see if your file contains warning exclamation marks like this example:

```
yaml
name: default
channels:
- wakari
- r
- https://conda.anaconda.org/wakari
- defaults
- anaconda-adam
dependencies:
- _license=1.1=py27_1'
# ...
```

If you see any warning exclamation points, run this script to modify the encoding and remove the warnings:

```
import ruamel_yaml
with open("_env.yml") as env_fd:
    env = ruamel_yaml.load(env_fd)
with open("environment.yml", "w") as env_fd:
    ruamel_yaml.dump(env, env_fd, Dumper=ruamel_yaml.RoundTripDumper)
```

6. Create a Project compatible for Enterprise v5:

```
anaconda-project init
```

7. Remove the platforms, Windows and macOS from the file `anaconda-project.yml` since AEN is linux only. Run:

```
anaconda-project remove-platforms win-64 osx-64
```

Verify using:

```
anaconda-project list-platforms
```

8. Lock the project dependencies:

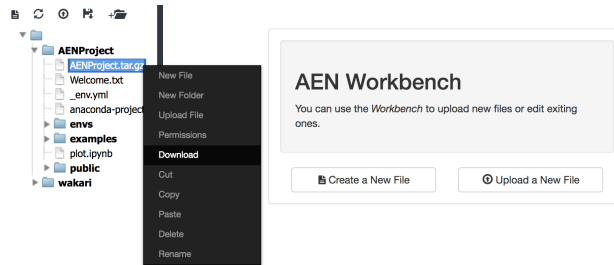
```
anaconda-project lock
```

9. Compress your project:

```
anaconda-project archive WAKARI_PROJECT_NAME.tar.gz
```

Note: If you get a permission denied error for `.indexer.pid`, add `/.indexer.pid` to the `.projectignore` file. To remove all `.git` directories, add `.git` to the `.projectignore` file.

10. In Anaconda Enterprise Notebooks, from your project home page, open the Workbench. In the file list, locate your file `WAKARI_PROJECT_NAME.tar.gz`. To download the project, right-click the file name and select Download.



Upload to Enterprise version 5

Log onto the Enterprise v5 interface and upload `FILENAME.tar.gz`:

On the Projects page, click the Add (+) button, click on “Upload project” and select your file.

Troubleshooting

- Does your project refer to channels in your on-premises repository or other channels in `anaconda.org`? Those channels should be migrated to Enterprise by your system administrator.
- Does your project use packages which are not in `anaconda`? If so, upload those packages to AE v5.
- Does your notebook refer to multiple kernels/environments? Try setting the kernel to just one environment.

1.6 Packages and channels

Creating a package is more advanced than creating an Anaconda Project.

Channels are the specific locations where packages are stored.

Packages are the raw materials you use to create an Anaconda Project. Each package contains an archive of files and information about the package, including its name, version, description, and its dependencies.

After downloading the *Anaconda command line interface (CLI)*, you will *create a channel*. Then you can *upload packages to the channel*.

1.6.1 Before you start

You should be familiar with using a CLI and the `conda` package manager.

Get Anaconda command line interface (CLI)

Download and install the Anaconda CLI so you can create and share channels and packages from your Anaconda Repository using normal conda commands.

NOTE: This is necessary only the first time you use it.

1. Install `anaconda-enterprise-cli` from your Anaconda Repository:

```
conda install -c https://docs.anaconda.example.com:30071/user-guide/getting-
↳started/ anaconda-enterprise-cli
```

NOTE: Replace `anaconda.example.com` with the actual URL to your Anaconda Repository.

2. Configure `anaconda-enterprise-cli`

Add the url of the Anaconda Repository you will be using to the set of sites available to `anaconda-enterprise-cli`:

```
anaconda-enterprise-cli config set sites.example.url https://repo.anaconda.
↳example.com:30089/api
```

NOTE: Replace `repo.anaconda.example.com` with the actual URL to your Anaconda Repository.

Configure it as the default site, the main instance of Anaconda Repository you will be using:

```
anaconda-enterprise-cli config set default_site example
```

3. Check the configuration:

```
anaconda-enterprise-cli config view
```

4. Log into your CLI:

```
anaconda-enterprise-cli login
```

5. Configure the `conda channel_alias` to retrieve packages from your local Repository:

```
conda config --set channel_alias https://repo.anaconda.example.com:30089/conda
```

NOTE: Replace `anaconda.example.com` with the actual URL to your Anaconda Repository.

TIP: The `anaconda-enterprise-cli` package is also available [on anaconda.org](https://anaconda.org/anaconda/anaconda-enterprise-cli).

6. Log into your CLI using the same username and password that you use in the Enterprise web interface:

```
anaconda-enterprise-cli login
Username: <your-username>
Password: <your-password>
```

You can now create and share channels and packages.

Creating a channel

After you have downloaded and set up the *Anaconda CLI* for the first time, create your first channel.

In your terminal, run:

```
anaconda-enterprise-cli channels create <username>/<channelname>
```


NOTE: The channel name `<username>/<channelname>` must not already exist.

NOTE: The slash character (“/”) is allowed in channel names.

This feature is like `labels` on [Anaconda Cloud](#).

Manage your channels

You can get a list of all the channels on the platform with the `channels list` subcommand:

```
anaconda-enterprise-cli channels list
```

Share a channel

Share a channel with a specific user using the `share` command:

```
anaconda-enterprise-cli channels share --user username --level r <username>/
↪<channelname>
```

You can also share a channel with an existing group created by your Administrator:

```
anaconda-enterprise-cli channels share --group GROUPNAME --level r <username>/
↪<channelname>
```

NOTE: Replace `GROUPNAME` with the actual name of your group.

This group has read-only access, level R.

Finally, you can remove a channel share using the following command:

```
anaconda-enterprise-cli channels share --user <username> --remove <channelname>
```

Learn more about channels

Run `anaconda-enterprise-cli channels --help` to see more information about what you can do with channels.

For help with a specific command, enter that command followed by `--help`:

```
anaconda-enterprise-cli channels share --help
```

After you’ve created your channel, you can *upload a package*.

Uploading a package

After you have *downloaded and logged into the CLI* and *created a channel*, you can upload a package to the channel.

To upload a package, in a terminal enter:

```
anaconda-enterprise-cli upload path/to/pkg/notebookname.bz2 --channel <username>/
↪<channelname>
```

NOTE: Replace `path/to/pkg` with the actual path to the package you want to upload, and `<username>/<channelname>` with the actual username and channel name.

Setting a default channel

NOTE: There is no `default_channel` in a fresh install until you create it. You will have to enter the specific channel each time.

If you don't want to enter the `--channel` option with each command, you can set a default channel:

```
anaconda-enterprise-cli config set default_channel <username>/<channelname>
```

To display your current default channel:

```
$ anaconda-enterprise-cli config get default_channel
'<username>/<channelname>'
```

After setting the default channel, upload to your default channel:

```
anaconda-enterprise-cli upload <path/to/pkg/packageName.bz2>
```

NOTE: Replace `<path/to/pkg/packageName.bz2>` with the actual path to the package you want to upload.

Viewing uploaded packages

To see the packages you have uploaded, log onto the Enterprise web interface. Then from the top menu, click **Packages**. You will see a list of all packages you have uploaded.

What's next?

Now that you have uploaded packages to Enterprise, you can make them into *Projects*.

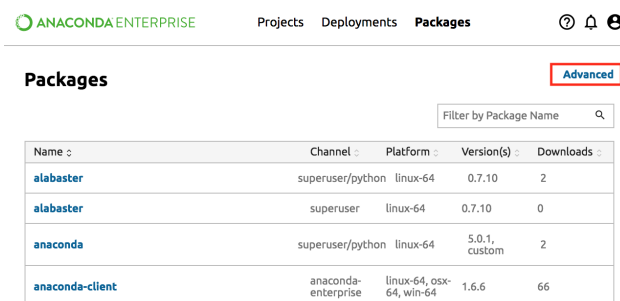
Environments and installers

Your administrator can create custom environments for you and your colleagues to access. These custom environments include specific packages and their dependencies.

Likewise, your administrator can make available to you custom project installers and project templates. These project templates will be available when you create a new project.

Viewing environments

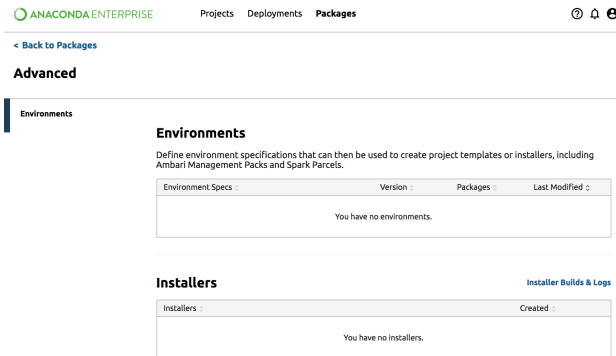
Select the top Packages menu, then click the far right Advanced link.



Name	Channel	Platform	Version(s)	Downloads
alabaster	superuser/python	linux-64	0.7.10	2
alabaster	superuser	linux-64	0.7.10	0
anaconda	superuser/python	linux-64	5.0.1, custom	2
anaconda-client	anaconda-enterprise	linux-64, osx-64, win-64	1.6.6	66

Viewing installers

Select the top Packages menu, then click the far right Advanced link. Scroll to the bottom Installers section.



If you do not see an installer that you expected to see, please contact your administrator.

Parcels

The manifest.json file is available at <https://repo.anaconda.example.com:30089/api/installers/parcels/manifest.json>.
NOTE: Replace repo.anaconda.example.com with your actual hostname.

1.7 Managing deployments

After you've created and deployed an Anaconda Project, Enterprise finds and builds all of the software dependencies—the programs on which the Project depends in order to run—and encapsulates them so they are completely self-contained. You can *deploy notebooks*, *Bokeh applications* and *REST APIs*.

You can then *share your deployment* with others who have accounts on Enterprise, or share your deployment with colleagues who are not on Enterprise by *creating a token*. Colleagues you share a deployment with can run the deployment, but cannot edit it (editing is done in projects, not in deployments.)

Other deployment management items include *viewing the logs* for your deployment or *terminating the deployment* to free up all of its resources.

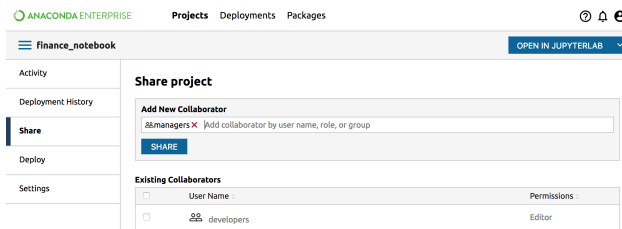
1.7.1 Sharing a deployment

With Anaconda Enterprise, you can select the users, groups or roles with whom you want to share access to a deployment. You can also view or remove access from the users, roles and groups with whom you previously shared access to a deployment.

NOTE: Your administrator creates the roles and groups with whom you can share.

To share a deployment:

1. From the top navigation bar, click **Deployments**.
2. Select the deployment you want to share.
3. In the left navigation bar, click **Share**.



4. In the Share pane, search for and select the users, groups, or roles that you want to share your deployment with.
5. Click the Share button.

The selected users can see your deployment when they log in to Enterprise.

Unshare a deployment

To remove access to a deployment, on the **Deployments** page, under Existing Collaborators, check the box next to the users, groups, or roles you want to remove, and click the Delete button.

1.7.2 Deploying a project

To create an interactive, shareable data science deployment, follow the steps in [Deploying a Project](#). This page covers deploying notebooks, Bokeh applications and REST APIs.

Deploy a notebook

To deploy a notebook, select a project from the Projects list that contains the notebook you want and deploy it the same as above. Select the notebook as the command you would like to launch with.

After it finishes the build, deploy your notebook by clicking Deployments from the top menu, then clicking its name in the Deployments list.

Deploy a Bokeh application

To deploy a Bokeh app, select a project containing a Bokeh launch script from the Projects list and deploy it the same as above.

After the build finishes, open your newly deployed Bokeh app by clicking Deployments from the top menu and clicking its name in the Deployments list.

Deploy a REST API

To deploy a REST API, select a project containing a REST API from the Projects list and deploy it the same as above. After it finishes the build, in the Deployments list, open your newly deployed REST API by clicking Deployments from the top menu and clicking its name in the Deployments list.

See also the tutorial, *Deploying a REST API using Anaconda Enterprise*.

1.7.3 Interact with a Bokeh App

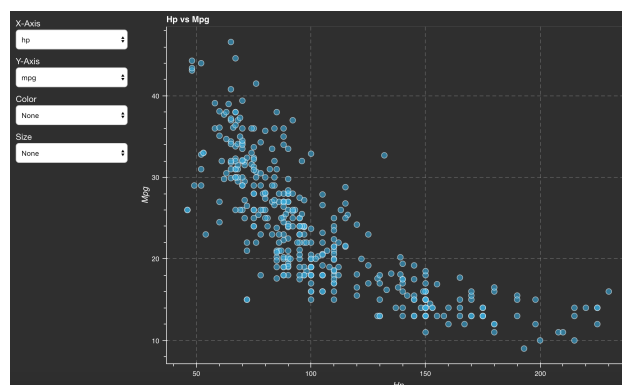
After *deploying a project*, you are ready to interact with your Deployment. The following example is a Bokeh app for data clustering.

1. From the Projects list, click the Sample Projects link.
2. Find the project named “data_clustering” and click the “Save to My Projects” link.
3. Return to the Projects page, click the project “data_clustering” to open the Project Detail page.
4. Click the left Deploy tab, then click the Deploy button to deploy the project. Click the Done button when finished.

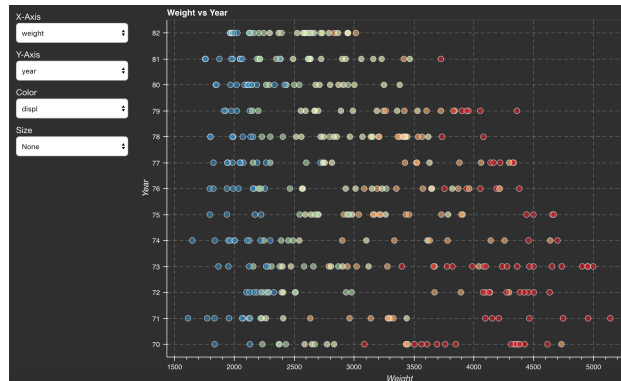
The screenshot shows the 'Deploy project' page in the Anaconda Enterprise interface. The top navigation bar includes 'Projects', 'Deployments', and 'Packages'. The left sidebar has tabs for 'Activity', 'Deployment History', 'Share', 'Deploy' (which is active), and 'Settings'. The main content area is titled 'Deploy project' and 'Deploy Project: data_crossfilter'. It contains form fields for 'App Name' (data_crossfilter), 'Revision' (0.1.0 (an hour ago)), and 'Command' (default). A 'DEPLOY' button is at the bottom. A note states: 'You can edit who this deployment is shared with later.'

5. Now you can interact with the data clustering Bokeh App by selecting different datasets and algorithms. You can also change the number of clusters and the number of data samples.

Mini Batch K-Means Algorithm:



Mean-shift Algorithm:

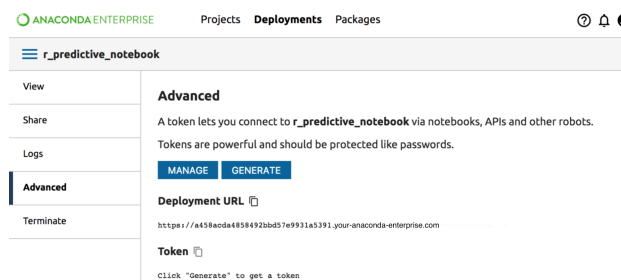


1.7.4 Manage deployment tokens

Tokens allow you to control access to a deployment you want to share. You can use tokens on any deployed project in Anaconda Enterprise.

To manage deployment tokens for a deployment:

1. From the top navigation bar, click **Deployments**.
2. In the Deployments list, click the name of the deployment you want to share by using a token.
3. On the left side-menu, click **Advanced**.



4. Click the **Generate** button to generate a new token. Copy it to the clipboard with the Copy icon, or by copying it with mouse or keyboard shortcuts like any other text. Use this token to connect to the deployment from Notebooks, APIs and other running code.

Revoke tokens


To revoke a token, click the Manage button. Scroll down the list to the token you want to revoke. In this example, the token we are looking for is `https://2c9d36023a0a427f8efdc0b18bf38926.your-server-name.com/`

In the Action column, click the Revoke Grant button.




Applications

Application	Available Permissions	Granted Permissions	Granted Personal Info	Additional Grants	Action
app_client_86f7a8f65af74d3084359dcbe839d2aa	Offline access	Full Access	Full Access		
space_client_530c24427644dc9b9e009b644d93c05	Offline access	Full Access	Full Access		
my client	Offline access	Full Access	Full Access		
app_client_bea607710ead4ab4ab42bd0c4c790108	Offline access	Full Access	Full Access		
app_client_2cf936023a0a4278fcd0cb18f38926	Offline access	Full Access	Full Access	Offline Token	Revoke Grant


Anaconda Enterprise keeps an extensive log of everything that occurs to your deployments. To view a deployment's log in Anaconda Enterprise:

- 
ANACONDA ENTERPRISE

[Projects](#)
[Deployments](#)
[Packages](#)

Deployments



Deployment Name	Owner	Created
Hello Anaconda Enterprise	superuser	a few seconds ago
weather_statistics	anaconda-enterprise	February 19, 2018, 2:32 pm

- ## 1.7. Managing deployments 79

If you want a copy of the log, highlight the log text and copy it with normal keyboard or mouse shortcuts such as Control-C or Command-C. You can paste the text of the log into any text editor with normal keyboard or mouse shortcuts such as Control-V or Command-V.

1.7.6 Terminating a deployment

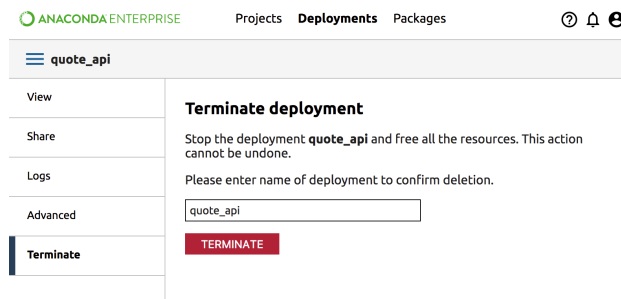
Terminating a deployment:

- Stops the deployment from running.
- Removes the deployment from the deployment server.
- Makes the deployment unavailable to you and any users you had shared it with.
- Frees all of its resources.

Terminating a deployment does not affect the original project from which the deployment was created. It affects only the deployment.

To terminate a deployment:

1. Go to the **Deployments** list and select your deployment.
2. From the left navigation bar, click **Terminate**.
3. Confirm that you want to stop the deployment in the Terminate pane by typing the name of the deployment to confirm, or copy and paste the name of the deployment shown at the top of the pane.



NOTE: If you have shared the deployment with a collaborator you will also be terminating their deployment.

4. Click the Terminate button.

The deployment stops, and its resources are returned to the system. You will be returned to the list of deployments.

1.8 Tutorials

These tutorials are provided to give you a deep dive into Anaconda Enterprise.

1.8.1 Before you begin

You should have already run through the introductory materials:

- *Twenty-minute getting started tutorial*
- *Cheat sheet*
- *Concepts*

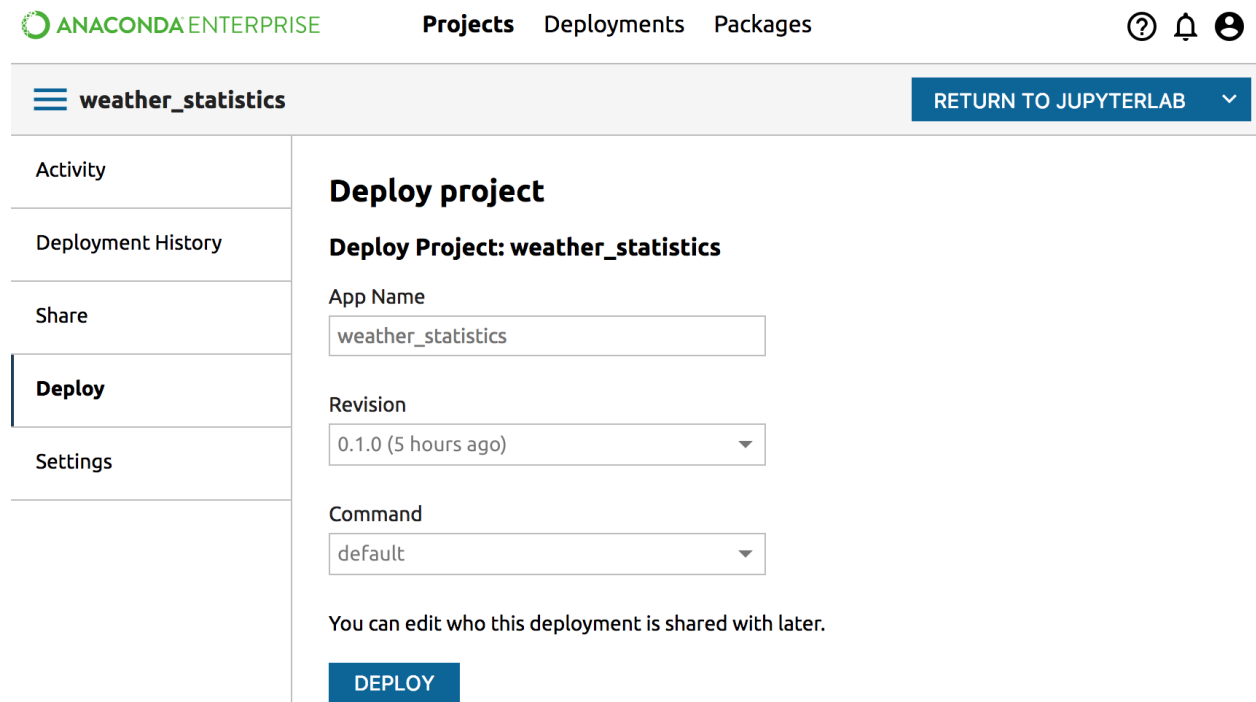
1.8.2 Basic tutorials

Deploying a Project as an interactive application

This tutorial walks you through deploying an Anaconda Project and sharing the deployment with others so they can interact with it.

In this tutorial you will use an example project included with Anaconda Enterprise.

1. In Anaconda Enterprise, click the **Projects** tab.
2. At the top right, click the Sample Projects link, then find the sample project “weather_statistics” and click the Save to My Projects link.
3. After it’s saved, click the View Project link. Then in the left navigation bar, click the Deploy tab.
4. In the Deploy Project pane, review the default deployment options:



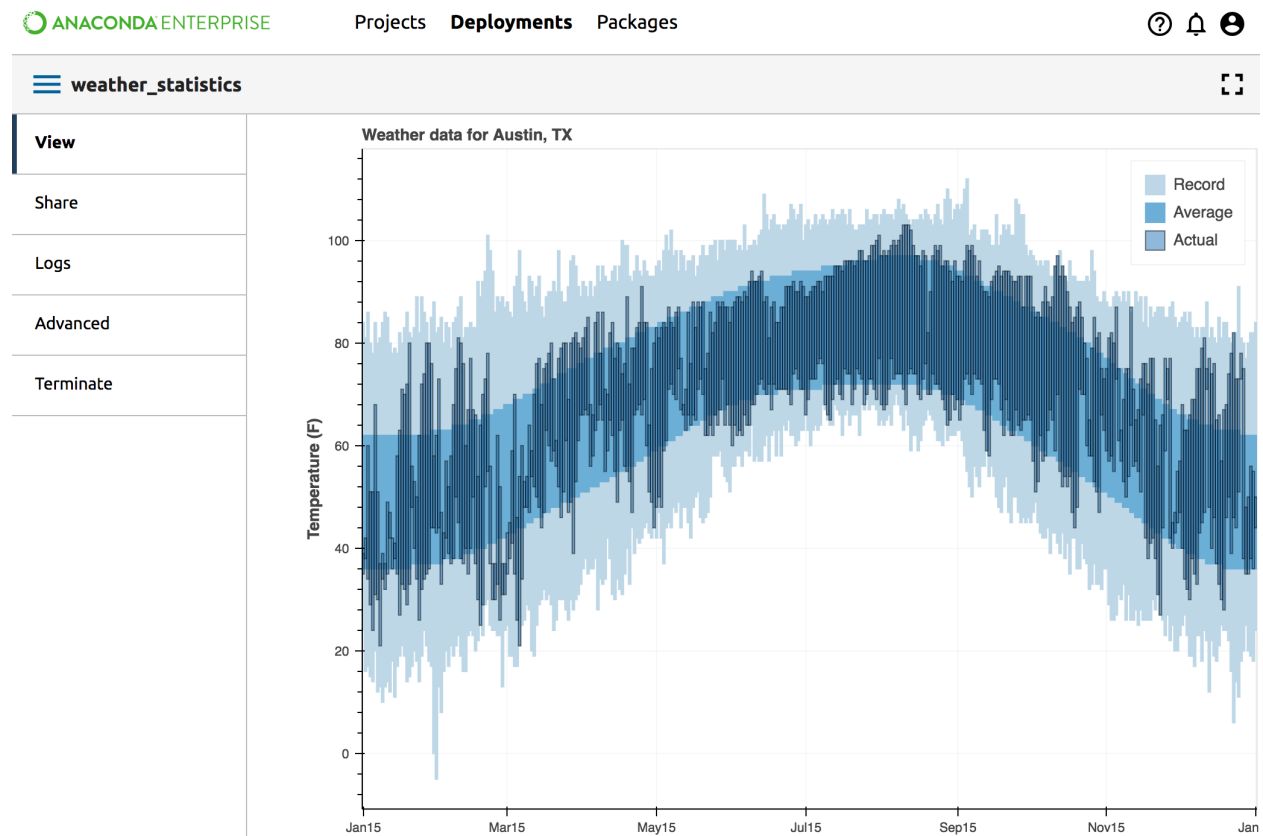
The screenshot shows the Anaconda Enterprise web interface. At the top, there's a navigation bar with the Anaconda logo, 'ANACONDA ENTERPRISE', and tabs for 'Projects', 'Deployments', and 'Packages'. On the right of the navigation bar are icons for help, notifications, and user profile. Below the navigation bar, a header bar shows a hamburger menu icon, the project name 'weather_statistics', and a 'RETURN TO JUPYTERLAB' button with a dropdown arrow. On the left is a sidebar with navigation links: 'Activity', 'Deployment History', 'Share', 'Deploy' (which is highlighted with a blue border), and 'Settings'. The main content area is titled 'Deploy project' and 'Deploy Project: weather_statistics'. It contains three form fields: 'App Name' with the value 'weather_statistics', 'Revision' with a dropdown showing '0.1.0 (5 hours ago)', and 'Command' with a dropdown showing 'default'. Below these fields is a text prompt: 'You can edit who this deployment is shared with later.' At the bottom of the main content area is a blue 'DEPLOY' button.

5. At the bottom of the pane, click the Deploy button.

Your project is deployed. This process may take a minute.

To share the deployment:

1. In the top navigation bar, click **Deployments**.
2. Click the Deployment Name column heading to sort the list alphabetically.
3. Click the deployment name “weather_statistics,” which is the deployment that you just built when you deployed the weather_statistics project:



4. In the left navigation bar, click **Share**. The Share pane will appear.
5. In the Share pane, search for and select the users, groups, or roles that you want to share your deployment with:
6. Click the Share button.

The selected users can now see your deployment when they log in to Enterprise.

ANACONDA ENTERPRISE Projects Deployments Packages ? 🔔

finance_notebook OPEN IN JUPYTERLAB

Activity
Deployment History
Share
Deploy
Settings

Share project

Add New Collaborator
 Add collaborator by user name, role, or group
SHARE

Existing Collaborators

<input type="checkbox"/>	User Name	Permissions
<input type="checkbox"/>	developers	Editor

Deploying a Jupyter Notebook project

This tutorial uses the sample project, Markowitz Notebook, included with Anaconda Enterprise. This project is an investment portfolio analysis that demonstrates a Jupyter Notebook running both R language and Python.

The notebook also demonstrates the pandas and Bokeh packages for analysis and visualization. It outputs four different interactive graphs. A step-by-step description of the analysis is available inside the notebook.

To copy the sample project into your list of projects, follow the instructions on [viewing and saving sample projects](#) to get a copy of markowitz_notebook.

To deploy the Markowitz Notebook, follow the [deployment instructions](#).

After your project has deployed, you can select it and see the running application - in this case a notebook.

A deployed notebook can be run and interacted with, but viewers cannot edit.

In the Jupyter Notebook top navigation bar, in the **Cell** menu, select Run All to run all of the included code blocks and generate the output:

ANACONDA ENTERPRISE Projects **Deployments** Packages ? 🔔

markowitz_notebook

View
Share
Logs
Advanced
Terminate

jupyter markowitz (read only)

File Edit View **Cell** Kernel Help

Run Cells
Run Cells and Select Below
Run All
Run All Above
Run All Below

Current Outputs
All Output

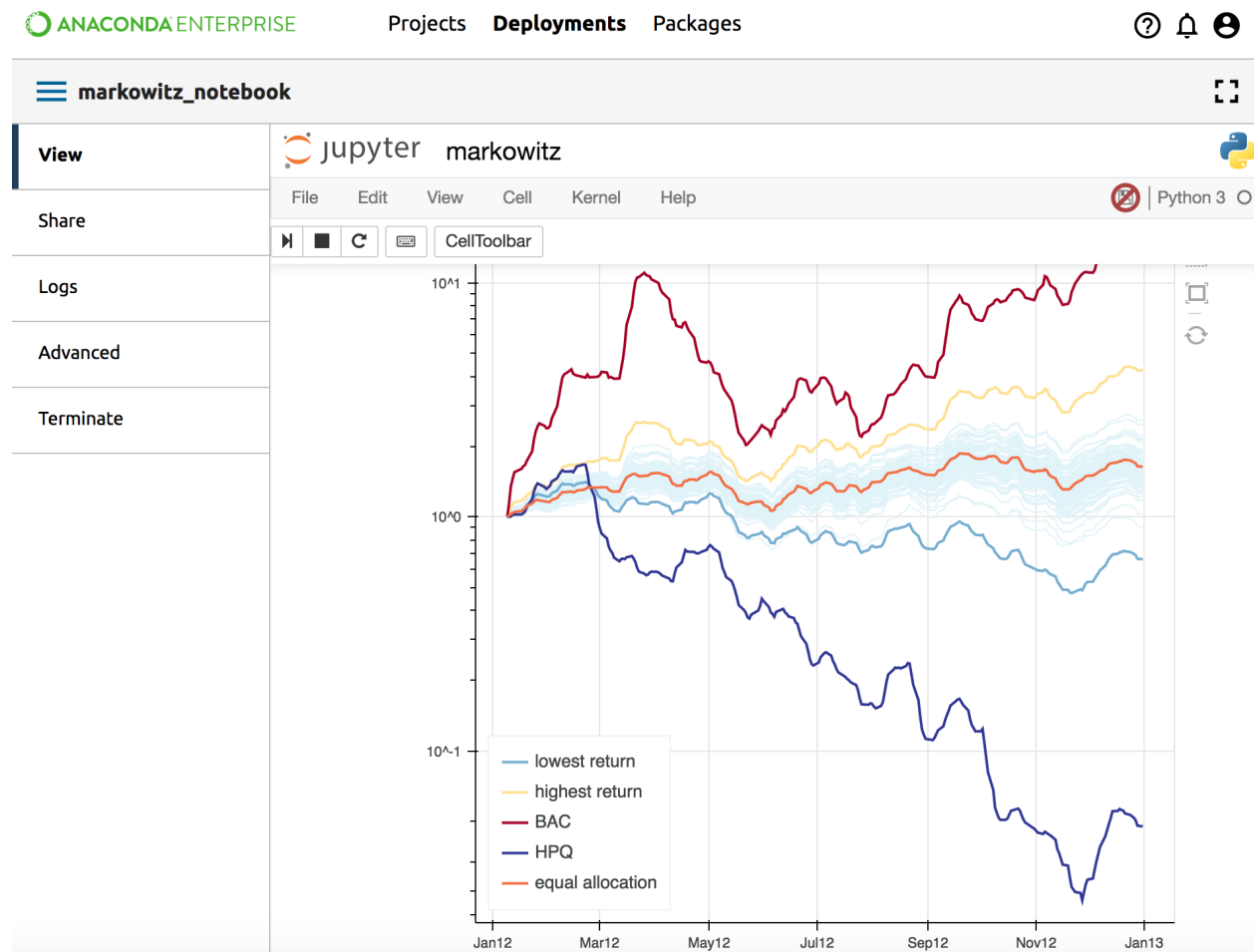
Markowitz Notebook

This notebook demonstrates how to design using **Pandas** and **Bokeh**.

The original data is obtained in R's binary data format, so we're using R itself and the **ropy** package to import it into Python. If you don't have either of those packages, just make sure that use_R is False; the notebook will load a pre-converted version of the data instead.

```
In [1]: use_R = False
save_Data = False
```

Scroll down to see the generated plots and graphs:



Interact with the deployment to change the data being sampled. Interactive selections in this notebook include pan, box zoom, resize and reset. The plots and graphs immediately adjust as you interact with them.

1.8.3 Advanced tutorials

Using Anaconda Project CLI

This tutorial walks you through creating a data science project using Anaconda Enterprise with the [Anaconda Project](#) command line interface (CLI).

After completing this tutorial, you will be able to:

- *Download and run a sample project*
- *Create and initialize a data science project*
- *Customize a data science project*
- *Run a data science project on your local machine*
- *Archive and upload a data science project*

Before you start

- Windows: From the Start Menu, open a new Anaconda Prompt.
- macOS and Linux: Open a new Terminal window.

Check that you have the right version of the Anaconda Distribution by running the command `conda list anaconda`.

If the output shows a version older than 4.3.1, update Anaconda by running `conda update anaconda`.

Downloading and running a sample project

To get a sample project see the page [Viewing sample projects](#).

When you click a sample project's Download link, an archive file `*.tar.bz2` is downloaded to your machine.

From your Anaconda Prompt, navigate to the folder where you download the file. For example on default Windows:

```
> cd Downloads
```

Expand the file using *anaconda-project*:

```
> anaconda-project unarchive <filename>
```

You will see a new folder that has the same name as the archive file you downloaded.

Finally, go into that directory, and run the project:

```
> cd <new_project> # directory name dependent on sample project downloaded
> anaconda-project run
```

Creating and initializing a data science project

This procedure creates a data science project definition file named `anaconda-project.yml`. This file specifies scripts, notebooks, project dependencies and other project-specific information.

The project directory also has a `.projectignore` file that contains an optional list of files to ignore when the project is uploaded.

To create a data science project:

1. In a terminal window, create a new directory for all of your data science project files:

```
mkdir data-science-project
```

2. Navigate to your newly created directory:

```
cd data-science-project
```

3. From within this directory, initialize your data science project by running:

```
anaconda-project init
```

Customizing a data science project

By default, a data science project specifies `anaconda` in its runtime environment, but you can also define a customized list of conda packages and specific versions for your project to use.

To customize your data science project: use the `anaconda-project` command or any text editor to open the `anaconda-project.yml` file describing your project and edit it directly.

EXAMPLE: To add a Jupyter Notebook to the project that depends on `numpy`, `pandas` and `bokeh`, run the following commands:

```
anaconda-project add-command --type notebook default data-science-notebook.ipynb
anaconda-project add-packages numpy pandas bokeh
```

The resulting data science project definition file, `anaconda-project.yml`, contains the following information:

```
name: data-science-notebook

commands:
  default:
    notebook: data-science-notebook.ipynb

packages:
- python=3.5
- jupyter
- numpy
- pandas=0.19.2
- bokeh=0.12.4

env_specs:
  default:
    channels: []
    packages: []
```

You can generate similar data science project definition files for projects that include scripts/models with REST APIs, as shown here:

```
name: quote_api
description: A simple script with an API.

commands:
  default:
    unix: python ${PROJECT_DIR}/quote.py
    windows: python %PROJECT_DIR%\quote.py
    supports_http_options: true

packages:
- six>=1.4.0
- gunicorn==19.1.0
- pip:
  - python-mimeparse
  - falcon==1.0.0

env_specs:
  default:
    packages: []
    channels: []
```

You can also generate data science project definition files for an interactive Bokeh application, as shown here:

```

name: stocks_app
description: An example Bokeh application.

commands:
  default:
    bokeh_app: .

downloads:
  QUANTQUOTE: http://quantquote.com/files/quantquote_daily_sp500_83986.zip

packages:
  - bokeh=0.12.4
  - pandas=0.19.2

env_specs:
  default:
    channels: []
    packages: []

```

Running a data science project on your local machine

To make sure your project is defined correctly, test and run your project locally before uploading it.

To test the project on your local machine, run:

```
anaconda-project run
```

Archive and upload a data science project

After testing the project locally, you can create an archive, upload it to Anaconda Enterprise, and continue working on it there.

To create an archive, run:

```
anaconda-project archive ARCHIVE_FILENAME
```

NOTE: Replace ARCHIVE_FILENAME with the actual name of the archive file.

Then continue to the page on [Uploading a project](#).

Deploying a REST API using Anaconda Enterprise

With Anaconda Enterprise, you can create functions or models that can be easily deployed so other data scientists can use them without having to add a web stack.

Rather than giving other data scientists your model, you can give them an interface to the model, which you can then update, improve and redeploy.

You should know how to deploy a project and how to use deployment tokens to share the deployment.

This example is based on the Scotch whisky recommendation engine and can also be found at the `ae_scotches_api` folder within the Anaconda Enterprise distribution.

Before you start

Create a new project and start editing the project.

Make a scotch-recommending API

The original scotch classification data is from the [Classification of whiskies](#) page.

The data have been made public as a Python pickled dataframe.

1. In a terminal window, enter these commands to add the data to download to the project:

```
anaconda-project add-download FEATURES https://ibm.box.com/shared/static/
↪2vntdqbozf9lzmukkeoql1fi2pcb00j1.dataframe
anaconda-project add-download SIM https://ibm.box.com/shared/static/
↪54kzs5zquv0vjycemjckjbh0n00e7m5t.dataframe
```

Or add the following to the file `anaconda-project.yml`:

```
downloads:
  FEATURES:
    url: 'https://ibm.box.com/shared/static/2vntdqbozf9lzmukkeoql1fi2pcb00j1.
↪dataframe'
  SIM:
    url: 'https://ibm.box.com/shared/static/54kzs5zquv0vjycemjckjbh0n00e7m5t.
↪dataframe'
```

And in a terminal window run:

```
anaconda-project prepare
```

2. Add the packages that your REST API will depend on:

```
anaconda-project add-packages pandas
anaconda-project add-packages -c anaconda-enterprise anaconda-enterprise-web-
↪publisher
```

3. Open a Jupyter Notebook, rename it to `Scotches` and load the data.

```
import os
import pandas as pd
import pickle

features_pickle = os.getenv('FEATURES', '2vntdqbozf9lzmukkeoql1fi2pcb00j1.
↪dataframe') # These ENV VARS were defined on add-download
sim_pickle = os.getenv('SIM', '54kzs5zquv0vjycemjckjbh0n00e7m5t.dataframe')

with open(features_pickle, 'rb') as pkl:
    features_df = pickle.loads(pkl.read())

with open(sim_pickle, 'rb') as pkl:
    sim_df = pickle.loads(pkl.read())
```

4. Add this code to be able to handle HTTP requests.

Define a global REQUEST JSON string that will be replaced on each invocation of the API.


```
import json
REQUEST = json.dumps({
    'path' : {},
    'args' : {},
    'body': {}
})
```

5. Import the Anaconda Enterprise publish function.

```
from anaconda_enterprise import publish
```

6. Create a function that will be available from the `/scotches` URL when deployed.

```
@publish()
def scotches():
    names = sim_df.columns.tolist()
    return json.dumps(dict(names=names), indent=4)
```

7. To be able to run, add this to the file `anaconda-project.yml`:

```
commands:
  api:
    rest_api: Scotches.ipynb
    supports_http_options: true
    default: true
  default:
    notebook: Scotches.ipynb
    description: Run the notebook for local development

variables:
  KG_FORCE_KERNEL_NAME: python3
```

8. Then you can execute:

```
anaconda-project run api
```

9. Now if you visit `http://localhost:8888/scotches` from within the editing session you will see the list of scotches.

From an editing session terminal, execute:

```
curl localhost:8888/scotches
```

10. To deploy the project as an API, go to the Project menu and select the Deploy project tab. From the command drop-down, select the `api` command.

This deploys the notebook as an API which you can then query.

11. To query externally, create a token and find the url to the running project. You can do both from the deployment view of your project.

Go to the deployments menu and click on your deployed project. On the Advanced tab, find the url for the running project. Click the Generate button. This generates a long JWT token. A JWT token is a JSON Web Token, and JSON is JavaScript Object Notation.

Example using `curl`:

```
export TOKEN="<generated-token-goes-here>" # save long string of text in variable
curl -L -H "Authorization: Bearer $TOKEN" <url-of-project>
```

The `-L` option tells curl to follow redirects. The `-H` adds a header. In this case `-H` adds the token required to authorize the client to visit that URL.

12. You can also use [Postman](#). Remember to include the authorization header.

13. You can specify the HTTP verb you want use to access the data:

```
@publish(methods=['POST'])
def scotch_details(name):
    features = features_df.loc[name]
    return '{"features":%s}' % features.to_json()
```

If you do `curl -X POST -d name=Talisker 127.0.0.1:8889/scotch_details` or `POST` to the platform, you can use the REST API to access data about the scotch named Talisker.

If you deploy the project as described above you can add the `-X POST` option to `curl` to access that function.

1.8.4 After you finish

Try the example projects in Anaconda Enterprise. Each is designed to show you specific features and capabilities. After logging in, click the top right “Sample Projects” link. Select the project you want, and click “Save to My Projects.” Launch an editing session to view its contents, or deploy or use as desired.

1.9 Troubleshooting

1.9.1 New classic Jupyter Notebook not found on IE11

On Internet Explorer 11, creating a new classic Jupyter Notebook may produce the error “404: Not Found”. This is an artifact of the way that Internet Explorer 11 locates files.

Solution

If you see this error, return to the Projects tab, click the name of the project, click the “Return to classic Notebook” button, and click the name of the notebook. This refreshes the file list and allows IE11 to find the file.

You can perform the following user management operations through the Anaconda Platform web UI.

This guide describes how to install, configure and administer an Anaconda Enterprise 5 deployment. IT Administrators use this guide to manage resources, availability, users, backup and recovery configurations.

The current version number is listed in the [Release notes](#). For documentation on previous versions of AE5, please choose from the version selector on the bottom right of the page.

2.1 Installation and configuration

This is a step-by-step guide to installing Anaconda Enterprise IT administrators, system integrators and implementation engineers.

To upgrade Anaconda Enterprise software, see [Upgrading Anaconda Enterprise](#).

If you have any questions about these instructions or you encounter any issues while installing or upgrading AE5, please contact your Priority Support team.

2.1.1 Installation Requirements

This section describes the installation requirements for Anaconda Enterprise.

Anaconda Enterprise can be installed on one to four nodes during the initial installation. After the initial installation, you can add or remove nodes from the Anaconda Enterprise cluster any time.

A rule of thumb for each project session or deployment is 1 CPU, 1 GB of RAM and 5 GB of disk space.

For more information about sizing for a particular component, see the following minimum requirements:

- [Hardware requirements](#)
- [Operating system requirements](#)
- [Browser requirements](#)

- *Network requirements*
- *Kernel module requirements*
- *Security requirements*
- *IOPS requirements (cloud installation)*
- *Verification of system requirements*

Hardware requirements

Minimum and recommended specs of the master and worker nodes:

Master node	Minimum	Recommended
CPU (cores)	16	16
RAM (GB)	32	32
Disk space in /opt/anaconda (GB)	100	500*
Disk space in /var/lib/gravity (GB)	100	100
Disk space in /tmp or \$TMPDIR (GB)	30	30

Worker nodes	Minimum	Recommended
CPU (cores)	4	16
RAM (GB)	16	32
Disk space in /var/lib/gravity (GB)	100	100
Disk space in /tmp or \$TMPDIR (GB)	30	30

*Note that the recommended disk space in `/opt/anaconda` includes project and package storage (including mirrored packages).

*Note that currently `/opt/anaconda` **must** be a supported filesystem such as `ext4` or `xfs` and **cannot** be an NFS mountpoint. Subdirectories of `/opt/anaconda` may be mounted through NFS.

*If you are installing Anaconda Enterprise on an `xfs` filesystem, then it needs to support `d_type` to work properly. XFS filesystems do not support `d_type` if they have been formatted with the `-nftype=0` option. Before installing Anaconda Enterprise, recreate the filesystem using the correct parameter for XFS, or use another supported filesystem.

Operating system requirements

Linux versions:

- RHEL/CentOS 7.2 through 7.4
- Ubuntu 16.04
- Hosted vSphere such as Rackspace or OVH
- SUSE 12 SP2 - 12 SP3

NOTE: On SUSE set `DefaultTasksMax=infinity` in `/etc/systemd/system.conf`.

Browser requirements

- Edge 14+
- Internet Explorer 11+

- Chrome 39+
- Firefox 49+
- Safari 10+

The minimum browser screen size for using the platform is 800 pixels wide and 600 pixels high.

Network requirements

The following network ports for Anaconda Enterprise are externally accessible:

- 80 - Deployed apps
- 443 - Deployed apps
- 8080 - Anaconda Enterprise
- 30071 - Documentation service
- 30080 - Authentication service
- 30081 - Deployment service
- 30082 - Authentication API service
- 30085 - Git Proxy service
- 30086 - Storage service
- 30087 - Object storage service
- 30088 - Git storage service
- 30089 - Repository service
- 30090 - UI service
- 30091 - Authentication escrow service
- 30095 - Spaces service
- 32009 - Operations center
- 30000 through 32767 - User-deployed applications
- 61009 - Install wizard (only used during cluster installation)

The following network ports for Anaconda Enterprise must be open *internally*, between cluster nodes:

- 53 - Internal cluster DNS
- 2379 - Etcd server communication
- 2380 - Etcd server communication
- 3008 - Internal Anaconda Enterprise service
- 3009 - Internal Anaconda Enterprise service
- 3010 - Internal Anaconda Enterprise service
- 3011 - Internal Anaconda Enterprise service
- 3012 - Internal RPC agent
- 3022 - Teleport internal SSH control panel
- 3023 - Teleport internal SSH control panel

- 3024 - Teleport internal SSH control panel
- 3025 - Teleport internal SSH control panel
- 3080 - Teleport web UI
- 4001 - Etcd server communication
- 5000 - Docker registry
- 6443 - Kubernetes API server
- 7001 - Etcd server communication
- 7373 - Peer-to-peer health check
- 7496 - Peer-to-peer health check
- 8472 - VXLAN (Overlay network)
- 10248 - Kubernetes components
- 10249 - Kubernetes components
- 10250 - Kubernetes components
- 10255 - Kubernetes components

The following domains are required for package mirroring:

- repo.continuum.io
- anaconda.org
- conda.anaconda.org
- binstar-cio-packages-prod.s3.amazonaws.com

IPv4 forwarding on servers is required for internal load balancing and must be turned on. Anaconda Enterprise performs pre-flight checks and only allows installation on nodes that have the required kernel modules and other correct configuration.

To enable IPv4 forwarding, run:

```
sysctl -w net.ipv4.ip_forward=1
```

To persist this setting on boot, run:

```
echo -e "# Enable IPv4 forwarding\nnet.ipv4.ip_forward=1" >> /etc/sysctl.d/99-ipv4_
↪forward.conf
```

Kernel module requirements

The following kernel modules are required for Kubernetes cluster to properly function.

br_netfilter module

The bridge netfilter kernel module is required for Kubernetes iptables-based proxy to work correctly.

The bridge kernel module commands are different for different versions of CentOS.

To find your operating system version run `cat /etc/*release*` or `lsb-release -a`.

On RHEL/CentOS 7.2 the bridge netfilter module name is `bridge` and on other operating systems and other versions of CentOS the module name is `br_netfilter`.

To check if the module is loaded run:

```
# For RHEL/CentOS 7.2
lsmod | grep bridge

# For all other supported platforms
lsmod | grep br_netfilter
```

If the above commands did not produce any result, then the module is not loaded. Run the following command to load the module:

```
# For RHEL/CentOS 7.2
modprobe bridge

# For all other supported platforms
modprobe br_netfilter
```

Now run:

```
sysctl -w net.bridge.bridge-nf-call-iptables=1
sysctl -w net.bridge.bridge-nf-call-ip6tables=1
```

To persist this setting on boot, run:

```
echo "# Enable bridge module" >> /etc/sysctl.d/99-bridge.conf
echo "net.bridge.bridge-nf-call-iptables=1" >> /etc/sysctl.d/99-bridge.conf
echo "net.bridge.bridge-nf-call-ip6tables=1" >> /etc/sysctl.d/99-bridge.conf
```

overlay module

The overlay kernel module is required to use overlay or overlay2 Docker storage driver.

To check that overlay module is loaded:

```
lsmod | grep overlay
```

If the above command did not produce any result, then the module is not loaded. Use the following command to load the module:

```
modprobe overlay
```

ebtables module

The ebtables kernel module is required to allow a service to communicate back to itself via internal load balancing when necessary.

To check that ebtables module is loaded:

```
lsmod | grep ebtables
```

If the above command did not produce any result, then the module is not loaded. Use the following command to load the module:

```
modprobe ebtables
```

NOTE: During installation, the Anaconda Enterprise installer alerts you if any of these modules are not loaded.

NOTE: If your system does not load modules at boot, add the following to ensure they are loaded in case the machine gets rebooted:

```
sudo bash -c "echo 'overlay' > /etc/modules-load.d/overlay.conf"
sudo bash -c "echo 'br_netfilter' > /etc/modules-load.d/netfilter.conf"
sudo bash -c "echo 'ebtables' > /etc/modules-load.d/ebtables.conf"
```

Mount Settings

Many linux distributions include the kernel setting `fs.may_detach_mounts = 0`. This can cause conflicts with the docker daemon and Kubernetes will show pods as stuck in the terminating state if docker is unable to clean up one of the underlying containers.

If the installed kernel exposes the option `fs.may_detach_mounts`, we recommend always setting this value to 1:

```
sudo sysctl -w fs.may_detach_mounts=1
sudo bash -c "echo 'fs.may_detach_mounts = 1' >> /etc/sysctl.d/10-may_detach_mounts.
↪conf"
```

Security requirements

For CentOS and RHEL:

- [Disable SELinux](#) on all of the cluster nodes.
- Various tools may be used to configure firewalls and open required ports, including `iptables`, `firewall-cmd`, `susefirewall2`, and others.

Make sure that the firewall is permanently set to keep the required ports open, and will save these settings across reboots. Then restart the firewall to load these settings immediately.

- Sudo access.

IOPS requirements (cloud installation)

Master and worker nodes require a minimum IOPS of 3000 IOPS. Fewer IOPS than 3000 will fail.

Verification of system requirements

Storage requirements

To check your available disk space, use the built-in Linux `df` utility with the `-h` parameter for human readable format:

```
df -h /var/lib/gravity
df -h /opt/anaconda
df -h /tmp
```

(continues on next page)

(continued from previous page)

```
# or  
df -h $TMPDIR
```

Memory requirements

To show the free memory size in GB, run:

```
free -g
```

CPU requirements

To check the number of cores, run:

```
nproc
```

2.1.2 Installation

Before you start

Verify that your system meets all *Anaconda Enterprise system requirements*.

Configure DNS records

Configure your A record in DNS for the master node.

```
anaconda.example.com
```

```
*.anaconda.example.com
```

NOTE: Replace `anaconda.example.com` with the actual domain name you will use for your Anaconda Enterprise installation.

Create temporary directory (optional)

Approximately 30 GB of available free space on each node is required for the Anaconda Enterprise installer to temporarily decompresses files to the `/tmp` directory during the installation process.

If adequate free space is not available in the `/tmp` directory, you can specify the location of the temporary directory to be used during installation by setting the `TMPDIR` environment variable to a different location.

Example:

```
sudo TMPDIR=/tmp2 ./gravity install
```

NOTE: When using `sudo` to install, the temporary directory must be set explicitly in the command line to preserve `TMPDIR`.

The master node and each worker node all require a temporary directory of the same size and should each use the `TMPDIR` variable as needed.

Or, you can install as root.

Docker

The machines on which you are installing must not already have Docker installed. Remove Docker or use a clean machine.

Installing on Amazon EC2 and other cloud providers

NOTE: etcd is very [sensitive to disk latency](#).

Use a configuration where etcd is located into its own EBS volume. This terraform configuration file has the recommended drive configuration: <https://gist.github.com/danielfrg/8db7f72d79694da60edb4f5b8d17e1e4#file-gravity-tf-L35>

To use that terraform configuration file you need to change the keyname and the AMI to the target you want to test.

- CentOS 7.2 (ami-692df57f)
- RHEL 7.2 (ami-9e2f0988)

If you don't want to use the terraform configuration file, we recommend that you install Anaconda Enterprise on machines (master AND worker) with IOPS of a minimum of 3000 IOPS on Amazon EC2.

NOTE: Fewer IOPS than 3000 will fail.

For Instance Type we recommend the *m4.xlarge* instance types or larger.

Custom user ID (UID) for service account

By default, Anaconda Enterprise will install using a service account with the user ID (UID) 1000. You can change the UID of the service account by using the `--service-uid` option or the `GRAVITY_SERVICE_USER` environment variable at installation time. Refer to the installation commands below for example usage.

\$TMPDIR needs to be writable by User ID:1000

Create a new directory and set TMPDIR before installing. User 1000 (or the desired UID for the service account) needs to be able to write to this directory. This means they can read, write and execute on the \$TMPDIR.

For example, to give write access to UID 1000, run:

```
sudo chown 1000 -R $TMPDIR
```

Installation Instructions

There are two methods to install Anaconda Enterprise:

- Browser installation (must be on the same network as the target machines)
- Unattended command-line interface (CLI) installation

With both methods, you can create any number of nodes from one to four nodes.

You can also add or remove nodes at any time after installation.

Option 1. Browser Installation

On the master node, create a new directory for the installer. Navigate to the new directory, then download and decompress the installer:

```
mkdir anaconda-enterprise
cd anaconda-enterprise
curl -O <location_of_installer>.tar.gz
tar xvzf anaconda-enterprise-<installer_version>.tar.gz
cd anaconda-enterprise-<installer_version>
```

NOTE: The tarball file is 7 GB, so it may take a while to download.

On the master node, run the installer as sudo or root user:

```
sudo ./gravity wizard
```

NOTE:: By default, Anaconda Enterprise will install using a service account with the user ID (UID) 1000. You can change the UID of the service account by using the `--service-uid` option or the `GRAVITY_SERVICE_USER` environment variable. For example, to use UID 1001, you can use the command `sudo ./gravity wizard --service-uid=1001` or `sudo GRAVITY_SERVICE_USER=1001 ./gravity wizard`.

NOTE: Run all commands as sudo or root user.

```
* [0/100] starting installer
confirm the config:

* IP address: 1.1.1.1

confirm (yes/no):
yes
* [0/100] starting installer
* [0/100] preparing for installation... please wait
* [0/100] application: AnacondaEnterprise:5.1.1
* [0/100] starting web UI install wizard

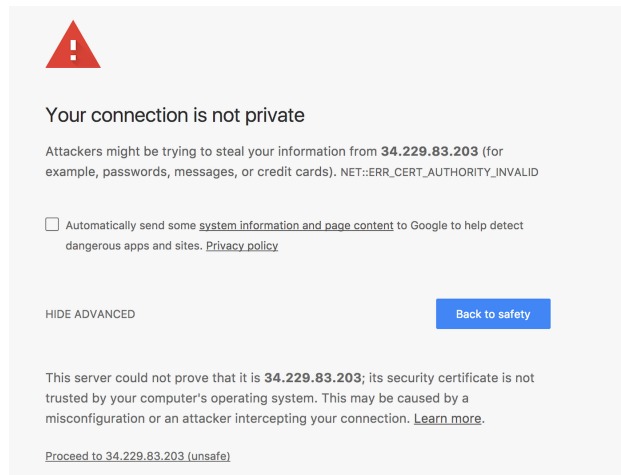
-----
↪-----
OPEN THIS IN BROWSER: https://1.1.1.1:61009/web/installer/new/gravitational.io/
↪AnacondaEnterprise/5.1.0?install_token=0a39d9a2f16036fc6583e78b502e7cae
-----
↪-----
```

Connect to the URL in your browser, ensuring that you are connecting to the public network interface:

https://1.1.1.1:61009/web/installer/new/gravitational.io/AnacondaEnterprise/5.0.1-343-g9c60a89?install_token=7b0415d5cc848f1d315efe2b92b099cf

The installer will install a self-signed TLS/SSL certificate, so you can proceed beyond the initial security warning:

Enter the Cluster Name. The Bare Metal option is already selected. Click Continue.



Your connection is not private


Attackers might be trying to steal your information from **34.229.83.203** (for example, passwords, messages, or credit cards). `NET::ERR_CERT_AUTHORITY_INVALID`

☐ Automatically send some [system information and page content](#) to Google to help detect dangerous apps and sites. [Privacy policy](#)

HIDE ADVANCED Back to safety

This server could not prove that it is **34.229.83.203**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection. [Learn more](#).

[Proceed to 34.229.83.203 \(unsafe\)](#)




Cluster Name

✓

Please enter a unique deployment name. Example: "AnacondaEnterprise.yourcompany"

Choose provider


Bare Metal

▼ Advanced Options

Continue

About this step

If you select a cloud provider, we will automate the infrastructure provisioning on your account with your provided keys in the next step. Your keys are not stored on our system.

If you select BareMetal we will provide you with a command to be run on each of your machines in the next step.

Select the number of nodes between one and four that you want to install to, where one node is the master node and the remaining nodes are worker nodes.

Run the following command on each node on which you want to install Anaconda Enterprise, copying the appropriate master node or worker node command:

```
curl -s --tlsv1.2 -k "https://1.1.1.1:61009/t/01eedaee7b9965c958336b5d31f334c3/worker" | sudo bash
```

As you run the command on each node, you'll see each node listed in the installer.

Click on Start Installation.

The installer will proceed through the installation stages. This process requires about 20 minutes.

LOCATIONCAPACITYINSTALLATION

How many nodes do you want?

1 node2 node3 node4 node

1 Anaconda Enterprise master node

Nodes required - RAM: 4.0GB, CPU: Core x 4

Add existing server

Copy and paste the command below into terminal. Your server will automatically appear in the list.

curl -s --tlsv1.2 --insecure "https://172.31.14.174:61009/t/de64daff4d8af327105cb79f151b140/master" | sudo bash

Copy Command

1 Anaconda Enterprise worker node(s)

Nodes required - RAM: 4.0GB, CPU: Core x 4

Add existing server

Copy and paste the command below into terminal. Your server will automatically appear in the list.

curl -s --tlsv1.2 --insecure "https://172.31.14.174:61009/t/de64daff4d8af327105cb79f151b140/worker" | sudo bash

Copy Command

1 Anaconda Enterprise master node

Nodes required - RAM: 4.0GB, CPU: Core x 4

Add existing server

Copy and paste the command below into terminal. Your server will automatically appear in the list.

curl -s --tlsv1.2 --insecure "https://172.31.14.174:61009/t/de64daff4d8af327105cb79f151b140/master" | sudo bash

Copy Command

Host name

ip-172-31-14-174

IP Address ⓘ

172.31.14.174

1 Anaconda Enterprise worker node(s)

Nodes required - RAM: 4.0GB, CPU: Core x 4

Add existing server

Copy and paste the command below into terminal. Your server will automatically appear in the list.

curl -s --tlsv1.2 --insecure "https://172.31.14.174:61009/t/de64daff4d8af327105cb79f151b140/worker" | sudo bash

Copy Command

Host name

ip-172-31-0-175

IP Address ⓘ

172.31.0.175

Verify

Start Installation

LOCATIONCAPACITYINSTALLATION

Installation progress

✓

✓

✓

Provisioning Instances ✓

Preparing configuration ✓

Installing application ⚙

Connecting to instances ✓

Installing dependencies ✓

Verifying application

Verifying instances ✓

Installing platform ✓

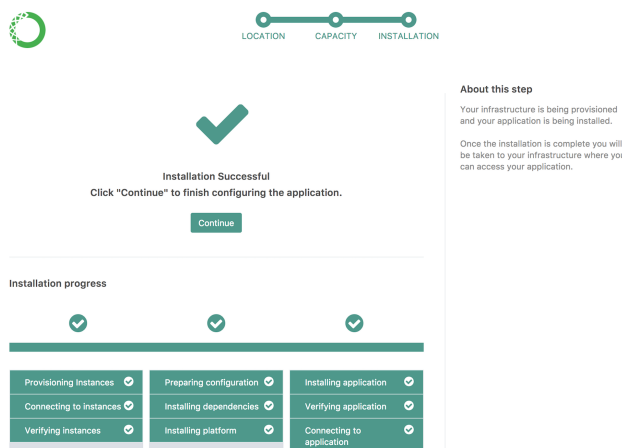
Connecting to application

About this step

Your Infrastructure is being provisioned and your application is being installed.

Once the installation is complete you will be taken to your infrastructure where you can access your application.

After the installation is complete, you will see the following screen.



NOTE: once the screen above is visible, the installer script in the terminal will note that installation is complete and that you can stop the installer process. Please **do not** do so until you have completed the post-install configuration, as below.

Click on the Continue button to proceed to *Post-Install Configuration*

Option 2. Unattended Command-line (CLI) Installation

Unattended CLI installation is used if you cannot connect to the nodes with a browser. This includes installation from a separate network.

On each node in the cluster, create a new directory for the installer, navigate to the new directory, then download and decompress the installer:

```
mkdir anaconda-enterprise
cd anaconda-enterprise
curl -O <location_of_installer>.tar.gz
tar xvfz anaconda-enterprise-<installer_version>.tar.gz
cd anaconda-enterprise-<installer_version>
```

Master node

On the master node, run the following command as sudo or root user, referring to details below this code block:

```
sudo ./gravity install --advertise-addr=192.168.1.1 --token=anaconda-enterprise --
↪flavor=small

* [0/100] starting installer
* [0/100] preparing for installation... please wait
* [0/100] application: AnacondaEnterprise:5.1.1
* [0/100] starting non-interactive install
* [0/100] still waiting for 1 nodes of role "worker" to join
```

(continues on next page)

(continued from previous page)

```
* [0/100] still waiting for 1 nodes of role "worker" to join
* [0/100] still waiting for 1 nodes of role "worker" to join
* [0/100] initializing the operation
* [20/100] configuring packages
* [50/100] installing software
```

NOTE: For the above “Advertised Address” option `advertise-addr`, replace the IP address with the address you want to be visible to the other nodes.

For the `flavor` option, choose one of:

- ‘tiny’: One node (one master node - DEFAULT)
- ‘small’: Two nodes (one master node and one worker node)
- ‘medium’: Three nodes (one master node and two worker nodes)
- ‘large’: Four nodes (one master node and three worker nodes)

NOTE:: By default, Anaconda Enterprise will install using a service account with the user ID (UID) 1000. You can change the UID of the service account by using the `--service-uid` option or the `GRAVITY_SERVICE_USER` environment variable. For example, to use UID 1001, you can use the command `sudo ./gravity install --service-uid=1001` or `sudo GRAVITY_SERVICE_USER=1001 ./gravity install`.

NOTE:: When installing to Amazon AWS, the installer autodetects EC2 and uses the VPC-based flannel backend instead of VXLAN. To force the use of VXLAN, you can use the `--cloud-provider generic` option in the above installation command.

Worker nodes

On each worker node, run the following command:

```
sudo ./gravity join 192.168.1.1 --advertise-addr=192.168.1.2 --token=anaconda-
↪enterprise --role=worker
* [0/100] joining cluster
* [0/100] connecting to cluster
* [0/100] connected to installer at 192.168.1.1
* [0/100] initializing the operation
* [20/100] configuring packages
* [50/100] installing software
```

NOTE: For the above “Advertised Address” option `advertise-addr`, replace the IP address with the address you want to be visible to the other nodes.

The installer will proceed through its stages. This process requires about 20 minutes.

After installing Anaconda Enterprise as part of an unattended installation, create a local user account and password to log into the Anaconda Enterprise Operations Center.

First, enter the Anaconda Enterprise environment on any of the master or worker nodes:

```
sudo gravity enter
```

Then, run the following command to create a local user account and password for the Anaconda Enterprise Operations Center:

```
gravity --insecure user create --type=admin --email=<email> --password=<yourpass> --
↪ops-url=https://gravity-site.kube-system.svc.cluster.local:3009
```

NOTE: Replace <email> with the email address, and <yourpass> with the password you want to use.

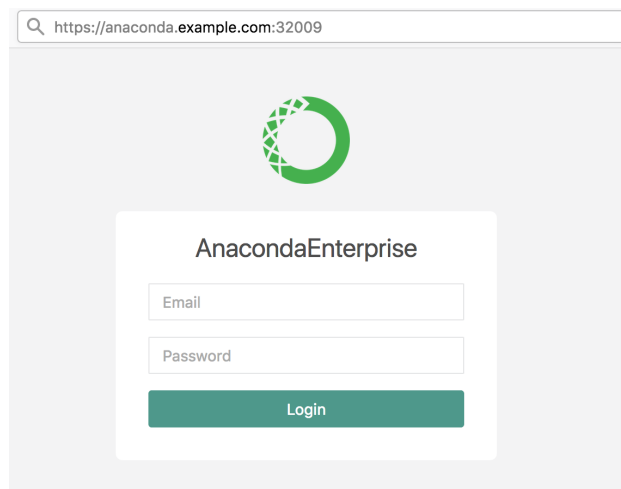
Both Options: Post-Install Configuration and Anaconda Enterprise Operations Center

After completing either installation path, you must continue to the *Post-Install Configuration*.

Access Anaconda Enterprise Operations Center

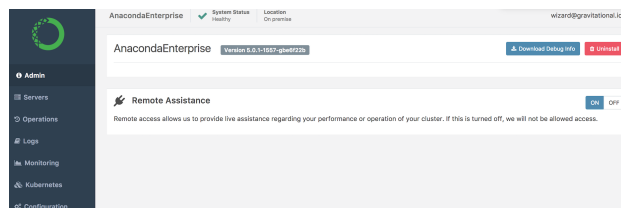
You can access the Anaconda Enterprise Operations Center in your browser by entering the following URL:

`https://anaconda.example.com:32009`



NOTE: Replace “anaconda.example.com” with your actual domain name.

After logging in, you will be viewing the Anaconda Enterprise Operations Center.



NOTE: Immediately after installation, the remote assistance feature may temporarily appear enabled. The connection is destroyed after a few minutes and will then stop appearing.

2.1.3 Air gap installation and configuration

This segment describes the instructions for installation on air gapped systems or other machines that do not have access to the internet.

The air gap archives contain installers and packages to mirror. The Anaconda Enterprise installer includes the necessary dependencies. Air gap installations must use offline licensing.

Please review and verify that you have met all *Anaconda Enterprise system requirements* before beginning your installation.

1. Download the installer tarball file to a jumpbox or USB key.
2. Move the installer tarball file to a designated head node in the airgap.
3. Untar the installer file and run `sudo ./gravity wizard` for browser-based installation or `sudo ./gravity install` for CLI-based installation.

Follow the [normal installation documentation](#).

2.1.4 License Configuration

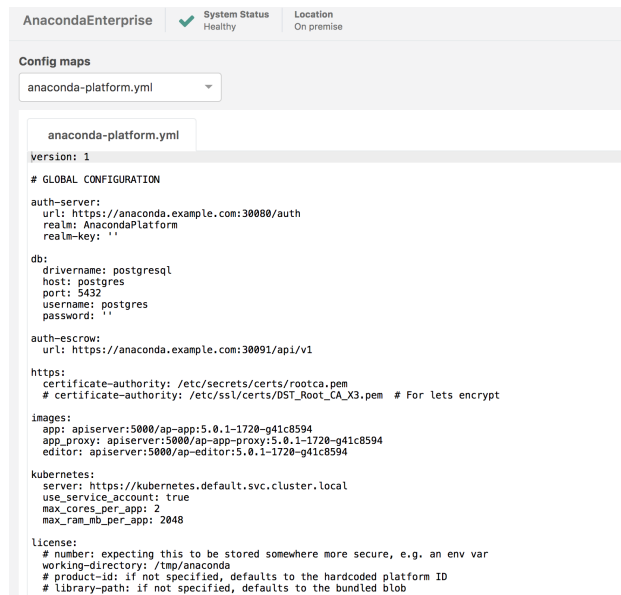
This page describes how to configure and activate your license.

Your Anaconda Enterprise license is included in your Welcome email from Anaconda, Inc.

Contact Priority Support if you cannot locate your Welcome email.

Online License Activation

1. Find your Anaconda Enterprise license code included in your Welcome email. The license code is an 18-digit number.
2. Navigate to the Configuration tab of the [Anaconda Enterprise Operations Center](#).



3. Scroll down to the License section, paste the license code as shown and click *Apply*:

```

license:
  number: PASTE_LICENSE_CODE_HERE

```

4. Restart the Enterprise UI service:

```
sudo gravity enter
kubectl get pods | grep ap-ui | cut -d' ' -f1 | xargs kubectl delete pods
```

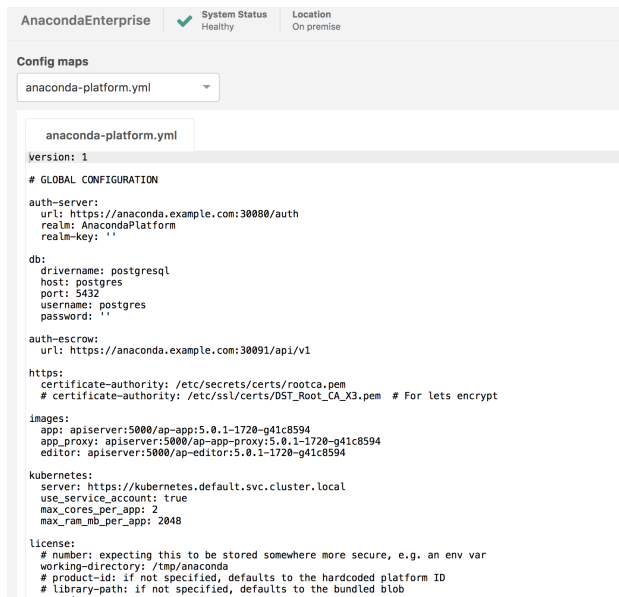
5. Confirm that the online license was activated by viewing the logs for the *ap-ui* container in the Anaconda Enterprise Operations Center:

```
2017-08-28 19:36:13.955 UTC INFO      ---- loaded license: LicenseStatus.ONLINE_
↪CONCURRENT ---- [root]
2017-08-28 19:36:13.956 UTC INFO      expiring at 2017-08-29 14:36:05 (in_
↪86392 seconds) [root]
```

Offline License Activation

An offline license can be used for air gapped installations or secure offline clusters.

1. Obtain an offline Enterprise license “number” and key from your License Fulfillment email. The “number” is an alphanumeric string, and the key is a longer alphanumeric string.
2. Navigate to the Configuration tab of the Anaconda Enterprise Operations Center.



3. Paste the offline license number and key into the following configuration section and click *Apply*:

```
license:
  number: PASTE_LICENSE_NUMBER_HERE
  key: PASTE_LICENSE_KEY_HERE
```

4. Restart the Anaconda Enterprise UI service:

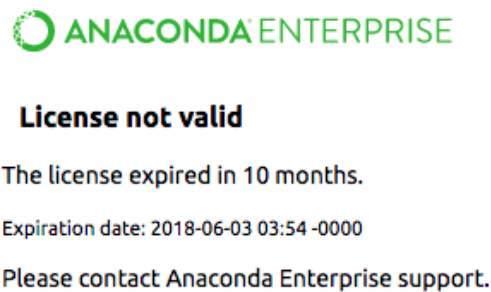
```
sudo gravity enter
kubectl get pods | grep ap-ui | cut -d' ' -f1 | xargs kubectl delete pods
```

5. Confirm that the offline license was activated by viewing the logs for the *ap-ui* container in the Anaconda Enterprise Operations Center:

```
2017-08-28 18:56:52.932 UTC WARNING Attempting offline activation for computer_
↪ID: yVd4H410uAnDCw2tbjtb [root]
2017-08-28 18:56:52.942 UTC INFO ---- loaded license: PassiveLicenseStatus.
↪FULL ---- [root]
2017-08-28 18:56:52.943 UTC INFO expiring at 2018-08-28 13:56:52.942696_
↪(in 31536000 seconds) [root]
```

Expired License

After your Anaconda Enterprise license expires, you will see the following message upon accessing the Anaconda Enterprise interface:



For licenses that were activated using the online method, contact your sales representative to renew your subscription period. In the case of online license activation, your license can be renewed and extended remotely without modifying your license code or configuration.

For licenses that were activated using the offline method, contact your sales representative to renew your subscription period and obtain a new license “number” and key. Then, follow the above steps for Offline License Activation and input your new license “number” and key.

2.1.5 Post-Install Configuration

The following configuration steps should be performed after a successful new installation of Anaconda Enterprise.

Follow either the instructions for a browser installation (Option 1) or the instructions for an unattended installation (Option 2). Then follow the instructions for final configuration.

Option 1: Browser Installation

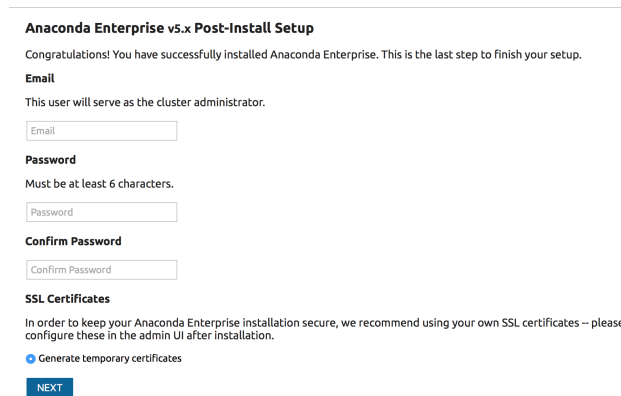
If the installation was performed using a browser, the post-install configuration UI will be available to guide you through many of the configuration steps.

NOTE: Please give the post-install UI a moment to start up. If you encounter an error immediately after clicking Continue at the end of the installation, please refresh your browser after a few seconds, and you should see the UI.

NOTE: When performing a browser-based installation, you must proceed through the post- install UI within 4 hours, or the installation UI will time out. If you would like to perform an unattended installation, please see Option 2 below.

On the screen immediately after clicking Continue at the end of the installation, create the initial admin account credentials that you will use to log in to the Anaconda Enterprise Operations Center and click the Next button.

This installation path will generate self-signed SSL certificates that can be changed later in the Admin page of Anaconda Enterprise. Enter the fully-qualified domain at which the cluster will be accessed and click the Finish setup button.



Anaconda Enterprise v5.x Post-Install Setup

Congratulations! You have successfully installed Anaconda Enterprise. This is the last step to finish your setup.

Email
This user will serve as the cluster administrator.

Email

Password
Must be at least 6 characters.

Password

Confirm Password

Confirm Password

SSL Certificates
In order to keep your Anaconda Enterprise installation secure, we recommend using your own SSL certificates -- please configure these in the admin UI after installation.

☒ Generate temporary certificates

NEXT

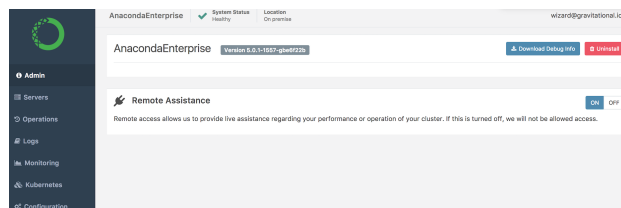
ANACONDA ENTERPRISE

Anaconda Enterprise v5.x Post-Install Setup

Cluster domain
Fully-qualified domain at which the cluster will be accessed. We need this to generate the appropriate SSL configuration to ensure your cluster is secure.

FINISH SETUP

You will be taken to the Anaconda Enterprise Operations Center where you can login with the admin credential you created in the beginning of this section.



Continue to the *Final Configuration* section.

Option 2: Unattended Installation

After completing an unattended installation, you must configure SSL, your FQDN, default authentication data, and allowed authentication endpoints.

1. Configure SSL for Anaconda Enterprise

- (a) Generate self-signed temporary certificates. On the master node, run:

```
cd path/to/Anaconda/Enterprise/unpacked/installer
cd DIY-SSL-CA
bash create_noprompt.sh DESIRED_FQDN
cp out/DESIRED_FQDN/secret.yaml /var/lib/gravity/planet/share/secrets.yaml
```

Replace `DESIRED_FQDN` with the fully-qualified domain of the cluster to which you are installing Anaconda Enterprise.

Saving this file as `/var/lib/gravity/planet/share/secrets.yaml` on the Anaconda Enterprise master node makes it accessible as `/ext/share/secrets.yaml` within the Anaconda Enterprise environment which can be accessed with the command `sudo gravity enter`.

- (b) Update the `certs` secret

Replace the built-in `certs` secret with the contents of `secrets.yaml`. Enter the Anaconda Enterprise environment and run these commands:

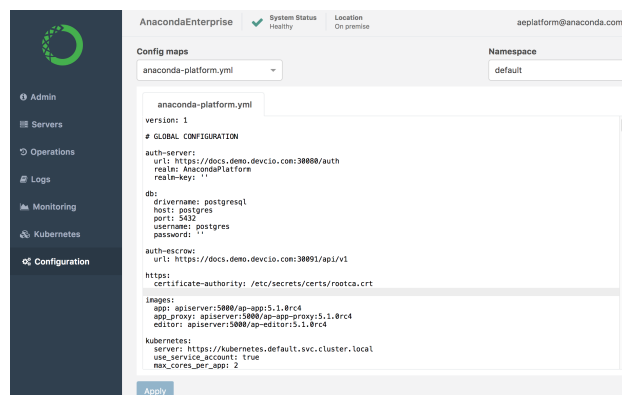
```
$ kubectl delete secrets certs
secret "certs" deleted
$ kubectl create -f /ext/share/secrets.yaml
secret "certs" created
```

2. Configure FQDN

To access Anaconda Enterprise, you'll need to edit the platform configuration settings to configure the fully qualified domain name (FQDN).

You can access the platform configuration in the Anaconda Enterprise Operations Center by visiting this URL in your browser: `https://anaconda.example.com:32009`

NOTE: Always replace `anaconda.example.com` with the domain name you are using.



3. Default Authentication Data

In the configuration file, add the following default data location to the `auth` section under the `import-file` key:

```
auth:
  import-file: /anaconda_enterprise/auth/anaconda_platform/auth/test_data/prod-
  ↪keycloak.json
```

4. Authentication Redirects and Allowed URLs

Now edit the redirect URLs in Anaconda Enterprise.

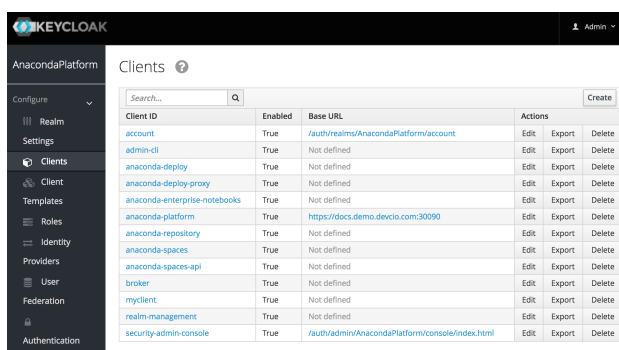
Access the authentication server on the Anaconda Enterprise cluster by visiting this URL in your browser: `https://anaconda.example.com:30080`

Access the Anaconda Enterprise Operations Center and login with the default username and password: `admin / admin`. You'll be asked to change the default password when you log in.

Check to be sure that you are on the `AnacondaPlatform` realm, then in the left menu, click on the `Clients` configuration tab.

Replace all URLs in the following client settings with the FQDN of the Anaconda Enterprise server:

- `anaconda-platform` - Valid Redirect URIs: `https://anaconda.example.com:30090/*`
- `anaconda-platform` - Base URL: `https://anaconda.example.com:30090`
- `anaconda-platform` - Web Origins: `https://anaconda.example.com:30090`
- `anaconda-spaces` - Valid Redirect URIs: `https://anaconda.example.com:*`



Client ID	Enabled	Base URL	Actions
account	True	/auth/realms/AnacondaPlatform/account	Edit Export Delete
admin-cli	True	Not defined	Edit Export Delete
anaconda-deploy	True	Not defined	Edit Export Delete
anaconda-deploy-proxy	True	Not defined	Edit Export Delete
anaconda-enterprise-notebooks	True	Not defined	Edit Export Delete
anaconda-platform	True	https://docs.demo.devic.com:30090	Edit Export Delete
anaconda-repository	True	Not defined	Edit Export Delete
anaconda-spaces	True	Not defined	Edit Export Delete
anaconda-spaces-api	True	Not defined	Edit Export Delete
broker	True	Not defined	Edit Export Delete
myclient	True	Not defined	Edit Export Delete
realm-management	True	Not defined	Edit Export Delete
security-admin-console	True	/auth/admin/AnacondaPlatform/console/index.html	Edit Export Delete

You can also add/remove users from the Anaconda Enterprise authentication server. The authentication server admin user exists under the `master` realm, and all other Anaconda Enterprise users exist under the `AnacondaPlatform` realm.

Both Options: Final Configuration

After completing either of the configuration paths above, follow these steps to complete the configuration.

1. *Install and configure your license.*

2. Access Anaconda Enterprise in your browser

In your browser, access Anaconda Enterprise by visiting this URL: `https://anaconda.example.com:30090`

3. Test installation

You can test your install by:

- *Creating* a new project and *starting* an editing session
- *Deploying* a project
- *Generating a token* from a deployment

Setting the database Password

1. Set a password in postgres.

(a) Connect to postgres pod: `kubectl exec -it postgres-<id> /bin/sh`

(b) Connect to database with psql: `psql -h localhost -U postgres`

(c) Set the password: `ALTER USER user_name WITH PASSWORD 'new_password';`

2. Open the Anaconda Enterprise Operations Center and navigate to the platform configuration page.

3. Enter the password at `db.password`.

4. Restart all the service pods: `kubectl get pods | grep ap- | cut -d' ' -f1 | xargs kubectl delete pods`

2.1.6 Storage Overview

Anaconda Enterprise includes an internal database, git server, and object storage server. All persistent storage is written to disk on the master node.

When projects or deployments run in Enterprise, the storage is ephemeral within containers. This means that when the editor or deployment is terminated, either intentionally or unintentionally, the data is not persisted in the container. To persist data, we mount disk storage from the underlying host into specific containers.

The default location for all storage-related assets including the database, project, and package storage in Enterprise is `/opt/anaconda/`. This location must be backed up frequently or located on a redundant disk array. Refer to the system requirements page for recommended disk space on the master and worker nodes.

The following outlines the storage configuration for Enterprise.

Database Storage

Enterprise stores state related to authentication/authorization, deployments, editor sessions, packages, projects, users, and other information.

We include an internal database server that writes to `/opt/anaconda/storage/pgdata` on the master node.

Git Storage

File storage for projects is backed by git.

We include an internal git server that writes to `/opt/anaconda/storage/git` on the master node.

Object Storage

The object storage is used for storing conda packages, Anaconda installers, custom Anaconda parcels for Cloudera CDH, and custom Anaconda management packs for Hortonworks HDP.

We include an internal object storage server that writes to `/opt/anaconda/storage/object` on the master node via an S3-compatible interface.

NFS Storage

Anaconda Enterprise supports an NFS storage option for the Object Storage Service.

To start, configure a [PersistentVolume](#) and a [PersistentVolumeClaim](#) for NFS on Kubernetes. The settings below configure an NFS drive with 300 GB and allocate 100GB for usage.

NFS Kube Setup

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-storage-remote
  labels:
    volume: nfs-storage-remote
spec:
  capacity:
    storage: 300Gi
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  nfs:
    server: NFS_HOST_IP
    path: "/var/nfs"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfs-volume-remote
spec:
  selector:
    matchLabels:
      volume: nfs-storage-remote
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
```

Object Storage Config

Re-configure the Kubernetes deployment spec for *Object Storage* and use the newly created *PersistentVolumeClaim* in *ap-object-storage.yaml* (deployment configuration):


```

volumeMounts:
  - mountPath: /export
    name: nfs-volume
volumes:
  - name: nfs-volume
    persistentVolumeClaim:
      claimName: nfs-volume-remote

```

Add in the NFS *PersistentVolumeClaim* and then restart the *object storage* pod. NFS paths will now be mounted inside the object-storage service.

2.1.7 Migrating Anaconda Repository 4.x packages

To migrate from Anaconda Repository 4.x to Anaconda Enterprise involves exporting a site-dump of all packages, then importing into Anaconda Enterprise.

Export all packages

You can use the command `anaconda-server-admin export-site` to create a dump of all the packages and information regarding owners and permissions of those packages.

This command creates a directory structure containing all files and user information from Repository.

Example:

```

site-dump/
├── anaconda-user-1
│   ├── 59921152446b5703f430383f--moto
│   ├── 5992115f446b5703fa30383e--pysocks
│   └── meta.json
├── anaconda-organization
│   ├── 5989fbd1446b575b99032652--future
│   ├── 5989fc1d446b575b99032786--iso8601
│   ├── 5989fc1f446b575b990327a8--simplejson
│   ├── 5989fc26446b575b99032802--six
│   ├── 5989fc31446b575b990328b0--xz
│   ├── 5989fc35446b575b990328c6--zlib
│   └── meta.json
└── anaconda-user-2
    └── meta.json

```

Each subdirectory of `site-dump` contains the contents of a user. For example `anaconda-user-1` has two packages, `moto` and `pysocks`. Inside the package directories, which are prefixed with the id of the database, are the package files. There is a `meta.json` in the user directories and the package directories with different metadata. The user's `meta.json` contains information regarding which groups the user belongs to (end users) or what groups the user has (organizations).

NOTE: Other files included in the site-dump such as projects and envs are NOT imported by the package import tool.

Importing packages

You can choose to import packages by username or directory, by all packages or by organization.

For all methods, log into the command line interface:

```
anaconda-enterprise-cli login
```

Then follow the instructions for the method you want to use.

Import by username or directory

The packages for each user is in a separate directory in the site-dump file, so the import process is the same for each username or directory.

Import a single directory from the site-dump with the command:

```
anaconda-enterprise-cli admin import site-dump/NAME
```

Where NAME is the name of the directory you want to import.

Import all packages

Then to import all packages, run the command:

```
anaconda-enterprise-cli admin import site-dump/*
```

As you can see from the glob operator (site-dump/*) you must pass a list of users you want to import.

What this script does:

For every user in a 4.x Repository, it will create a new channel for each label the user has, using the username as a prefix.

EXAMPLE:

Let's say the user anaconda-user-1 has the following packages:

- moto-0.4.31-2.tar.bz2 with label "main"
- pysocks-1.6.6-py35_0.tar.bz2 with label "test"

For this user, the script creates the following channels:

- anaconda-user-1 with file moto-0.4.31-2.tar.bz2
- anaconda-user-1/test with file pysocks-1.6.6-py35_0.tar.bz2

The default label "main" is dropped in preference of just keeping the username as channel name.

Import an organization

For each organization you want to import, the script adds the organization groups to the channel name it creates.

EXAMPLE:

Let's say anaconda-organization has a group called "Devs" and there are packages there such as xz-5.2.2-1.tar.bz2 which has label "Test".

This script creates the following channels:

- anaconda-organization - This contains everything the Owner group has
- anaconda-organization/Devs - This contains everything that is available for the Dev group
- anaconda-organization/Devs/Test - This contains everything in the Dev group with label "Test".

So for an organization, group, and label a new channel is created. The default labels “main” and group “Owner” are dropped in preference of shortening names.

After everything is uploaded, the channels are shared with the users. That is the channel `anaconda-user-1` is made read-writable by `anaconda-user-1`. The members of a group are given read permission on the organization’s channel.

2.1.8 Fault Tolerance

This page describes the fault tolerant behavior of Anaconda Enterprise.

The status of core Enterprise services and user deployments can be monitored from the [Anaconda Enterprise Operations Center](#).

Anaconda Enterprise can be deployed with automatically provided service redundancy and fault tolerance. Anaconda Enterprise employs automatic service restarts and health monitoring to remain operational if a process halts or a worker node becomes unavailable. Additional levels of fault tolerance, such as service migration, are provided if there are at least three nodes in the deployment. However, the Master Node cannot currently be configured for automatic failover and does present a single point of failure.

When Anaconda Enterprise is deployed to a cluster with three or more nodes the core services will automatically be configured into a fault tolerant mode. This configuration will take effect whether Anaconda Enterprise is initially configured this way, or due to changes at a later point in time. Once there are three or more nodes available the service fault tolerance features will come into effect.

Service migration is only possible when there are multiple worker nodes in the deployment.

Core Services

Anaconda Enterprise core services will automatically be restarted or, if possible, migrated in the event of any service failure.

User Deployments

User-initiated project deployments will automatically be restarted or, if possible, migrated in the event of any failure.

Worker Nodes

If any worker node becomes unresponsive or unavailable, it will be flagged while the core Enterprise services and backend continue to run without interruption. If additional worker nodes are available the services that had been running on the failed worker node will be migrated or restarted on other still-live worker nodes. This migration may take a few minutes.

New worker nodes can be added to the Enterprise cluster from the [Anaconda Enterprise Operations Center](#).

Storage and Persistency Layer

Anaconda Enterprise does not automatically configure storage or persistency layer fault tolerance when using the default storage and persistency services. This includes the database, git server, and object storage. If you have configured Anaconda Enterprise to use external storage and persistency services then you will need to configure these for fault tolerance.

Recovering Anaconda Enterprise After Node Failure

The Master Node presents a single point of failure. If it fails it can be recovered using the steps outlined on the *Master Node Failure Recovery* page.

2.1.9 Configuration Reference

This page explains how to configure the resource limits and the security settings in the Anaconda Enterprise configuration file, `anaconda-platform.yml`.

To access the Anaconda Enterprise configuration file, login to the Anaconda Enterprise Operations Center by visiting the following URL in your browser: `https://anaconda.example.com:32009` and click the Configuration link.

NOTE: Replace `anaconda.example.com` with the domain name you are using.

Complete Anaconda Enterprise Configuration File

See the full configuration file below, or download a copy to review.

Additional details regarding the configuration settings are shown below:

```
version: 1

# GLOBAL CONFIGURATION

auth-server: # Common authentication client settings for all services
  url: https://anaconda.example.com:30080/auth
  realm: AnacondaPlatform
  realm-key: ''

db: # Database client configuration
  drivename: postgresql # Database driver (default postgresql, which is currently
→the only driver supported)
  host: postgres # Database hostname
  port: 5432
  username: postgres
  password: ''

auth-escrow: # Common authentication client settings for all services
  url: https://anaconda.example.com:30091/api/v1

https: # Common HTTPS client and server settings for all services
  certificate-authority: /etc/ssl/certs/ca-certificates.crt # Path to Certificate
→Authority bundle for private CA or self-signed certificates
  # certificate-authority: /etc/ssl/certs/DST_Root_CA_X3.pem # For lets encrypt

images:
  app: apiserver:5000/ap-app:5.0.1-1896-g6d27a29
  app_proxy: apiserver:5000/ap-app-proxy:5.0.1-1896-g6d27a29
  editor: apiserver:5000/ap-editor:5.0.1-1896-g6d27a29

kubernetes:
  server: https://kubernetes.default.svc.cluster.local
  use_service_account: true
  max_cores_per_app: 2
```

(continues on next page)

(continued from previous page)

```

max_ram_mb_per_app: 2048

license:
  number: PASTE_LICENSE_CODE_OR_CLIENT_ID_HERE
  # key: PASTE_OFFLINE_KEY_HERE_FOR_OFFLINE_ACTIVATION
  working-directory: /tmp/anaconda
  security:
    x: 207
    y: 705
    z: 278
  analytics:
    enabled: true

# PER-SERVICE CONFIGURATION

auth: # Authentication server configuration
  port: 9080
  db:
    database: anaconda_auth
  https: # HTTPS configuration
    keystore: /etc/secrets/certs/keystore.jks # Name of server keystore in Java
    ↪keystore (.jks) format
    keystore-password: anaconda # Keystore password defined when generating the Java
    ↪keystore
    key-alias: auth # Name of the key in the keystore
    truststore: null # (optional) Path to the trust store to use for outgoing HTTPS
    ↪requests (e.g. for LDAPS)
    truststore-password: null # (optional) Truststore password defined when
    ↪generating the Java keystore
    debug: False # If true, enable use of a pregenerated SSL key for testing. DO NOT
    ↪SET TO TRUE IN PRODUCTION.
  api: # Service settings for auth-api
    port: 9090
    limit: 12
    https:
      key: /etc/secrets/certs/server.key
      certificate: /etc/secrets/certs/server.crt
  escrow: # Service settings for auth-escrow
    port: 9091
    db:
      database: anaconda_auth_escrow
    hosts: # List of hosts (host:port pairs) to allow in API request headers
      - anaconda.example.com:30091
    prefix: '' # URL prefix
    https:
      key: /etc/secrets/certs/server.key
      certificate: /etc/secrets/certs/server.crt
    auth-server:
      client-secret: ed7ec3ff-c535-455b-b431-5ed97d78b8be
      client-id: anaconda-platform

deploy: # Deployment server configuration
  port: 8081
  prefix: '' # URL prefix
  url: https://anaconda.example.com:30081/ # Deployment server URL
  https:
    key: /etc/secrets/certs/server.key

```

(continues on next page)

(continued from previous page)

```

    certificate: /etc/secrets/certs/server.crt
hosts: # List of hosts (host:port pairs) to allow in API request headers
    - anaconda.example.com:30081
db:
    database: anaconda_deploy
users: '*' # Users/groups who have permission to access deployed apps
deployers: # Users/groups who have permission to deploy here
    users: []
    groups:
        - developers
    roles: []
superusers: # Users/groups who have unrestricted access
    users: []
    groups: []
    roles: []
auth-server:
    client-id: anaconda-deploy
apps-host: anaconda.example.com # Hostname where apps are deployed, if different
↳from the one in kubernetes.server
auth-proxy: # Settings for deployed app proxy
    client-id: anaconda-deploy-proxy # Client ID of the proxy, as registered in the
↳auth service
    dns-server: 10.100.0.4 # IP address of DNS server used by the app proxy. Default
↳is the internal kubernetes resolver.
    https:
        key: /etc/secrets/certs/server.key
        certificate: /etc/secrets/certs/server.crt

    debug: False # If true, enable debugging. DO NOT SET TO TRUE IN PRODUCTION.

spaces: # Spaces server configuration
    port: 8090
    prefix: '' # URL prefix
    url: https://anaconda.example.com:30095/ # Spaces server URL
    https:
        key: /etc/secrets/certs/server.key
        certificate: /etc/secrets/certs/server.crt
    hosts: # List of hosts (host:port pairs) to allow in API request headers
        - anaconda.example.com:30095
    db:
        database: anaconda_spaces

    users: '*' # Users/groups who have permission to create spaces
    superusers: # Users/groups who have unrestricted access
        users: []
        groups: []
        roles: []

    auth-server:
        client-id: anaconda-spaces-api
        spaces-host: anaconda.example.com # Hostname where spaces are hosted, if different
↳from the one in kubernetes.server
        auth-proxy: # Settings for spaces access control proxy
            client-id: anaconda-spaces # Client ID of the proxy, as registered in the auth
↳service
            dns-server: 10.100.0.4 # IP address of DNS server used by the app proxy. Default
↳is the internal kubernetes resolver.

```

(continues on next page)

(continued from previous page)

```

https:
  key: /etc/secrets/certs/server.key
  certificate: /etc/secrets/certs/server.crt

debug: False # If true, enable debugging. DO NOT SET TO TRUE IN PRODUCTION.

storage: # Storage server configuration
  host: anaconda.example.com # full hostname of the storage server
  port: 8086
  prefix: '' # URL prefix
  hosts: # List of hosts (host:port pairs) to allow in API request headers
    - anaconda.example.com:30086
  url: https://anaconda.example.com:30086 # Base URL of storage server
  db:
    database: anaconda_storage
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  git:
    default:
      name: Example.com Anaconda Enterprise Server # human-readable name of this git_
↪server
      type: internal # server type. There is support for "internal" and planned_
↪support for "github" and "gitlab".
      url: https://anaconda.example.com:30088 # URL of git server
      repository: '{name}-{id}' # Template for repository names; use {name}, {id},_
↪and {owner} as placeholders.
      auth-header: Anaconda-User # Name of HTTP header for proxy authentication_
↪(internal server type only)
      username: anaconda # Username of git service account
      # no password needed when using auth-header
    proxy:
      url: https://anaconda.example.com:30085 # URL of git proxy
      client-id: anaconda-git-proxy # Auth client ID of this proxy
      dns-server: 10.100.0.4 # IP address of DNS server used by the git proxy.
      run-as-user: www-data # System user account to run the proxy under
      api-key: f49fece0b2ef8d122d4a2473278465f7c77781617428b7e18401f2d0139b39e7 #_
↪secret api key to allow storage service API calls through the proxy. Should be_
↪uniquely generated for each installation.
      port: 8095
      probe-port: 8096
      https:
        key: /etc/secrets/certs/server.key
        certificate: /etc/secrets/certs/server.crt
  objects:
    projects: # storage location for objects in projects. You may use placeholders
↪{name} {owner} and {id} for project name, project owner and project ID.
      bucket: anaconda-projects
      path: projects/{owner}-{id}
    global: # storage location for global objects (available to all logged-in users)
      bucket: anaconda-objects
      path: 'global/'
    public: # storage location for public objects (available to everyone without_
↪logging in)
      bucket: anaconda-objects
      path: 'public/'
  users: '*' # Users/groups who can create projects

```

(continues on next page)

(continued from previous page)

```

creators: # Users/groups who can create new projects
  users: []
  groups:
    - developers
  roles: []
superusers: # Users/groups who have unrestricted access
  users: []
  groups: []
  roles: []

repository: # Repository server configuration
  port: 8089
  hosts: # List of hosts (host:port pairs) to allow in API request headers
    - anaconda.example.com:30089
  prefix: '' # URL prefix
  db:
    database: anaconda_repository
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  users: '*' # Users/groups who can access the repository
  uploaders: # Users/groups who can create and upload packages
    users: []
    groups:
      - developers
    roles: []
  superusers: # Users/groups who have unrestricted access
    users: []
    groups: []
    roles: []
  bucket: anaconda-repository # S3/object storage bucket to store repository files
  auth-escrow:
    url: https://anaconda.example.com:30091/api/v1
  cleanup-upload-seconds: 3600 # How long an unfinished upload will be kept before_
↳being cleaned up
  cleanup-period-seconds: 73 # How frequently the server will check for files that_
↳should be removed from disk
  index-update-cooldown-seconds: 7 # How much time without new uploads is required_
↳before index will be rebuilt
  index-update-period-seconds: 23 # How frequently the server will check for channels_
↳that require rebuilding of index information (repodata.json)

s3: # configuration for the object-storage service
  host: 0.0.0.0 # full hostname of the object store server S3 API
  port: 8087
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  access-key: 's3-access-key'
  secret-key: 's3-secret-key'
  directory: /export

s3-client: # configuration for clients to the object storage service
  endpoint-url: https://anaconda.example.com:30087 # AWS endpoint URL
  access-key: 's3-access-key'
  secret-key: 's3-secret-key'
  region-name: 'us-east-1' # the AWS region where your S3 bucket is located

```

(continues on next page)

(continued from previous page)

```

git:
  url: https://anaconda.example.com:30088 # externally visible URL of the git server
  host: anaconda.example.com # full hostname of the git server
  port: 8088
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  db:
    database: anaconda_git
  directory: /export # directory where git server will store its data
  username: anaconda # OS username that the git server should run under
  lfs-secret: AohzzmIZVHYSTYJ7HM1E1GWhjRYCTcfLdxHHGR8fKCM # LFS authentication token
  ↪secret. Should be uniquely generated for each installation.
  secret-key: E3P99Z3XRAXaoJHGygmCjZ613pIZ9nvg6SnVRrPHTBU # git server secret key.
  ↪Should be uniquely generated for each installation.

conda: # Common conda settings for editing sessions and deployments
  channels: # List of channels to put in .condarc
    - defaults
  default-channels: [] # List of channels that should be used for channel 'defaults'
  channel-alias: https://anaconda.example.com:30089/conda # Default conda URL prefix
  ↪for channels given by name only

offline_docs:
  url: https://anaconda.example.com:30071 # Docs server URL
  hosts: # List of hosts (host:port pairs) to allow in API request headers
    - anaconda.example.com:30071
  port: 8091
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  directory: docs/_build/ # The path relative to the base directory of the static
  ↪docs.
  prefix: '' # URL prefix

ui: # Anaconda Platform UI server configuration
  base-url: / # URL prefix
  cookie-secret: this-is-a-very-insecure-secret # secret key used to sign session
  ↪cookies
  cookie-session:
    name: anaconda-platform-ui-session-v1
  cookie-next:
    name: anaconda-platform-ui-next-v1
  db:
    database: anaconda_ui
  debug: False # If true, enable debugging. DO NOT SET TO TRUE IN PRODUCTION.
  host: anaconda.example.com # full hostname of the UI server
  public-url: https://anaconda.example.com:30090/ # User-facing URL of site, if
  ↪different than host/port
  https:
    key: /etc/secrets/certs/server.key
    certificate: /etc/secrets/certs/server.crt
  port: 6990
  auth-server:
    client-secret: ed7ec3ff-c535-455b-b431-5ed97d78b8be
    client-id: anaconda-platform

```

(continues on next page)

(continued from previous page)

```

services:
  anaconda-storage:
    storage:
      icon: fa-anaconda
      label: Storage
      url: https://anaconda.example.com:30086/api/v1
  anaconda-deploy:
    deploy:
      icon: fa-anaconda
      label: Deploy
      url: https://anaconda.example.com:30081/api/v1
  anaconda-spaces:
    spaces:
      icon: fa-anaconda
      label: Spaces
      url: https://anaconda.example.com:30095/api/v1
    options:
      spaces:
        tools:
          notebook:
            default: true
            label: Jupyter Notebook Classic
            packages: [notebook]
          lab-pre:
            label: JupyterLab
            packages: [jupyterlab]
          sync:
            label: Anaconda Project Sync
            packages: [anaconda-platform-sync]
    templates:
      jupyter-5:
        label: Jupyter Notebook Classic
        tools:
          - notebook
          - sync
      jupyterlab:
        label: JupyterLab
        default: true
        tools:
          - lab-pre
          - sync
  anaconda-repo5:
    repo:
      html-url: https://anaconda.example.com:30089
      icon: fa-anaconda
      label: Repo Service
      url: https://anaconda.example.com:30089/api
  auth-api:
    auth-api:
      icon: fa-anaconda
      label: Auth API
      url: https://anaconda.example.com:30082/api/v1
  documentation:
    offline_docs:
      html-url: https://anaconda.example.com:30071
      icon: fa-anaconda
      label: Documentation

```

(continues on next page)

(continued from previous page)

```

    url: https://anaconda.example.com:30071
help: # Help links
  docs:
    label: Anaconda Documentation - Home
    external: true
    href: https://anaconda.example.com:30071
    position: 0
  started:
    label: Getting Started with Anaconda Enterprise
    external: true
    href: https://anaconda.example.com:30071/user-guide/getting-started.html
    position: 1
  release:
    label: Release Notes
    external: true
    href: https://anaconda.example.com:30071/release-notes.html
    position: 2
  support:
    label: Support
    external: true
    href: https://anaconda.example.com:30071/help-support.html
    position: 3
  feedback:
    label: Feedback
    external: true
    href: https://continuum.typeform.com/to/TnHsme
    position: 4

postgresql: # PostgreSQL server configuration
  port: 7080

```

Setting Resource Limits for Project Editor Sessions and Deployments

Each project editor session and deployment uses compute resources on the Anaconda Enterprise cluster.

NOTE: We strongly recommend to save a copy of the original file before making any edits.

You can configure the maximum number of cores and amount of memory/RAM that each project editor or deployment consumes. This is a global setting across the cluster that applies to all users, nodes, editor sessions, and deployments:

```

kubernetes:
  max_cores_per_app: 2
  max_ram_mb_per_app: 2048

```

Required security settings

These values and credentials must be set for every installation.

- `s3.access-key` and `s3.secret-key` for the Minio internal object store
- `s3-client.access-key` and `s3-client.secret-key` for the object store client. When using the internal object store, these must match `s3.access-key` and `s3.secret-key`.
- `auth.https.keystore-password` matching the password used when creating the Java keystore for the auth service

- `git.lfs-secret` and `git.secret-key` for the internal git server
- `storage.git.<server>.proxy.api-key`
- `ui.cookie-secret`

Configuring Outbound SSL (for systems such as Secure LDAP)

See *LDAP configuration*.

2.1.10 Mirroring the Anaconda repository

You can create a local copy of the [Anaconda repository](#). The mirror can be complete, partial, or include or exclude specific packages or types of packages. You can also create a mirror in an air gapped environment.

This page explains how to use Anaconda Enterprise’s convenient syncing tools to create and configure local mirrors for Anaconda’s Python and R packages.

NOTE: It can take hours to mirror the full repository.

Before you begin

You need to have completed installing and configuring Anaconda Enterprise.

1. Install `anaconda-enterprise-cli`

The Anaconda Enterprise installer tarball contains a `cas-mirror-VERSION.sh` script, which is a custom Miniconda installation that contains the `anaconda-enterprise-cli` package.

NOTE: `bzip` is required to install packages from Miniconda.

Navigate to the directory where you downloaded and extracted the Anaconda Enterprise installer, then install the bootstrap Miniconda environment to `~/cas-mirror`:

```
$ cd anaconda-enterprise
$ ./cas_mirror-5.x.x.x-linux-64.sh
Welcome to cas_mirror <anaconda-enterprise-installer_version>
[...]
```

NOTE: Replace “5.x.x.x” with your actual version number.

The installer prompts “In order to continue the installation process, please review the license agreement.” Click Enter to view license terms. Scroll to the bottom of the license terms and enter “Yes” to agree.

The installer prompts you to click Enter to accept the default install location, CTRL-C to cancel the installation, or specify an alternate installation directory. We recommend that you accept the default install location.

The installer prompts “Do you wish the installer to prepend the `cas_mirror` install location to PATH in your `/home/centos/.bashrc` ?”. We recommend “yes”.

The installer finishes and displays “Thank you for installing `cas_mirror`!”. Close and open your terminal window for the installation to take effect.

In your new terminal window, activate the custom Miniconda environment with the following command:

```
source ~/cas-mirror/bin/activate
```

2. Configure Anaconda URL

Configure the Anaconda URL using the following commands:

```
anaconda-enterprise-cli config set sites.master.url https://anaconda.example.
↪com:30089/api
anaconda-enterprise-cli config set default_site master
```

NOTE: Always replace `anaconda.example.com` with the domain name you are using.

3. Verify and add SSL certificates

If the root CA is contained in the certificate bundle at `/etc/pki/tls/certs/ca-bundle.crt`, use `openssl` to verify the certificates and make sure the final Verify return code is 0:

```
openssl s_client -connect anaconda.example.com:30089 -CAfile /etc/pki/tls/certs/
↪ca-bundle.crt
...
Verify return code: 0 (ok)
```

If you are using privately signed certificates, *extract the rootca*, then use `openssl` to verify the certificates and make sure the final Verify return code is 0:

```
openssl s_client -connect anaconda.example.com:30089 -CAfile rootca.crt
...
Verify return code: 0 (ok)
```

Configure the SSL certificates for the repository using the following commands:

```
$ anaconda-enterprise-cli config set ssl_verify true

# On Ubuntu
$ anaconda-enterprise-cli config set sites.master.ssl_verify /etc/ssl/certs/ca-
↪certificates.crt

# On CentOS/RHEL
$ anaconda-enterprise-cli config set sites.master.ssl_verify /etc/pki/tls/certs/
↪ca-bundle.crt
```

NOTE: If you are using a self-signed certificate or a certificate signed by a private CA, *extract the rootca*. Then either:

- Use the `anaconda-enterprise-cli config set sites.master.ssl_verify` command to add that root certificate, or
- Add that root certificate to the default Ubuntu or CentOS/RedHat trusted CA bundles.

4. Log into Anaconda Enterprise as an existing user using the following command:

```
$ anaconda-enterprise-cli login
Username: anaconda-enterprise
Password:
Logged anaconda-enterprise in!
```

NOTE: If Anaconda Enterprise 5 is installed in a proxied environment, see *Mirroring in an environment with a proxy* for information on setting the `NO_PROXY` variable.

Extracting self-signed SSL certificates

You may need the temporary self-signed Anaconda Enterprise certificates for later use. For example, when installing the `anaconda-enterprise-cli` tool, you will need to configure it to point to the self-signed certificate authority.

First, enter the Anaconda Enterprise environment:

```
sudo gravity enter
```

Then, run the below command for each certificate file you wish to extract, replacing `rootca.crt` below with the name of the specific file:

```
kubectl get secrets certs -o go-template='{{index .data "rootca.crt"}}' | base64 -d >   
↪ /ext/share/rootca.crt
```

Once you have run this command, the file will be available on the master node filesystem at `/var/lib/gravity/planet/share/<filename>`.

The available certificate files are:

- `rootca.crt`: the root certificate authority bundle
- `server.crt`: the SSL certificate for individual services
- `server.key`: the private key for the above certificate
- `wildcard.crt`: the SSL certificate for “wildcard” services, such as deployed apps and spaces
- `wildcard.key`: the private key for the above certificate
- `keystore.jks`: the Java Key Store containing these certificates used by some services

Mirror Anaconda

An example configuration file is provided here for mirroring the default Anaconda packages for the `linux-64` platform. These files are also included in the mirror tool installation:

```
# This is destination channel of mirrored packages on your local repository.
dest_channel: anaconda

# conda packages from these channels are mirrored to dest_channel on your local   
↪ repository.
channels:
  - https://repo.continuum.io/pkgs/main/
  - https://repo.continuum.io/pkgs/free/
  - https://repo.continuum.io/pkgs/pro/

# if doing a mirror from an airgap tarball, the channels should point to the tarball:
# channels:
#   - file:///path-to-expanded-tarball/repo-mirrors-<date>/anaconda-suite/pkgs/

# Only conda packages of these platforms are mirrored.
# Omitting this will mirror packages for all platforms available on specified   
↪ channels.
# If the repository will only be used to install packages on the v5 system, it only   
↪ needs linux-64 packages.
platforms:
  - linux-64
```

Mirror the contents of the repository:

```
cas-sync-api-v5 --file ~/cas-mirror/etc/anaconda-platform/mirrors/anaconda.yaml
```

This mirrors all of the packages from the Anaconda repository into the anaconda channel. If the channel does not already exist, it will be automatically created and shared with all authenticated users.

You can customize the permissions on the mirrored packages by [sharing the channel](#).

Verify in your browser by logging into your account and navigating to the Packages tab. You should see a list of the mirrored packages.

Mirror R packages

An example configuration file for R packages is also provided:

```
# This is destination channel of mirrored packages on your local repository.
dest_channel: r

# conda packages from these channels are mirrored to dest_channel on your local
↪ repository.
channels:
  - https://repo.continuum.io/pkgs/r/

# if doing a mirror from an airgap tarball, the channels should point to the tarball:
# channels:
#   - file:///path-to-expanded-tarball/repo-mirrors-<date>/r/pkgs/

# Only conda packages of these platforms are mirrored.
# Omitting this will mirror packages for all platforms available on specified
↪ channels.
# If the repository will only be used to install packages on the v5 system, it only
↪ needs linux-64 packages.
platforms:
  - linux-64
```

```
cas-sync-api-v5 --file ~/cas-mirror/etc/anaconda-platform/mirrors/r.yaml
```

Configure conda

After creating the mirror, configure Anaconda Enterprise to add this new mirrored channel to the default channels. This will make the packages available to users using the project editing and deployment features. You can do this by [editing your Anaconda Enterprise configuration](#) to include the appropriate channel:

```
conda:
  channels:
    - defaults
  default-channels:
    - anaconda
    - r
  channel-alias: https://<anaconda.example.com>:30089/conda
```

NOTE: Replace `<anaconda.example.com>` with the actual URL to your installation of Anaconda Enterprise.

NOTE: The ap-spaces service must be restarted for the configuration change to take effect on new project editor sessions.

Share channels

To make your new channels visible to your users in the web interface Packages list, share the channels with them.

EXAMPLE: To share new channels “anaconda” and “r” with group ‘everyone’ for read access:

```
anaconda-enterprise-cli channels share --group everyone --level r anaconda
anaconda-enterprise-cli channels share --group everyone --level r r
```

After running the share command, verify by logging onto the user interface and viewing the Packages list.

SEE ALSO: *Creating and sharing channels*

Partial mirror

Alternately, you may not wish to mirror all packages. You can specify which platforms you want to include, or use the whitelist, blacklist or license_blacklist functionality to control which packages are mirrored, by editing the provided mirror files:

```
cas-sync-api-v5 --file ~/my-custom-anaconda.yaml
```

In an air-gapped environment

To mirror the repository in a system with no internet access, create a local copy of the repository using a USB drive provided by Anaconda, and point cas-sync-api-v5 to the extracted tarball.

First, mount the USB drive and extract the tarball. In this example we will extract to /tmp:

```
cd /tmp
tar xvf <path to>/mirror.tar
```

NOTE: Replace <path to> with the actual path to the mirror file.

Now you have a local file-system repository located at /tmp/mirror/pkgs. You can mirror this repository. Edit /etc/anaconda-platform/mirrors/anaconda.yaml to contain:

```
channels:
  - /tmp/mirror/pkgs
```

And then run the command:

```
cas-sync-api-v5 --file etc/anaconda-platform/mirrors/conda.yaml
```

This mirrors the contents of the local file-system repository to your Anaconda Enterprise installation under the user-name ‘anaconda.’

Mirror configuration options

remote_url

Specifies the remote URL from which the conda packages and the Anaconda and Miniconda installers are downloaded. The default value is: <https://repo.continuum.io/>.

channels

Specifies the remote channels from which conda packages are downloaded. The default is a list of the channels `<remote_url>/pkgs/free/` and `<remote_url>/pkgs/pro/`

All specification information should be included in the same file, and can be passed to the `cas-sync-api-v5` command via the `--file` argument:

```
cas-sync-api-v5 --file ~/cas-mirror/etc/anaconda-platform/mirrors/anaconda.yaml
```

destination channel

The configuration option `dest_channel` specifies where files will be uploaded. The default value is: `anaconda`

SSL verification

The mirroring tool uses two different settings for configuring SSL verification. When the mirroring tool connects to its destination, it uses the `ssl_verify` setting from `anaconda-enterprise-cli` to determine how to validate certificates. For example, to use a custom certificate authority:

```
anaconda-enterprise-cli config set sites.master.ssl_verify /etc/ssl/certs/ca-
↪certificates.crt
```

The mirroring tool uses conda's configuration to determine how to validate certificates when connecting to the source that it is pulling packages from. For example, to disable certificate validation when connecting to the source:

```
conda config --set ssl_verify false
```

Mirroring in an environment with a proxy

If Anaconda Enterprise 5 is installed in a proxied environment, set the `NO_PROXY` variable. This ensures the mirroring tool does not use the proxy when communicating with the repository service, and prevents errors such as “Max retries exceeded”, “Cannot connect to proxy”, and “Tunnel connection failed: 503 Service Unavailable”.

```
export NO_PROXY=<master-node-domain-name>
```

Platform-specific mirroring

By default, the `cas-sync-api-v5` tool mirrors all platforms. If you do not need all platforms, edit the YAML file to specify the platform(s) you want mirrored:

```
platforms:
  - linux-64
  - win-32
```

Package-specific mirroring

In some cases you may want to mirror only a small subset of the repository. Rather than blacklisting a long list of packages you do not want mirrored, you can instead simply enumerate the list of packages you DO want mirrored.

NOTE: This argument cannot be used with the `blacklist`, `whitelist` or `license_blacklist` arguments. It can be used with the `platform-specific` argument.

EXAMPLE:

```
pkg_list:
- accelerate
- pyqt
- zope
```

This example mirrors only the three packages: Accelerate, PyQt & Zope. All other packages will be completely ignored.

Python version-specific mirroring

Mirror the repository with a Python version or versions specified.

EXAMPLE:

```
python_versions:
- 3.3
```

Mirrors only Anaconda packages built for Python 3.3.

License blacklist mirroring

The mirroring script supports license blacklisting for the following license families:

```
AGPL
GPL2
GPL3
LGPL
BSD
MIT
Apache
PSF
Public-Domain
Proprietary
Other
```

EXAMPLE:

```
license_blacklist:
- GPL2
- GPL3
- BSD
```

This example mirrors all the packages in the repository EXCEPT those that are GPL2-, GPL3-, or BSD-licensed, because those three licenses have been blacklisted.

Blacklist mirroring

The blacklist allows access to all packages EXCEPT those explicitly listed.

EXAMPLE:

```

blacklist:
- bzip2
- tk
- openssl

```

This example mirrors the entire repository except the bzip2, Tk, and OpenSSL packages.

Whitelist mirroring

The whitelist functions in combination with either the `license_blacklist` or `blacklist` arguments, and re-adds packages that were excluded by a previous argument.

EXAMPLE:

```

license_blacklist:
- GPL2
- GPL3
whitelist:
- readline

```

This example mirrors the entire repository EXCEPT any GPL2- or GPL3-licenses packages, but including readline, despite the fact that it is GPL3-licensed.

Combining multiple mirror configurations

You may find that combining two or more of the arguments above is the easiest way to get the exact combination of packages that you want.

The platform argument is evaluated before any other argument.

EXAMPLE: This example mirrors only Linux-64 distributions of the dnspython, Shapely and GDAL packages:

```

platforms:
- linux-64
pkg_list:
- dnspython
- shapely
- gdal

```

If the `license_blacklist` and `blacklist` arguments are combined, the `license_blacklist` is evaluated first, and the `blacklist` is a supplemental modifier.

EXAMPLE: In this example, the mirror configuration does not mirror GPL2-licensed packages. It does not mirror the GPL3 licensed package pyqt because it has been blacklisted. It does mirror all other packages in the repository:

```

license_blacklist:
- GPL2
blacklist:
- pyqt

```

If the `blacklist` and `whitelist` arguments are both employed, the `blacklist` is evaluated first, with the `whitelist` functioning as a modifier.

EXAMPLE: This example mirrors all packages in the repository except astropy and pygments. Despite being listed on the blacklist, accelerate is mirrored because it is listed on the whitelist.

```
blacklist:
- accelerate
- astropy
- pygments
whitelist:
- accelerate
```

2.1.11 Point conda to on-premises repository

This page explains the configuration to install packages from your local Anaconda Repository.

You can configure conda to search for packages in your on-premises repository. This can either be done at the system level, which overrides any user-level configuration files installed by the user, or on an individual machine basis.

Either way, you create a `.condarc` system configuration file in the root directory.

Edit your system `~/ .condarc` file to add the appropriate channel:

```
channels:
- http://<anaconda.enterprise>:30089/conda/anaconda/
```

NOTE: Replace `<anaconda.enterprise>` with the actual URL to your installation of Anaconda Enterprise.

Users can log in to the on-premises repository and install packages from the on-premises repository by *installing the Anaconda command line interface (CLI)*.

Allow access to other package repositories

If users are permitted to install packages from off-site package repositories, it is convenient to provide apps and editing sessions with access to the Anaconda channels by default.

To do so, *edit your Anaconda Enterprise configuration* to include the appropriate channels, as below:

```
conda:
  channels:
  - defaults
  default-channels:
  - https://repo.continuum.io/pkgs/main
  - https://repo.continuum.io/pkgs/free
  - https://repo.continuum.io/pkgs/r
  - https://repo.continuum.io/pkgs/pro
  channel-alias: https://<ANACONDA_ENTERPRISE_FQDN>:30089/conda
```

As with all changes to the configuration, you must then restart the Anaconda Enterprise services:

```
sudo gravity enter
kubectl get pods | grep 'ap-' | cut -d' ' -f1 | xargs kubectl delete pods
```

2.1.12 Federating Users with LDAP

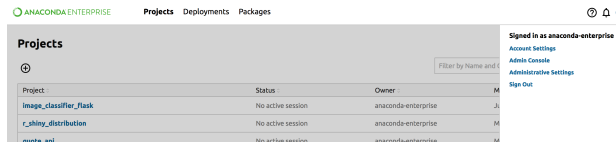
Anaconda Enterprise comes with out-of-the-box support for LDAP / Active Directory. As each enterprise configuration of LDAP / AD is different, this section offers general guidance only. We strongly recommend coordinating with your LDAP administrator.

NOTE: You must have pagination turned off before starting.

Adding a Provider

If you are not logged in as an administrator, do so.

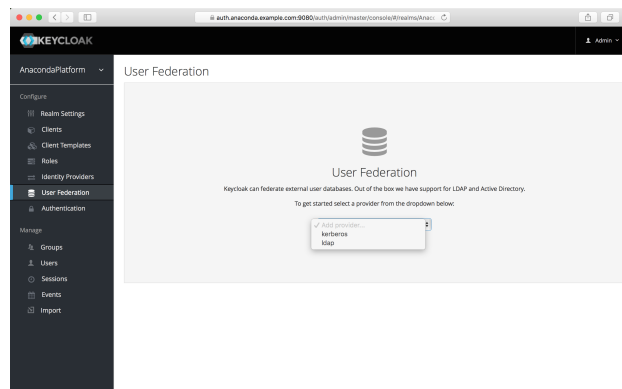
In the right-hand user menu click the Admin Console entry.



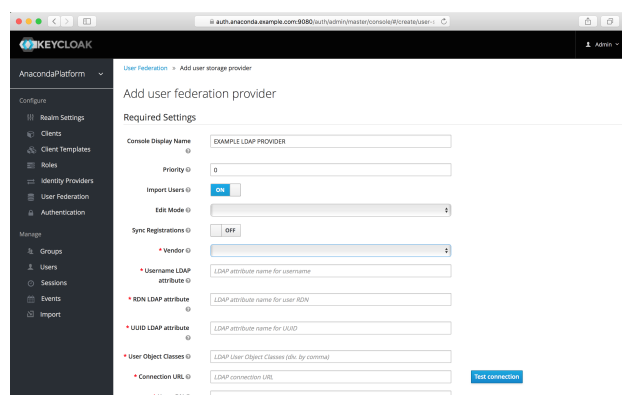
The login screen displays. Log in again as the admin.

In the Admin Console select the User Federation left menu item.

To begin configuring an LDAP identity provider, go to the right side to the Add Provider list box and choose the “ldap” provider type.



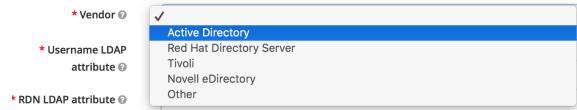
The initial required settings screen for your LDAP provider displays:



Multiple fields are required. The most important is the Vendor dropdown, which allows settings to be pre-filled to the defaults for different LDAP providers. Options include Active Directory, Red Hat Directory Server, Tivoli, and Novell eDirectory.

Make sure you select the correct one. If none of these matches, select Other and coordinate with your LDAP administrator when filling out this list of required fields:

Username LDAP attribute Name of the LDAP attribute that will be mapped to the username. Active Directory installations may use `cn` or `sAMAccountName`. Others often use `uid`.

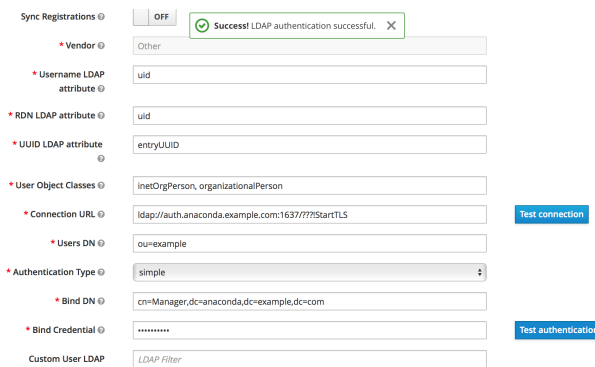


RDN LDAP attribute Name of the LDAP attribute that will be used as the RDN for a typical user DN lookup. This is often the same as the above “Username LDAP attribute”, but does not have to be. For example, Active Directory installations may use `cn` for this attribute while using `sAMAccountName` for the “Username LDAP attribute”.

UUID LDAP attribute Name of an LDAP attribute that will be unique across all users in the tree. For example, Active Directory installations should use `objectGUID`. Other LDAP vendors typically define a UUID attribute, but if your implementation does not have one, any other unique attribute (such as `uid` or `entryDN`) may be used.

User Object Classes Values of the LDAP `objectClass` attribute for users, separated by a comma. This is used in the search term for looking up existing LDAP users, and if read-write sync is enabled, new users will be added to LDAP with these `objectClass` values as well.

Make sure your connection to the LDAP server is correctly configured. The settings screen provides convenient buttons to test that a connection is working and is correctly authenticated:



Syncing Settings

By default, users will not be synced from the LDAP / Active Directory store until they log in. If you have a large number of users to import, it can be helpful to set up batch syncing and periodic updates.




Configuring Mappers



After you complete the initial setup the auth system generates a set of “mappers” for your configuration. Each mapper takes a value from LDAP and maps it to a value in the internal auth database. For example, the `username` mapper sets the Anaconda Enterprise username from the LDAP attribute configured.

Go through each mapper and make sure it is set up appropriately.

- Most important, check that each mapper reads the correct “LDAP attribute” and maps it to the right “User Model Attribute”.
- Check that the attribute’s “read-only” setting is correct.


Ldap 

Settings Mappers


Search...  


Name	Type
username	user-attribute-ldap-mapper
first name	user-attribute-ldap-mapper
last name	user-attribute-ldap-mapper
email	user-attribute-ldap-mapper
creation date	user-attribute-ldap-mapper
modify date	user-attribute-ldap-mapper


- Check whether the attribute should always be read from the LDAP store and not from the internal database.


Username 


ID


Name * 


Mapper Type 


User Model Attribute 

LDAP Attribute 

Read Only  ☒

Always Read Value From LDAP  ☐

Is Mandatory In LDAP  ☒

Is Binary Attribute  ☐

LDAP Authorization

To authorize LDAP group members, or roles synced from LDAP, to perform various functions, add them to the `anaconda-platform.yml` configmap.

EXAMPLE: To give users in the LDAP group “AE5”, and users with the LDAP-synced role “Publisher”, permission to deploy apps, the deploy section would look like this:

```
deploy:
  port: 8081
  prefix: ''
  url: https://abc.demo.anaconda.com:30081/
  https:
    key: /etc/secrets/certs/privkey.pem
    certificate: /etc/secrets/certs/cert.pem
  hosts:
    - abc.demo.anaconda.com:30081
  db:
    database: anaconda_deploy
  users: '*'
  deployers:
    users: []
    groups:
      - developers
      - AE5
  roles:
    - Publisher
```

As always when editing the configmap, delete the appropriate pods for changes to take effect.

Advanced Mapper Configuration

Instead of manually configuring each user, you can automatically import user data from LDAP using additional mappers. The following mappers are available:

User Attribute Mapper (`user-attribute-ldap-mapper`) Maps LDAP attributes to attributes on the user. These are the default mappers set up in the initial configuration.

FullName Mapper (`full-name-ldap-mapper`) Maps the full name of the user from LDAP into the internal database.

Role Mapper (`role-ldap-mapper`) Sets role mappings from LDAP into realm role mappings. One role mapper can be used to map LDAP roles (usually groups from a particular branch of an LDAP tree) into realm roles with corresponding names.

Multiple role mappers can be configured for the same provider. It's possible to map roles to a particular client (such as the `anaconda-deploy` service), but it's usually best to map in realm-wide roles.

Hardcoded Role Mapper (`hardcoded-ldap-role-mapper`) Grants a specified role to each user linked with LDAP.

Hardcoded Attribute Mapper (`hardcoded-ldap-attribute-mapper`) Sets a specified attribute to each user linked with LDAP.

Group Mapper (`group-ldap-mapper`) Sets group mappings from LDAP. Can map LDAP groups from a branch of an LDAP tree into groups in the Anaconda Platform realm. It will also propagate user-group membership from LDAP. We generally recommend using roles and not groups, so the role mapper may be more useful. **CAUTION:** The group mapper provides a setting `Drop non-existing groups during sync`. If this setting is turned on, existing groups in Anaconda Enterprise Authentication Center will be erased.

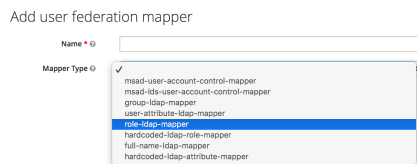
MSAD User Account Mapper (`msad-user-account-control-mapper`) Microsoft Active Directory (MSAD) specific mapper. Can tightly integrate the MSAD user account state into the platform account state, including whether the account is enabled, whether the password is expired, and so on. Uses the `userAccountControl` and `pwdLastSet` LDAP attributes.

For example if `pwdLastSet` is 0, the user is required to update their password and there will be an `UPDATE_PASSWORD` required action added to the user. If `userAccountControl` is 514 (disabled account), the platform user is also disabled.

Mapper Configuration Example

To map LDAP group membership to Anaconda Platform roles, use a role mapper.

Add a mapper of the `role-ldap-mapper` type:



In consultation with your LDAP administrator and internal LDAP documentation, define which LDAP group tree will be mapped into roles in the Anaconda Platform realm. The roles are mapped directly by name, so an LDAP membership of `ae-deployer` will map to the role of the same name in Anaconda Platform.

Add user federation mapper

Name *	<input type="text" value="rolemap1"/>
Mapper Type	<input type="text" value="role-ldap-mapper"/>
LDAP Roles DN	<input type="text"/>
Role Name LDAP Attribute	<input type="text" value="cn"/>
Role Object Classes	<input type="text" value="groupOfNames"/>
Membership LDAP Attribute	<input type="text" value="member"/>
Membership Attribute Type	<input type="text" value="DN"/>
Membership User LDAP Attribute	<input type="text" value="uid"/>
LDAP Filter	<input type="text"/>
Mode	<input type="text" value="READ_ONLY"/>
User Roles Retrieve Strategy	<input type="text" value="LOAD_ROLES_BY_MEMBER_ATTRIBUTE"/>
Use Realm Roles Mapping	<input checked="" type="checkbox"/>
Client ID	<input type="text" value="Select One..."/>
<input type="button" value="Save"/> <input type="button" value="Cancel"/>	

Configuring LDAPS (Outbound SSL)

To make correct requests to secure internal resources such as internal enterprise LDAP servers using corporate SSL certificates, you must configure a “trust store”. This is entirely optional. If your internal servers instead use certificates issued by a public root CA, then the default trust store will work fine.

To create a trust store, you must have the public certificates you wish to trust available.

There are two methods described below. Select the option that is right for you.

NOTE: These are certificates for your **trusted server** such as Secure LDAP, not for Anaconda Enterprise.

Option 1

The simplest procedure is if you have the CA certificates available directly. In that case, you can issue:

```
keytool -import -file CAFILE.cert -alias auth -keystore OUTPUT.jks
```

NOTE: Replace `CAFILE.cert` with your CA certificate file, and replace `OUTPUT.jks` with the desired output file name.

Now continue on to *For both methods*.

Option 2

Alternatively, if you have the server certificate and key as well, you can construct a full trust chain in the store.

First, convert the files to PKCS12 format, if they are not in that form already:

```
openssl pkcs12 -export -chain -in CERT.pem -inkey CERT-KEY.pem -out PKCS-CHAIN.p12 -
-name auth -CAfile CA-CHAIN.pem
```

In this example, replace `CERT.pem` with the server’s certificate, `CERT-KEY.pem` with the server’s key, `PKCS-CHAIN.p12` with your desired temporary file name and `CA-CHAIN.pem` with the trust chain file (such as up to the root certificate of your internal CA).

Then, create a Java Keystore to store the trusted certs:

```
keytool -importkeystore -destkeystore OUTPUT.jks -srckeystore PKCS-CHAIN.p12 -alias_  
↪auth
```

Now continue on to *For both methods*.

For both methods

You will be asked to set a password. You must record the password selected.

Edit the platform configuration. Set the `auth.https.truststore` configuration key to the full path to the created `OUTPUT.jks` file. Set `auth.https.truststore-password` to the matching password.

Restart the auth service:

```
sudo gravity enter  
kubectl get pods | grep ap-auth | cut -d' ' -f1 | xargs kubectl delete pods
```

2.1.13 Upgrading Anaconda Enterprise

This page describes how to update your Anaconda Enterprise to its latest version.

NOTE: Before you begin any upgrade, you must *back up* and test your Anaconda Enterprise configuration and data files. Also, ensure all users have saved their work and logged out.

In-place Upgrade Anaconda Enterprise from 5.1.0 to 5.1.2

This is the recommended upgrade path, requiring nearly no downtime.

1. Download the 5.1.2 installer file.
2. Add OS settings required for 5.1.2:

```
sudo sysctl -w fs.may_detach_mounts=1  
sudo sysctl -w net.bridge.bridge-nf-call-iptables=1  
sudo sysctl -w net.ipv4.ip_forward=1
```

Add settings to `/etc/sysctl.conf`:

```
net.ipv4.ip_forward = 1  
net.bridge.bridge-nf-call-iptables = 1  
fs.may_detach_mounts = 1
```

3. Run `sudo ./upgrade`
4. Update the version of the app images in the configmap.

First, edit the configmap:

```
sudo gravity enter  
kubectl edit cm
```

Next, in the configmap, update the app images to the new version of the images in the installer:

```
data:
  anaconda-platform.yml
  images:
    app: apiserver:5000/ap-app:5.1.2-0
    app_proxy: apiserver:5000/ap-app-proxy:5.1.2-0
    editor: apiserver:5000/ap-editor:5.1.2-0
```

- Restart all Anaconda Enterprise pods:

```
kubectl get pods | grep ap- | cut -d' ' -f1 | xargs kubectl delete pods
```

Upgrading Anaconda Enterprise from 5.0.X to 5.1.x

Upgrading your Anaconda Enterprise (AE) installation from version 5.0.x to a newer version of Anaconda Enterprise requires the uninstallation of the current version and installation of the new version of the platform. The steps include backing up data, uninstalling AE, and installing the newer version of AE.

Stage 1 - Backup Anaconda Enterprise

NOTE: All of the following commands should be run on the master node.

- Back up the Anaconda Enterprise configuration:

```
sudo gravity backup anaconda-enterprise-backup.tar.gz
```

- Ensure all users have saved their work and logged out. To prevent any database transactions, stop the AE database with:

```
sudo gravity enter
kubectl delete deploy postgres
```

- Exit the Anaconda Enterprise environment by typing `exit`.
- All of the persistent data in AE is stored on the master node in `/opt/anaconda/storage`, you can backup your data by running the following command:

```
sudo tar -zcvf anaconda-data.tar.gz /opt/anaconda/
```

- Restart the AE database:

```
sudo gravity enter

kubectl apply -f /var/lib/gravity/local/packages/unpacked/gravitational.io/
↔AnacondaEnterprise/*/resources/postgres.yaml

# Restart service pods
kubectl get pods | grep ap- | cut -d' ' -f1 | xargs kubectl delete pods
```

- Exit the Anaconda Enterprise environment by typing `exit`.

Stage 2 - Uninstall Anaconda Enterprise

- To uninstall Anaconda Enterprise on a healthy master node, run:

```
sudo gravity system uninstall
sudo killall gravity
sudo killall planet
sudo rm -rf /var/lib/gravity
```

If `/var/lib/gravity` is present after the uninstallation, you should reboot your machine and retry the `sudo gravity system uninstall` command.

2. Reboot.

This ensures that any Anaconda Enterprise state is flushed from your system.

Stage 3 - Install Anaconda Enterprise

1. Download the installer file for the newer AE version that you want to upgrade to.
2. Follow the [installation instructions](#) to install the newer version of Anaconda Enterprise, which will use the existing data in `/opt/anaconda`.
3. Update the Anaconda Enterprise configuration to match the latest [configuration schema](#). Note that we do not currently version the schema of the `anaconda-platform.yml`, so there may be breaking changes between versions.

Check the logs for each service for errors about new or missing fields. If you see any errors, manually update the configuration to match the new schema.

Significant known schema changes, with the version they were added in, are detailed below:

5.1.x

The field format for specifying passive license information has changed. The field `license.client-id` is now `license.number`, and the field `license.client-certificate` is now `license.key`.

4. Ensure that your SSL certificate filenames are correct.

In Anaconda Enterprise 5.1.0 and newer, the default SSL certificate filenames provided by the installer are different than in previous versions. It is recommended that you update any Kubernetes secrets you created and update the Anaconda Enterprise configuration to match the new filenames.

Previous	Updated
<code>rootca.pem</code>	<code>rootca.crt</code>
<code>cert.pem</code>	<code>server.crt</code>
<code>privkey.pem</code>	<code>server.key</code>
<code>tls.crt</code>	<code>wildcard.crt</code>
<code>tls.key</code>	<code>wildcard.key</code>

NOTE: the `keystore.jks` filename is unchanged.

5. Add roles and associate them with the appropriate users (if upgrading from 5.0.x):

```
ae-admin
ae-creator
ae-deployer
ae-uploader
```

6. Restart all Anaconda Enterprise pods:

```
kubectl get pods | grep ap- | cut -d' ' -f1 | xargs kubectl delete pods
```

Troubleshooting

In-place upgrades from a version other than 5.1.0 to 5.1.2

If an attempt was made to perform an in-place upgrade from a version other than 5.1.0 to 5.1.2, the service pods will be in the *ImagePullBackOff* state. To recover, execute the following command with the correct original version:

```
kubectl get deployments -n default -o yaml | sed "s/:original-version/:5.1.2-0/g\" | \
↪ kubectl replace -f - && kubectl get pods -n default | grep ap- | cut -d' ' -f1 | \
↪ xargs kubectl delete pods -n default
```

2.1.14 Uninstall

Use the following instructions to uninstall Anaconda Enterprise.

To uninstall Anaconda Enterprise on a healthy cluster worker nodes, run:

```
sudo gravity leave --force
sudo killall gravity
sudo killall planet
```

To uninstall Anaconda Enterprise on a healthy cluster master node, run:

```
sudo gravity system uninstall
sudo killall gravity
sudo killall planet
sudo rm -rf /var/lib/gravity /opt/anaconda
```

To uninstall a failed or faulty cluster node, run:

```
sudo gravity remove --force
```

To remove an offline node that cannot be reached from the cluster, run:

```
sudo gravity remove <node>
```

<node> - specifies the node to be removed and can be either the node's assigned hostname or its IP address (the one that was used as an "advertise address" or "peer address" during install) or its Kubernetes name (can be obtained via `kubectl get nodes`).

2.2 User management

This section describes the Anaconda Enterprise administration functions for creating and managing user accounts.

User management includes creating or removing users and roles, changing passwords, requiring terms and conditions to be accepted on registration, allowing users to self-register and more.

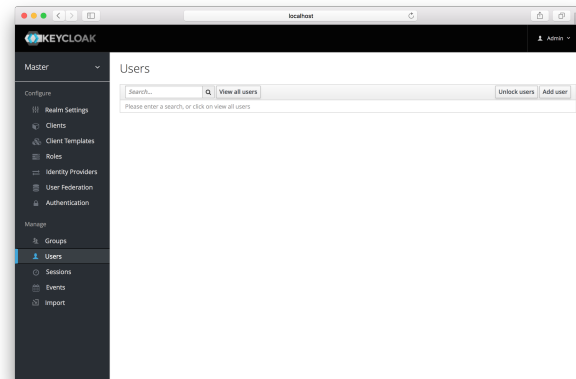


Fig. 1: Users

2.2.1 Searching For Users

If you need to manage a specific user, click on `Users` in the left menu bar.

This menu option brings you to the user list page. In the search box you can type in a full name, last name, or email address you want to search for in the user database. The query will bring up all users that match your criteria. The `View all users` button will list every user in the system. This will search just the local user database and not the federated database (such as LDAP) because some backends like LDAP don't have a way to page through users. So if you want the users from federated backend to be synced into the local database you need to either:

- Adjust search criteria. That will sync just the backend users matching the criteria into the local database.
- Go to `User Federation` tab and click `Sync all users` or `Sync changed users` in the page with your federation provider.

2.2.2 Creating New Users

To create a user click on `Users` in the left menu bar.

This menu option brings you to the user list page. On the right side of the empty user list, you should see an `Add User` button. Click that to start creating your new user.

The only required field is `Username`. Click `save`. This will bring you to the management page for your new user.

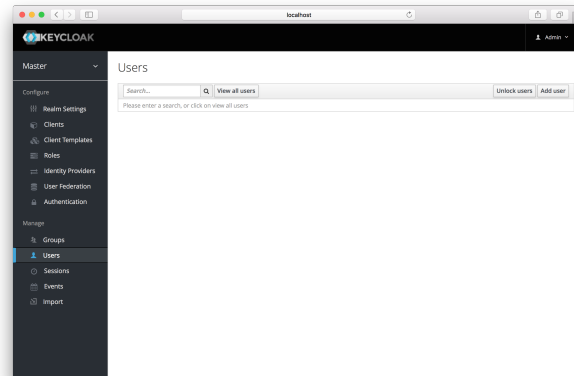


Fig. 2: Users

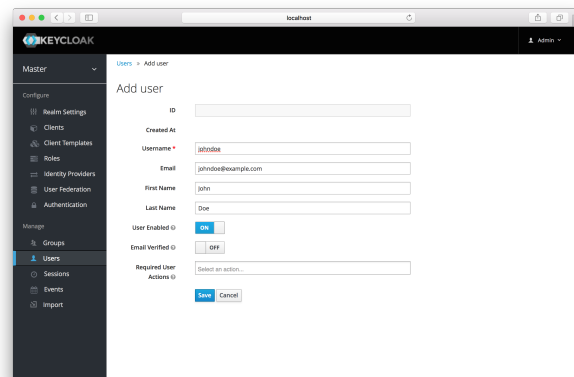


Fig. 3: Add User

2.2.3 User Attributes

Beyond basic user metadata like name and email, you can store arbitrary user attributes. Choose a user to manage then click on the `Attributes` tab.

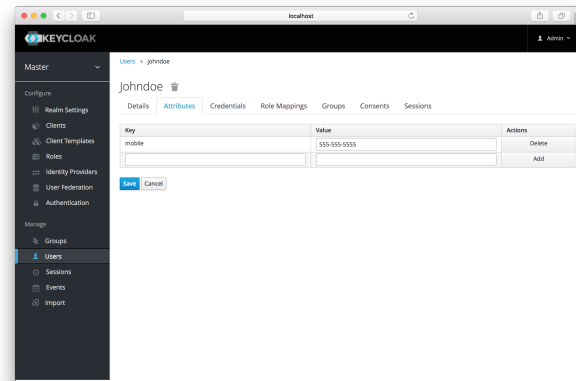


Fig. 4: Users

Enter in the attribute name and value in the empty fields and click the `Add` button next to it to add a new field. Note that any edits you make on this page will not be stored until you hit the `Save` button.

2.2.4 Roles

Roles identify a type or category of user. Typical roles that may exist in an organization include `admin`, `user`, `manager`, and `employee`.

Assigning access and permissions to individual users can be too fine-grained and difficult to manage, so Anaconda Enterprise recommends assigning access and permissions to roles. There is a global namespace for roles, and each “client” (that is, service) also has its own dedicated namespace where roles can be defined.

Pre-defined Roles

Anaconda Enterprise comes with a set of roles that provide users access within the system. It’s a good practice to use these roles to assign access where necessary. See [User Role Mappings](#) for how to do this.

The built-in roles are:

ae-admin Allows a user to act as an administrator for the platform, exposing administrative settings in the UI and access to the authentication Admin Console.

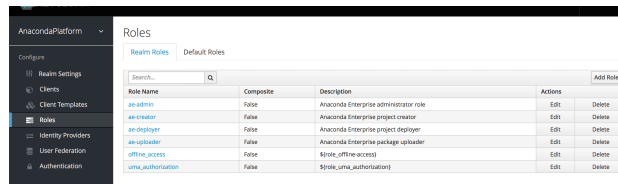
ae-creator Allows a user to create new Projects.

ae-deployer Allows a user to create new Deployments from Projects.

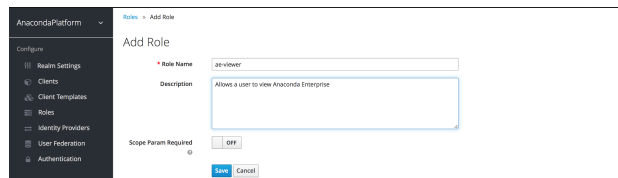
ae-uploader Allows a user to upload Projects and Packages.

Realm Roles

Realm-level roles are a global namespace. You can see the list of built-in and created roles by clicking the Roles left menu item in the Admin Console.



To create a role, click Add Role on this page, enter the name and description of the role, and click Save.

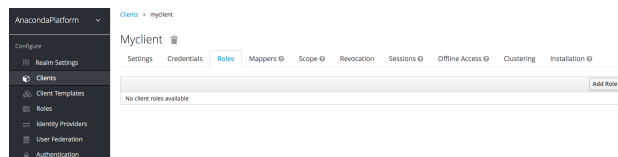


Roles can be assigned to a client automatically or require an explicit request. If the client has to explicitly request a realm role, set the Scope Param Required switch to On. The role must then be specified using the `scope` parameter when requesting a token.

Multiple realm roles are separated by a space. EXAMPLE: `scope=admin user`

Client Roles

Client roles are little more than a namespace for each client (that is, component of Anaconda Enterprise). Each client has its own namespace. Client roles are managed under the Roles tab under each individual client.



If the client has to explicitly request another client's role, the role must be prefixed with the client ID when performing a request using the scope parameter. EXAMPLE: If the client ID is `account` and the role is `admin`, the scope parameter is `scope=account/admin`.

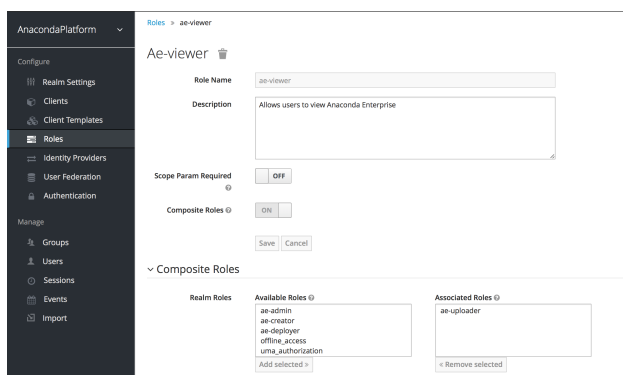
As with requesting realm roles, multiple roles are separated by spaces.

Composite Roles

Any realm or client level role can be turned into a composite role, a role that has one or more additional roles associated with it. When a composite role is mapped to the user, the user also gains the roles associated with that composite. This inheritance is recursive, so any composite of composites is also inherited.

To turn a regular role into a composite role, go to the role detail page and flip on the Composite Role switch.

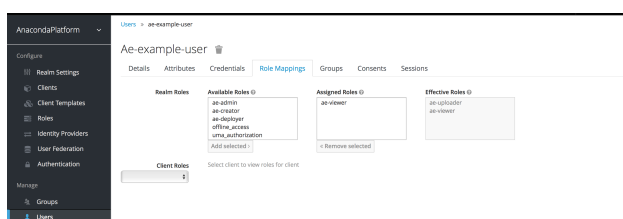
The role selection UI will display lower on the page and you'll be able to associate realm level and client level roles to the composite you are creating. In this example, the `ae-uploader` realm-level role was associated with the `ae-viewer` composite role. Any user with the `ae-viewer` role will now also inherit the `ae-uploader` role.



NOTE: When tokens and SAML assertions are created, any composite will also have its associated roles added to the claims and assertions of the authentication response sent back to the client. See *Client Scope* for more detail.

User Role Mappings

User role mappings can be assigned individually to each user through the Role Mappings tab for that single user.



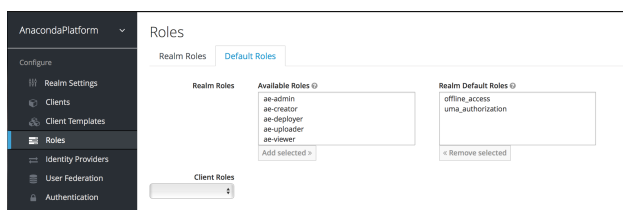
In the above example, we have assigned to the example user the `ae-viewer` role we created in the *Composite Roles* section.

The Effective Roles list on the right includes all roles that are explicitly assigned to the user and all roles that are inherited from composites. In this example, note that the `ae-uploader` role that is associated with the `ae-viewer` composite is shown in the Effective Roles list.

Default Roles

Default roles allow you to automatically assign user role mappings when any user is newly created or imported (for example, through *LDAP*).

To specify default roles go to the Roles left menu item and click the Default Roles tab.



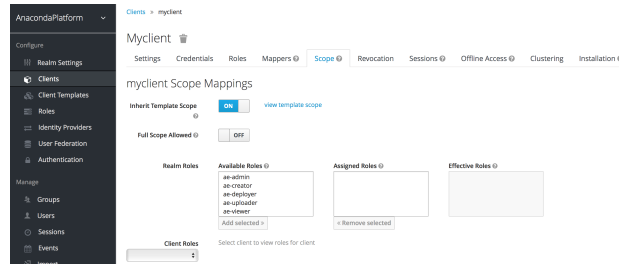
Client Scope

When an OIDC access token or SAML assertion is created, all the user role mappings of the user are, by default, added as claims within the token or assertion. Applications use this information to make access decisions on the resources

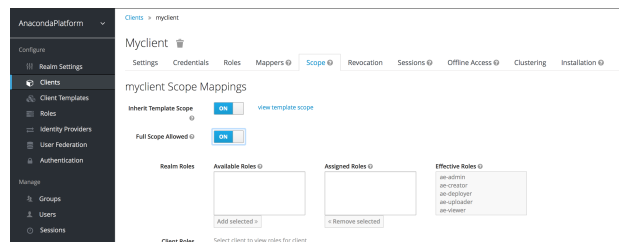
controlled by that application. In Anaconda Enterprise, access tokens are digitally signed and the application can re-use tokens and invoke them on other remotely secured REST services. This means that if an application is compromised or a rogue client is registered with the realm, attackers can get access tokens that have a broad range of permissions and your whole network can be compromised. This is where *client scope* becomes important.

Client scope is a way to limit the roles that are declared inside an access token. When a client requests that a user be authenticated, the access token they receive back will only contain the role mappings you’ve explicitly specified for the client’s scope. This allows you to limit the permissions each individual access token has rather than giving the client access to all of the user’s permissions. By default, each client gets all the role mappings of the user.

You can view this in the Scope tab of each client.



Note that it’s also possible to enable a client to gain all the available roles in the realm, by turning on the “Full Scope Allowed” switch. With this active, you can see that the effective roles of the scope include every declared role in the realm.



2.2.5 Groups

Groups in Anaconda Enterprise allow you to manage a common set of attributes and role mappings for a set of users. Users can be members of zero or more groups. Users inherit the attributes and role mappings assigned to each group. To manage groups go to the Groups left menu item.

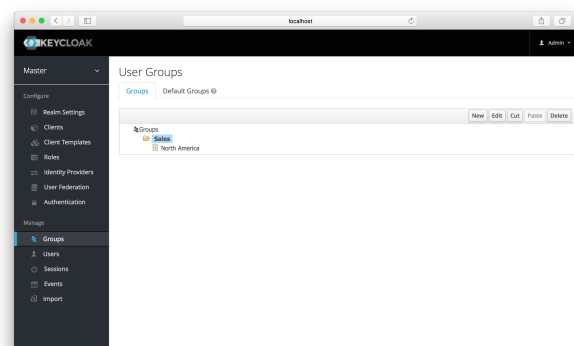


Fig. 5: Groups

Groups are hierarchical. A group can have many subgroups, but a group can only have one parent. Subgroups inherit the attributes and role mappings from the parent. This applies to the user as well. So, if you have a parent group and a child group and a user that only belongs to the child group, the user inherits the attributes and role mappings of both the parent and child. In this example, we have a top level `Sales` group and a child `North America` subgroup. To add a group, click on the parent you want to add a new child to and click `New` button. Select the `Groups` icon in the tree to make a top-level group. Entering in a group name in the `Create Group` screen and hitting `Save` will bring you to the individual group management page.

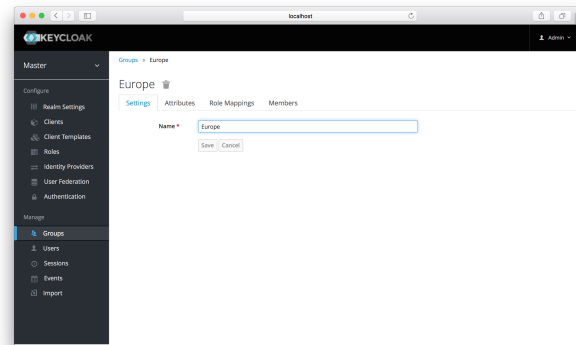


Fig. 6: Group

The `Attributes` and `Role Mappings` tab work exactly as the tabs with similar names under a user. Any attributes and role mappings you define will be inherited by the groups and users that are members of this group.

To add a user to a group you need to go all the way back to the user detail page and click on the `Groups` tab there.

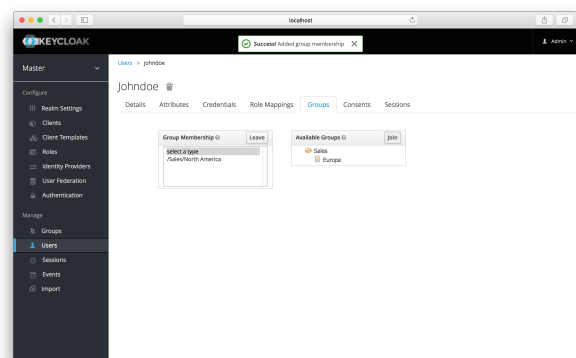


Fig. 7: User Groups

Select a group from the `Available Groups` tree and hit the `join` button to add the user to a group. Vice versa to remove a group. Here we've added the user *Jim* to the *North America* sales group. If you go back to the detail page for that group and select the `Membership` tab, *Jim* is now displayed there.

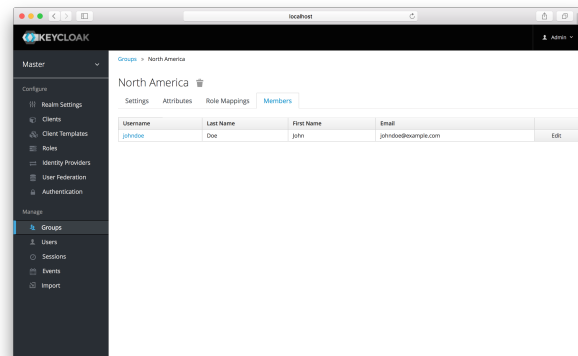


Fig. 8: Group Membership

NOTE: When you set a group to be a default group, the change is applied only to users created after that action. To apply the change to all federated users, remove them by clicking the `Remove imported` button, then replace them by clicking the `Synchronize all users` button.

Groups vs. Roles

In the IT world the concepts of Group and Role are often blurred and interchangeable. In Anaconda Enterprise, Groups are just a collection of users that you can apply roles and attributes to in one place. Roles define a type of user and applications assign permission and access control to roles.

Aren't *Composite Roles* also similar to Groups? Logically they provide the same exact functionality, but the difference is conceptual. Composite roles should be used to apply the permission model to your set of services and applications. Groups should focus on collections of users and their roles in your organization. Use groups to manage users. Use composite roles to manage applications and services.

Default Groups

Default groups allow you to automatically assign group membership whenever any new user is created or imported through User Storage Federation or Identity Brokering. To specify default groups go to the `Groups` left menu item, and click the `Default Groups` tab.

2.2.6 User Credentials

When viewing a user if you go to the `Credentials` tab you can manage a user's credentials.

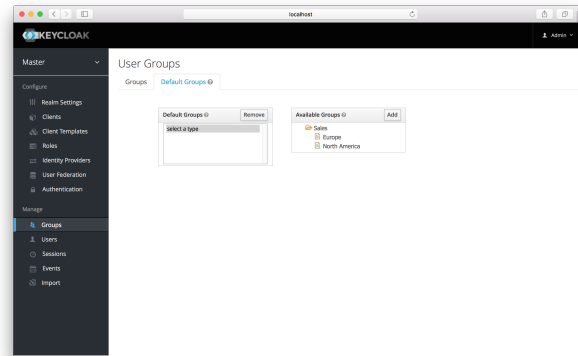


Fig. 9: Default Groups

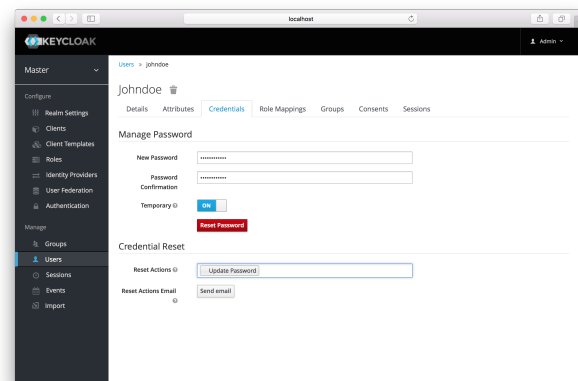


Fig. 10: Credential Management

Changing Passwords

To change a user's password, type in a new one. A `Reset Password` button will show up that you click after you've typed everything in. If the `Temporary` switch is on, this new password can only be used once and the user will be asked to change their password after they have logged in.

Alternatively, if you have email set up, you can send an email to the user that asks them to reset their password. Choose `Update Password` from the `Reset Actions` list box and click `Send Email`. You can optionally set the validity of the e-mail link which defaults to the one preset in `Tokens` tab in the realm settings. The sent email contains a link that will bring the user to the update password screen.

Changing OTPs

You cannot configure One-Time Passwords for a specific user within the Admin Console. This is the responsibility of the user. If the user has lost their OTP generator all you can do is disable OTP for them on the `Credentials` tab. If OTP is optional in your realm, the user will have to go to the User Account Management service to re-configure a new OTP generator. If OTP is required, then the user will be asked to re-configure a new OTP generator when they log in.

Like passwords, you can alternatively send an email to the user that will ask them to reset their OTP generator. Choose `Configure OTP` in the `Reset Actions` list box and click the `Send Email` button. The sent email contains a link that will bring the user to the OTP setup screen.

2.2.7 Required Actions

Required Actions are tasks that a user must finish before they are allowed to log in. A user must provide their credentials before required actions are executed. Once a required action is completed, the user will not have to perform the action again. Here are an explanation of some of the built-in required action types:

Update Password When set, a user must change their password.

Configure OTP When set, a user must configure a one-time password generator on their mobile device using either the Free OTP or Google Authenticator application.

Verify Email When set, a user must verify that they have a valid email account. An email will be sent to the user with a link they have to click. Once this workflow is successfully completed, they will be allowed to log in.

Update Profile This required action asks the user to update their profile information, i.e. their name, address, email, and/or phone number.

Admins can add required actions for each individual user within the user's `Details` tab in the Admin Console.

In the `Required User Actions` list box, select all the actions you want to add to the account. If you want to remove one, click the X next to the action name. Also remember to click the `Save` button after you've decided what actions to add.

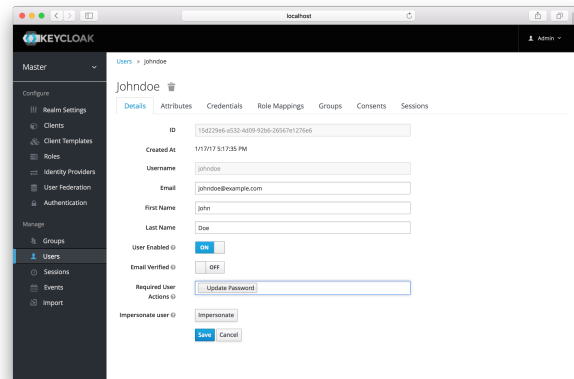


Fig. 11: Setting Required Action

Default Required Actions

You can also specify required actions that will be added to an account whenever a new user is created with the **Add User** button on the user list screen or the [user registration](#) link on the login page. To specify the default required actions go to the **Authentication** left menu item and click on the **Required Actions** tab.

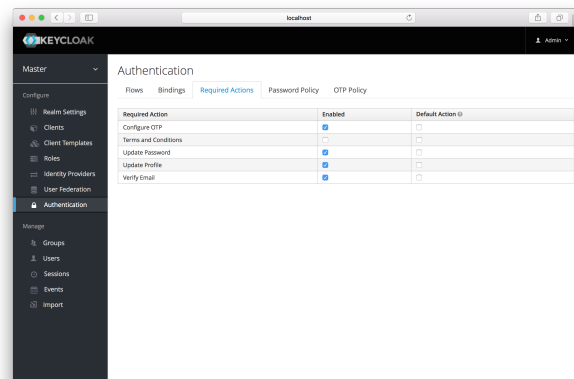


Fig. 12: Default Required Actions

Simply click the checkbox in the **Default Action** column of the required actions that you want to be executed when a brand new user logs in.

Terms and Conditions

Many organizations have a requirement that when a new user logs in for the first time, they need to agree to the terms and conditions of the website. This functionality can be implemented as a required action, but it requires some configuration. For one, you have to go to the **Required Actions** tab described earlier and enable the **Terms**

and `Conditions` action. You must also edit the *terms.ftl* file in the *base* login theme. See the [Server Developer Guide](#) for more information on extending and creating themes.

2.2.8 Impersonation

It is often useful for an admin to impersonate a user. For example, a user may be experiencing a bug in one of your applications and an admin may want to impersonate the user to see if they can duplicate the problem. Admins with the appropriate permission can impersonate a user. There are two locations an admin can initiate impersonation. The first is on the `Users` list tab.

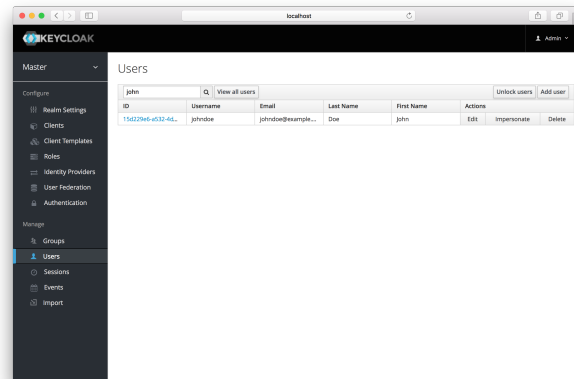


Fig. 13: Users

You can see here that the admin has searched for `jim`. Next to Jim's account you can see an impersonate button. Click that to impersonate the user.

Also, you can impersonate the user from the user `Details` tab.

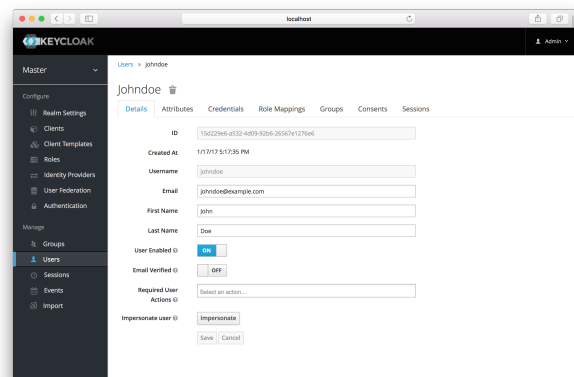


Fig. 14: User Details

Near the bottom of the page you can see the `Impersonate` button. Click that to impersonate the user.

When impersonating, if the admin and the user are in the same realm, then the admin will be logged out and automatically logged in as the user being impersonated. If the admin and user are not in the same realm, the admin will remain logged in, but additionally be logged in as the user in that user's realm. In both cases, the browser will be redirected to the impersonated user's User Account Management page.

Any user with the realm's `impersonation` role can impersonate a user.

2.2.9 User Registration

You can enable user self registration in the authentication center. When enabled, the login page has a registration link the user can click on to create their new account. Enabling registration is pretty simple. Go to the `Realm Settings` left menu and click it. Then go to the `Login` tab. There is a `User Registration` switch on this tab. Turn it on, then click the `Save` button.

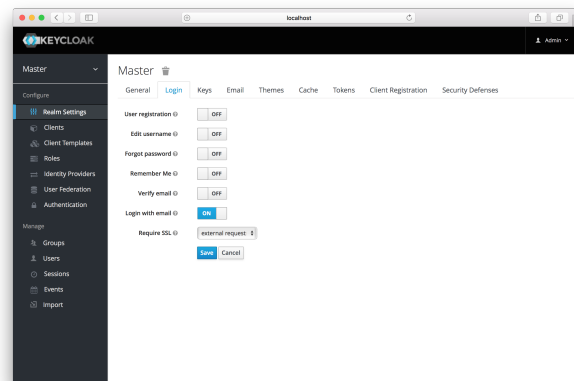


Fig. 15: Login Tab

After you enable this setting, a `Register` link should show up on the login page.

Clicking on this link will bring the user to the registration page where they have to enter in some user profile information and a new password.

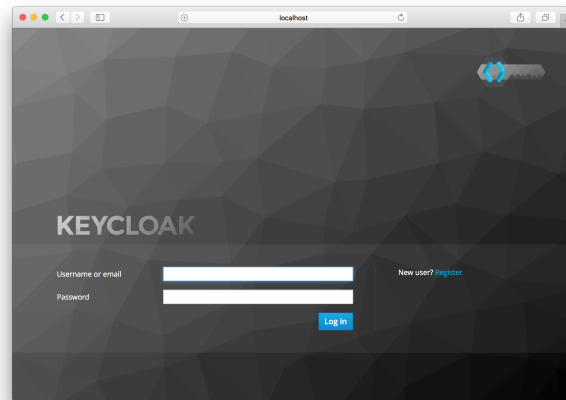


Fig. 16: Registration Link

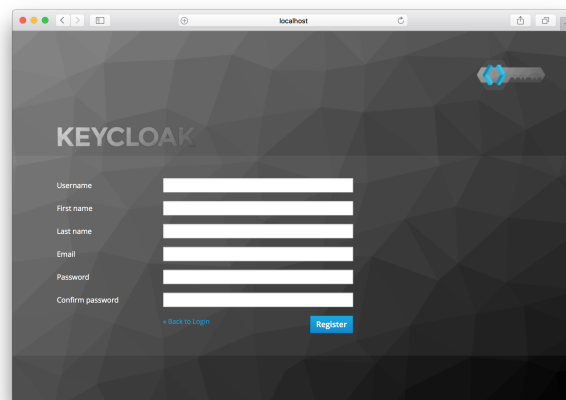


Fig. 17: Registration Form

You can change the look and feel of the registration form as well as removing or adding additional fields that must be entered. See the [Server Developer Guide](#) for more information.

2.3 System management

This section describes the Anaconda Enterprise administration functions for managing the system.

System management includes using the Operations Center, configuring SSL, creating custom environments and installers, backup and restore and recovery after master node failure.

2.3.1 Using the Operations Center

The Operations Center is a web-based administrative dashboard where IT administrators can monitor and manage many of the routine operations for Anaconda Enterprise.

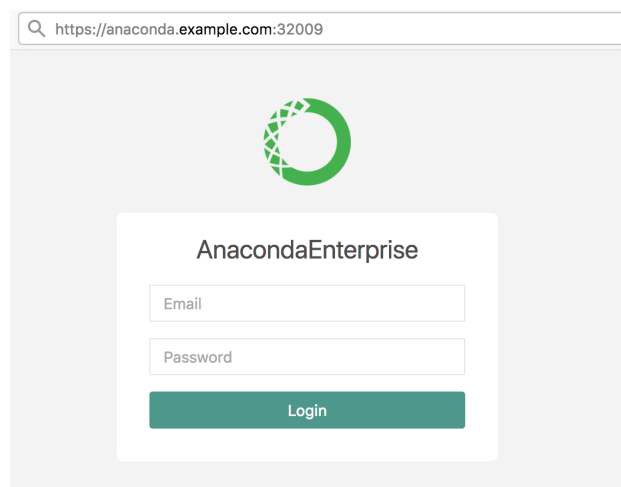
Logging into the Operations Center

Access the Anaconda Enterprise Operations Center by visiting the following URL in your browser:

`https://anaconda.example.com:32009`

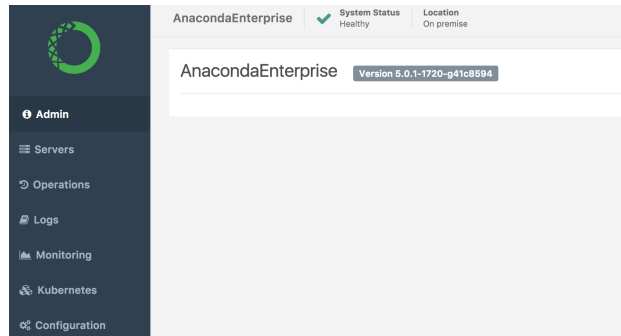
NOTE: Replace `anaconda.example.com` with the domain name you are using.

Log in with your Operations Center username and password:



View Anaconda Enterprise info

To view the Anaconda Enterprise version number or download a debugging report, open the Operations Center, then from the left navigation click the Admin link.



View version number

Your Anaconda Enterprise version number is displayed in the body of the page.

Download debug report

Click the Download Debug Info button at the top right.

The report downloads to your computer in .tar.gz format.

Add or manage servers

You can view, add, edit and delete servers from Anaconda Enterprise from the Operations Center's Servers menu.

To manage the servers on your system, open the Operations Center, then from the left navigation click the Servers link.

The IP address, public IP, Hostname and profile of each node on your system is identified.

 The screenshot shows the 'Servers' page in the Anaconda Enterprise interface. It features a table with columns for Private IP, Public IP, Hostname, and Profile. Two nodes are listed: a worker node and a master node.

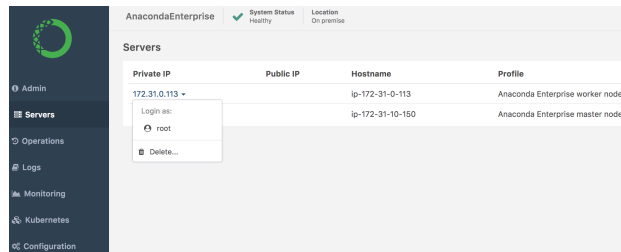
Private IP	Public IP	Hostname	Profile
172.31.0.113		ip-172-31-0-113	Anaconda Enterprise worker node(s)
172.31.10.150		ip-172-31-10-150	Anaconda Enterprise master node

View existing servers

A list of worker and master node servers is displayed in the body of the page.

Modify existing servers

You can log onto any existing server on the system and make modifications. Select the IP address of the server you want to modify, and click Log on as Root.

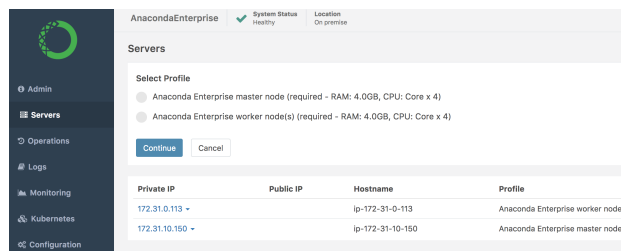


When you are finished, close the console by clicking the X icon.

Add existing servers

Add existing servers by clicking the Add Existing button at the top right.

You will be prompted to select a profile of a master node or worker node. Choose one, then click the Continue button.



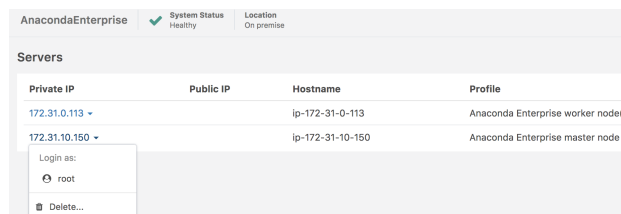
A pane opens containing the command you need to add the server.

Open a terminal window, then copy and paste the command into terminal.

When you refresh the page, your server will automatically appear in the list.

Delete existing server

Delete an existing server by selecting the IP address of the server you want to delete. Then from the drop-down menu that appears, select Delete.



Monitor operations

You can monitor the status or progress of a cluster installation from the Operations Center.

Open the Operations Center, then from the left navigation click the Operations link.

A list of current operations and their status displays.

AnacondaEnterprise	System Status Healthy	Location On premise
Operations		
Type	Started	Status
Installing this cluster	8/3/2017 3:38:31 PM	Completed

Click an operation to select and monitor it, or click the Logs button at the far right of an operation to view its log.

View system logs

You can view and filter system logs from the Operations Center.

To view logs, open the Operations Center, then from the left navigation click the Logs link.

The complete system log displays.

Logs	Filter	Refresh
<pre> 2017-08-04 19:46:17.849 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.tools.notebook.packages=["f 2017-08-04 19:46:17.849 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.tools.notebook.defaultTru 2017-08-04 19:46:17.849 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.templates.jupyterlab.label 2017-08-04 19:46:17.849 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.templates.jupyterlab.too 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.templates.jupyterlab.defau 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.templates.jupyter-5.label 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-spaces.spaces.options.spaces.templates.jupyter-5.tool 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-deploy.deploy.icon="fa-anaconda" [was fa-anaconda] [ana 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-deploy.deploy.label="Deploy" [was Deploy] [anaconda-pla 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-deploy.deploy.url="https://anaconda.example.com:38081/ap 2017-08-04 19:46:17.850 UTC INFO : ui.services.github.github.com.icon="fa-github" [was fa-github] [anaconda-pla 2017-08-04 19:46:17.850 UTC INFO : ui.services.github.github.com.label="Github" [was Github] [anaconda-pla 2017-08-04 19:46:17.850 UTC INFO : ui.services.github.github.com.url="https://api.github.com" [was https://api.g 2017-08-04 19:46:17.850 UTC INFO : ui.services.github.github.com.html.url="https://github.com" [was https://gith 2017-08-04 19:46:17.850 UTC INFO : ui.services.github.github.com.disabled=True [was True] [anaconda-platform.co 2017-08-04 19:46:17.850 UTC INFO : ui.services.auth-api.auth-api.icon="fa-anaconda" [was fa-anaconda] [anaconda 2017-08-04 19:46:17.850 UTC INFO : ui.services.auth-api.auth-api.label="Auth API" [was Auth API] [anaconda-plat 2017-08-04 19:46:17.850 UTC INFO : ui.services.auth-api.auth-api.url="https://anaconda.example.com:38082/api/v1 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-storage.storage.icon="fa-anaconda" [was fa-anaconda] [an 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-storage.storage.label="Storage" [was Storage] [anaconda 2017-08-04 19:46:17.850 UTC INFO : ui.services.anaconda-storage.storage.url="https://anaconda.example.com:38086 2017-08-04 19:46:17.850 UTC INFO : ui.services.documentation.offline_docs.icon="fa-anaconda" [was fa-anaconda] 2017-08-04 19:46:17.850 UTC INFO : ui.services.documentation.offline_docs.label="Documentation" [was Documentat 2017-08-04 19:46:17.851 UTC INFO : ui.services.documentation.offline_docs.url="https://anaconda.example.com:380 2017-08-04 19:46:17.851 UTC INFO : ui.services.documentation.offline_docs.html.url="https://anaconda.example.com 2017-08-04 19:46:17.851 UTC INFO : ui.services.anaconda-enterprise-notebooks.aen.icon="fa-anaconda" [was fa-ana 2017-08-04 19:46:17.851 UTC INFO : ui.services.anaconda-enterprise-notebooks.aen.html.url="https://notebooks.ex 2017-08-04 19:46:17.851 UTC INFO : ui.services.anaconda-enterprise-notebooks.aen.disabled=True [was True] [anac 2017-08-04 19:46:17.851 UTC INFO : ui.auth-server.client-secret [value hidden] [anaconda-platform.common.config 2017-08-04 19:46:17.851 UTC INFO : ui.port6908 [was 6908] [anaconda-platform.common.config] 2017-08-04 19:46:17.851 UTC INFO : ui.base-url="/" [was /] [anaconda-platform.common.config] 2017-08-04 19:46:17.851 UTC INFO : ui.public-url="https://anaconda.example.com:38089/" [was https://ui.anaconda 2017-08-04 19:46:17.851 UTC INFO : ui.is.application.analytics.hotjar.hide# [was 0] [anaconda-platform.common.c 2017-08-04 19:46:17.851 UTC INFO : ui.is.application.analytics.hotjar.baseUrl# [was 0] [anaconda-platform.common.c 2017-08-04 19:46:17.851 UTC INFO : ui.is.application.analytics.hotjar.hisv# [was 0] [anaconda-platform.common.c 2017-08-04 19:46:17.851 UTC INFO : ui.cookie-next.name="anaconda-platform-ui-next-v2" [was anaconda-platform-ui 2017-08-04 19:46:17.851 UTC INFO : ui.help.docs.label="Anaconda Documentation - Home" [was Anaconda Documentatio 2017-08-04 19:46:17.851 UTC INFO : ui.help.docs.position# [was 0] [anaconda-platform.common.config] 2017-08-04 19:46:17.852 UTC INFO : ui.help.docs.href="https://anaconda.example.com:38071" [was https://docs.ana </pre>		

Filter system logs

You can filter the logs for a specific container or pod.

Click the Search box at the top of the page, type the name or ID of the container or pod you want to view as pod: <pod-id> and click the Refresh button.

EXAMPLE: `pod:ap-spaces-3430848090-m3k9j`

You may also see the logs for a pod by clicking Kubernetes on the left menu, clicking Pods, clicking the name of a pod, and clicking Logs.

Monitor cluster usage

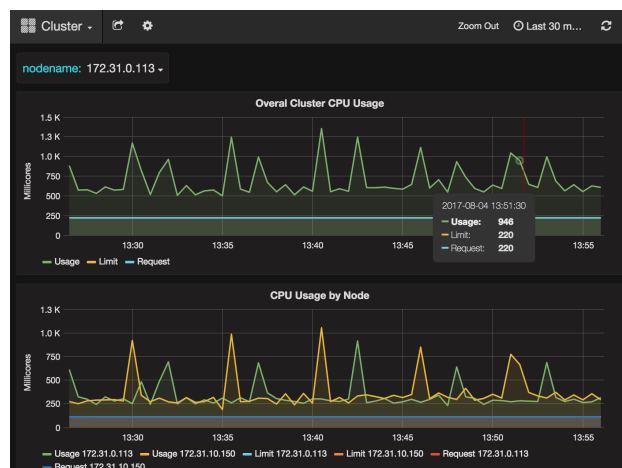
View, monitor and share the status of your clusters, pods and individual nodes from the Anaconda Enterprise Operations Center. The graphs include:

- Overall cluster CPU usage
- CPU usage by node
- Individual CPU usage
- Overall cluster memory usage
- Memory usage by node
- Individual node memory usage
- Overall cluster network usage
- Network usage by node
- Individual node network usage
- Overall cluster filesystem usage
- Filesystem usage by node
- Individual filesystem usage

All graphs can be modified by time range. The default time range is the previous thirty minutes.

View graphs

To view any or all of these graphs, open the Operations Center, then from the left navigation click the Monitoring link. The graphs listed above all display.



Select cluster, pod or node

Select the cluster, pod and/or node you want to monitor by clicking the “Cluster” link at the top of the page, then from the drop-down menu that appears select the cluster or Pod. You can then optionally select a specific nodename to monitor.

Modify time range displayed

Click an operation to select and monitor it, or click the Logs button at the far right of an operation to view its log.

Share monitoring graphs

Click the “Share” icon at the top to get a unique URL that can be shared. The link contains an access token so the viewer does not need a password.

You can also share a snapshot that strips out sensitive data, leaving only the visible metric data and series names embedded into your dashboard.

After clicking the “Share” icon, from the drop-down menu select Snapshot sharing. Then click the Local Snapshot button to save the image locally, or the Publish to Snapshot button to save the image to the cloud.

The cloud snapshot can be manipulated by the viewer with zoom and time range changes. The viewer cannot change the node, cluster or pod.

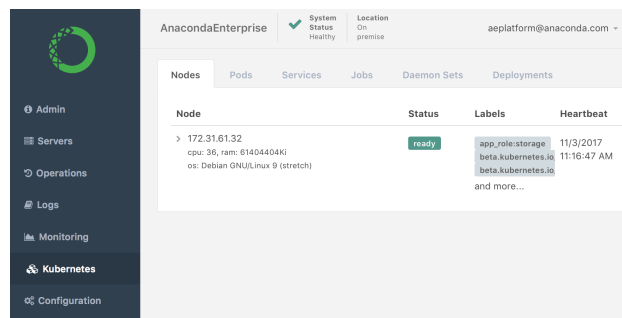
NOTE: Anyone who has the link and can reach the URL can view the report.

Monitor Kubernetes

View and monitor the status of your Kubernetes nodes, pods, services, jobs, daemon sets and deployments from the Operations Center.

Open the Operations Center, then from the left navigation click the Kubernetes link.

From the top tabs, select the item you wish to view.



Nodes

The Node list includes the current status (Running, Stopped or Ready), the name of the label, and last time accessed. Click any node name to see its underlying code.

Pods

Click the name of any pod for more information.

Quickly locate any item by typing a few letters or numbers of its name, container name, pod IP address or host IP address in the top right Search box.

You can also search in the same Search box by status Running or Stopped.

Services

Click the name of any service for more information and to see its underlying code.

Jobs

See a list of jobs by name, desired, status or age.

Daemon Sets

See list of daemon sets by name, desired, current, ready, missed or age.

Deployments

See list of deployments by name, desired, current, up-to-date, available, and age.

Anaconda Enterprise configuration (YAML file)

The Anaconda Enterprise configuration YAML file, `anaconda-platform.yml`, contains both global and per-service configuration settings for your Anaconda Enterprise installation. You can edit these directly from the Ops Center.

This `yaml-format` file contains the locations and specifications for the following:

- Admin
- Authentication
- Authentication escrow
- Authentication server
- Conda channels and defaults
- Database
- Deploy
- Git
- HTTPS
- Images
- Kubernetes
- License

- Offline docs
- PostgreSQL
- Repository
- S3
- S3 Client
- Spaces
- Storage
- Sync
- UI

Download a `sample` `anaconda-platform` configuration file.

To edit your `anaconda-platform.yml` file, from the Ops Center’s left navigation, click the Configuration link. In the box named Config Maps, select the file `anaconda-platform.yml`. Edit as desired.

Edits are made directly to the configuration file.

After editing, at the bottom left an Apply button appears. To save your work, click the Apply button.

If you attempt to abandon your edits by navigating away from the page, you will see a warning “You have unsaved changes!” You can choose to navigate away with the “Disregard and continue” button, or return to your editing with the “Close” button.

After making changes, restart the appropriate service so the changes will take effect. You can restart all services with these commands:

```
sudo gravity enter
kubectl get pods | grep ap- | cut -d' ' -f1 | xargs kubectl delete pods
```

View or edit configuration

You can view or edit your configuration files directly from the Operations Center.

Open the Operations Center, then from the left navigation click the Configuration link.

From the top right “Config Maps” box, select the name of the configuration file that you want to view or edit.

The default configuration file `anaconda-platform.yml` is displayed.

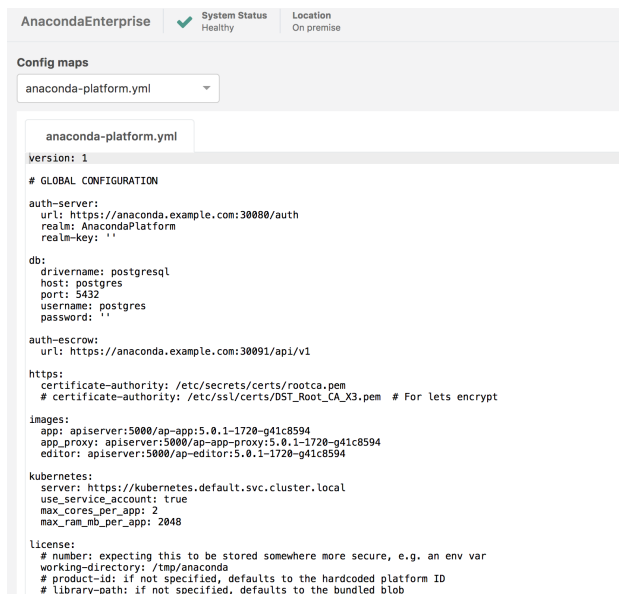
All the configuration of Anaconda Enterprise and the underlying Kubernetes system can be accessed and changed in this view.

Anaconda Enterprise configuration is available in the “default” Namespace:

- `anaconda-platform.yml`– main Anaconda Enterprise configuration file
- `nginx-config`– modify configuration of nginx web server

Kubernetes specific configuration is available in the “kube-system” Namespace:

- `alerting-addresses`–modify email alerts
- `dashboard-default`–modify default dashboard configuration



- extension-apiserver-authentication— modify client-ca-file
- grafana—modify grafana.ini
- gravity-opscenter—modify gravity.yaml, proxy.cert, proxy.key or teleport.yaml

Edit configuration file

NOTE: Edits are made directly to the configuration file. We strongly recommend that you copy and save a copy of the original file before making any edits.

To edit any of the above configuration files, after selecting the file from the “Config Maps” box, begin typing in the in-page text editor.

After editing, at the bottom left an Apply button appears. Click the Apply button to save your work.

Operations Center settings

The Anaconda Enterprise Settings menu includes managing your account, creating and managing users and roles, forwarding logs, setting and monitoring metrics retention periods, and viewing or changing the TLS/SSL certificates for the Operations Center web endpoint.

My account

You can change your Admin user account password for the Operations Center from the Settings menu.

To change your password from any page in the Operations Center, at the top right next to your username, click the triangle icon and from the drop-down menu select Settings.

This brings you to the Change Password page.

The screenshot shows the 'Telekube Cluster / Settings' page. On the left, there is a sidebar with 'User and Groups' selected, containing links for 'My Account', 'Auth', 'Users', and 'Roles'. Below this is a 'Settings' section with links for 'Logs', 'Monitoring', and 'HTTPS Certificate'. The main content area is titled 'Change Password' and contains three input fields: 'Current Password' (with placeholder 'Current'), 'New Password' (with placeholder 'New'), and 'Confirm Password' (with placeholder 'Confirm New'). An 'Update' button is located below the input fields.

Enter your current password, then enter a new password and confirm by entering the new password again.

Click the Update button.

Manage Operations Center users

From the Users Settings menu, you can:

- View a list of all Operations Center users
- Add a new Operations Center user
- Resend an Operations Center user invitation
- Revoke an invitation
- Edit an Operations Center user
- Delete an Operations Center user

To manage Operations Center users, from any page of the Operations Center open the Settings menu by clicking the icon next to your username, then from the drop-down menu that appears, select Settings.

Then from the Settings left navigation menu, click the Users link.

View list of Operations Center users

The list of Operations Center users is displayed in the body of the page.

The list contains their username and email address, roles, and date the user was invited or joined.

Users			New User
Admin 	roles: admin	invited on: 04/08/2017	Actions ▾
	roles: admin	joined on: 31/12/0000	Actions ▾
	roles: admin	joined on: 31/12/0000	Actions ▾

Add new Operations Center user

To add a new Operations Center user, on the Users page at the top right, click the New User button.

Enter the new user's full name and email, and select the role you wish to assign to them, then click the Invite button.

New User

Full Name:
The user will receive an invitation email with instructions for how to complete the sign-up process

Email:
Every user must be given at least one role, otherwise he will not be able to login

Roles:
Every user must be given at least one role, otherwise he will not be able to login

An invitation is sent to the email address you specified. The user must click the link in the email to complete the sign-up process.

NOTE: If you prefer to use the command line interface, open a terminal and run the following command:

```
gravity --insecure user create --type=admin --email=<email> --password=<yourpass> --  
↪ops-url=https://gravity-site.kube-system.svc.cluster.local:3009
```

Resend an Operations Center user invitation

This can also be used to reset the password of an Operations Center user who forgot their password.

To resend an invitation to an Operations Center user, on the Users page, click the Actions icon to the far right of their name. Then from the drop-down menu that appears, select Re-send email, then click the Send button.

Revoke an invitation

To revoke an invitation you have sent to a prospective Operations Center user, on the Users page, click the Actions icon to the far right of their name. Then from the drop-down menu that appears, select Revoke invitation.

In the Are you sure? dialog box, click the Delete button.

Edit Operations Center user

You can edit an Operations Center user's name, email and/or roles at any time.

To edit an Operations Center user, on the Users page, click the Actions icon to the far right of their name. Then from the drop-down menu that appears, select Edit.

Make the changes as necessary, then click the Save button.

Delete Operations Center user

To delete an Operations Center user, on the Users page, click the Actions icon to the far right of their name. Then from the drop-down menu that appears, select Delete.

In the Are you sure? dialog box, click the Delete button.

Manage Operations Center roles

You can create new roles and give or remove authorization to all users assigned to a specific role. This includes allowing users to create and modify roles and authentication settings, allowing them API access, and allowing users to use SSH to log into the servers of this cluster.

To get to the Operations Center roles menu, from any page of the Operations Center open the Settings menu by clicking the icon next to your username, then from the drop-down menu that appears, select Settings.

Then from the left navigation menu, click the Roles link.

You will now see a list of roles, and a resource file for each role that defines the role.

Resource files

The resource file controls:

- Which resources and clusters the role can access.
- Whether the access is to `read` or `write` settings or both.
- Whether the role is mapped to a Kubernetes user group. **EXAMPLE:** The `admin` privileged group.
- What Kubernetes namespaces the role has access to. **EXAMPLE:** Access to the `default` namespace enables seeing most cluster deployments and pods.
- Whether the role has access to SSH into the servers of this cluster.
- Which user IDs the role can authenticate as if the role does have access to SSH into the servers of this cluster. **EXAMPLE:** `root` is the default for admins.

Create or edit Operations Center roles

To add a new role, click the New Role button. This shows a blank resource configuration.

To edit an existing role, click the name of a role in the list. This shows the resource configuration that defines the role.

Below is an example of a resource configuration with the definition of an admin role. The admin has `read` and `write` access to all resources, including roles, other users, and authentication settings such as OIDC connectors. The admin also belongs to a privileged Kubernetes group.

```
kind: role
version: v2
metadata:
  name: administrator
spec:
  resources:
    "*":
      - read
      - write
  clusters:
    - "*"
  generate_licenses: true
  kubernetes_groups:
    - admin
  logins:
    - root
  max_session_ttl: "30h0m0s"
```

(continues on next page)

(continued from previous page)

```
namespaces:
  - "*"
repositories:
  - "*"

```

Below is an example of a non-admin role spec providing access to a particular cluster `example.com` and its applications:

```
kind: role
version: v2
metadata:
  name: developer
spec:
  resources:
    cluster:
      - read
      - write
    app:
      - read
      - write
  clusters:
    - example.com
  kubernetes_groups:
    - admin
  logins:
    - root
  max_session_ttl: "10h0m0s"
  namespaces:
    - default
  repositories:
    - "*"

```

After completing the resource configuration, click the Save button to save and return to the list of roles.

Deleting roles

Click the name of a role in the list and click the Delete button.

Forwarding logs

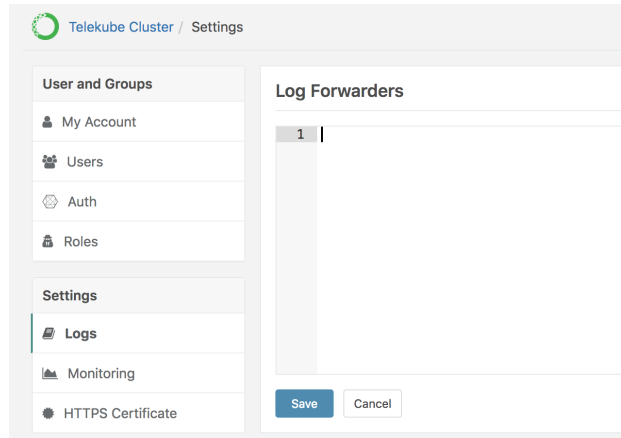
Logs can be forwarded to an IP address and port via TCP or UDP. You will need to know the destination IP addresses and a protocol (TCP or UDP).

Forward logs via TCP or UDP

To forward logs, from any page of the Operations Center, open the Settings menu by clicking the icon next to your username, then from the drop-down menu that appears, select Settings.

Then from the left navigation menu, click the Logs link.

To create a log forwarder configuration, Click the Create button. A Log Forwarders page opens:



Below is a sample log forwarder configuration:

```
kind: logforwarder
version: v2
metadata:
  name: forwarder1
spec:
  address: 192.168.100.1:514
  protocol: udp
```

Note that the `protocol` field is optional and defaults to `tcp`.

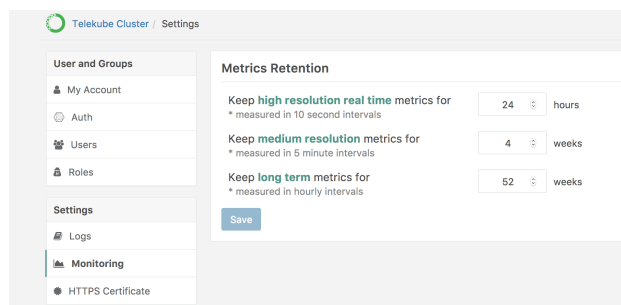
After you have created the resource file, click the Save button.

Monitoring metrics retention

Set the length of time that you retain high and medium resolution metrics, and long term metrics.

To set a metrics retention period, from any page of the Operations Center click the icon next to your username to open the Settings menu, then from the drop-down menu that appears, select Settings.

From the left navigation menu, click the Monitoring link.



Enter a value for length of time to keep high resolution real time metrics, medium resolution metrics, and long term metrics, then click the Save button.

View or change HTTPS Certificate

NOTE: The configuration of TLS/SSL certificates in the Operations Center applies only to the Operations Center web endpoint (on port 32009). Configuration of TLS/SSL certificates for all other Anaconda Enterprise services is a separate configuration process that is documented in the [configuration](#) section.

To add a new HTTPS certificate for the Operations Center, from any page of the Operations Center open the Settings menu by clicking the icon next to your username, then from the drop-down menu that appears, select Settings.

From the left navigation menu, click the HTTPS Certificate link.

HTTPS Certificate		Change
Issued To		
Common Name (CN)	localhost	
Organization (O)	localhost	
Organization Unit (OU)		
Issued By		
Organization (O)	localhost	
Organization Unit (OU)		
Validity Period		
Issued On	Thu, 03 Aug 2017 20:48:34 GMT	
Expires On	Sun, 01 Aug 2027 20:48:34 GMT	

To change any of the settings, click the Change button in the upper right, then follow the directions below for each field.

Add private key

After you have received a Private Key from an SSL provider, save the file to your local computer. Then from the Operations Center's HTTPS Certificate page, click the Browse... button next to the Private Key form box, locate the SSL file on your computer, and click the Upload or Open button.

Click the Save button for the file to be uploaded.

Add HTTPS certificate

After you have received the HTTPS Certificate from an SSL provider, save the file to your local computer. Then from the Operations Center's HTTPS Certificate page, click the Browse... button next to the HTTPS Certificate form box, locate the Certificate file on your computer, and click the Upload or Open button.

Click the Save button for the file to be uploaded.

Add intermediary certificate

If you are using an Intermediate Certificate, save the certificate file to your local computer. Then from the Operations Center's Intermediate Certificate page, click the Browse... button next to the HTTPS Certificate form box, locate the Certificate file on your computer, and click the Upload or Open button.

Click the Save button for the file to be uploaded.

2.3.2 Configuring SSL

After *initial configuration*, configuring SSL certificates for Anaconda Enterprise is done from the Anaconda Enterprise Administrative Settings menu.

This page explains how to change administrative settings through the Anaconda Platform web UI.

1. In the top-right corner of the Anaconda Enterprise screen, click the user icon.
2. In the menu that appears, select Administrative Settings.



Configuring SSL certificates

Assemble the following:

- Registered domain name
- SSL certificate for `servername.domain.tld`, filename `server.crt`
- SSL private key for `servername.domain.tld`, filename `server.key`
- Root SSL certificate (such as [this default Root CA](#)), filename `rootca.crt`. A root certificate is optional but recommended.
- SSL intermediate chain/bundle, filename `intermediate.pem`
- Wildcard domain name
- SSL wildcard certificate for `*.servername.domain.tld`, filename `wildcard.crt`. A wildcard certificate is not necessary if the existing SSL certificate has a Subject Alternative Name for the wildcard domain. If you're not sure, ask your network administrator.
- SSL private key for `*.servername.domain.tld`, filename `wildcard.key`. An SSL private key is not necessary if the existing SSL certificate has a Subject Alternative Name for the wildcard domain. If you're not sure, ask your network administrator.

Copy and paste the files from the previous step as shown:

NOTE: If you have upgraded from a previous version of Anaconda Enterprise, you may need to update your configuration to make sure all services are referencing the correct SSL certificate filenames.

In Anaconda Enterprise 5.1.0 and newer, the default SSL certificate filenames provided by the installer are different than in previous versions. We recommend that you update any Kubernetes secrets you created and update the Anaconda Enterprise configuration to match the new filenames.

[Projects](#)
[Deployments](#)
[Packages](#)

Administrative Settings

Web Certificates

Web Certificates

Anaconda Enterprise ships with self-signed certificates. To use your own certificates, please select the certificates that you wish to use and replace the certificates included by default

Secret last updated at: 22 minutes ago

Domain name

SSL Certificate

Paste value here

SSL private key

Paste value here

Root certificate (Optional)

Paste value here

Intermediate cert (Optional)

Paste value here

Wildcard domain (Optional)

Wildcard certificate (Optional)

Paste value here

Wildcard private key (Optional)

Paste value here

RESET

SAVE

Previous	Updated
rootca.pem	rootca.crt
cert.pem	server.crt
privkey.pem	server.key
tls.crt	wildcard.crt
tls.key	wildcard.key

NOTE: the `keystore.jks` filename is unchanged.

2.3.3 Environments and custom Anaconda installers

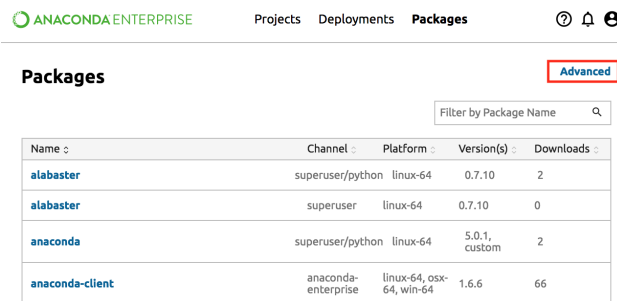
This page explains the custom environments and installers created by Administrators and Superusers for users.

Administrators and Superusers can create custom environments for your users to access. These environments include specific packages and their dependencies. You can give your users access to these environments by creating custom Anaconda installers.

Users can view only the list of environments available to them.

Viewing environments

To see a list of environments and custom Anaconda installers available to you, from the top navigation select the Packages menu, then click the far right Advanced link.



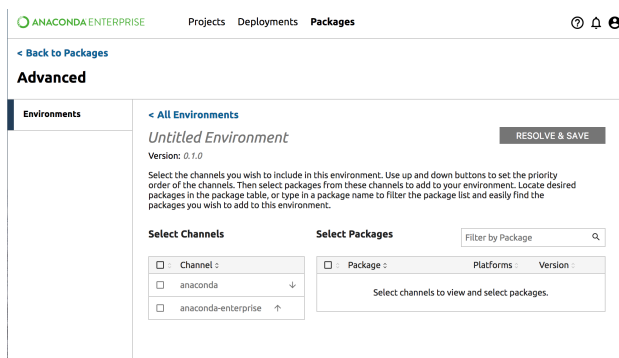
Name	Channel	Platform	Version(s)	Downloads
alabaster	superuser/python	linux-64	0.7.10	2
alabaster	superuser	linux-64	0.7.10	0
anaconda	superuser/python	linux-64	5.0.1, custom	2
anaconda-client	anaconda-enterprise	linux-64, osx-64, win-64	1.6.6	66

Creating an environment

NOTE: See your Administrator if you need permissions to create new environments.

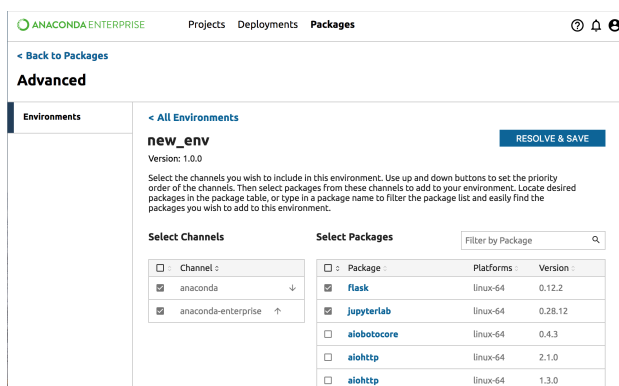
To start creating a new environment, click the top right New Environment button (+). This button is visible only to Superusers and above.

Give your new environment a name by editing the title area - where it says “Untitled Environment” - and optionally edit the version number.



Put a check in the box of any channels that you wish to use in this environment. Once a channel is selected, you can choose from packages available in that channel.

Select packages at specific versions from the package list. To see package descriptions, click on the package name in the list.



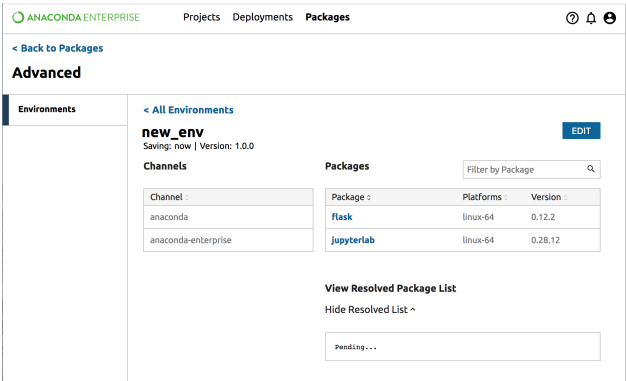
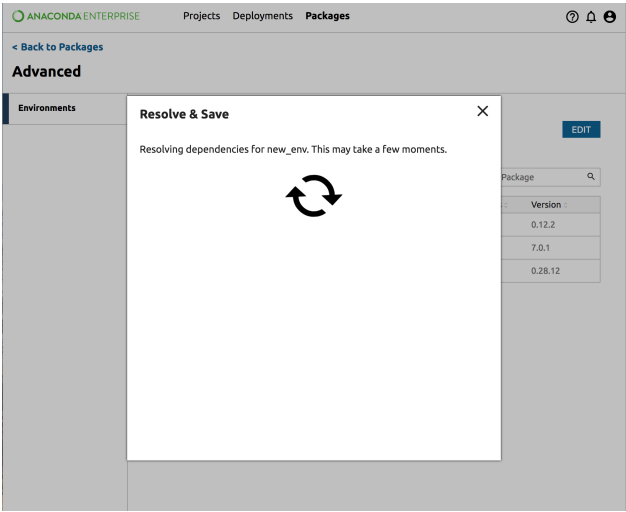
TIP: Use the up and down arrows to set the priority order of the channels.

TIP: Use the sort icon next to the checkbox to move all the selected packages to the top of the list.

Click the Resolve & Save button to resolve dependencies and save the environment.

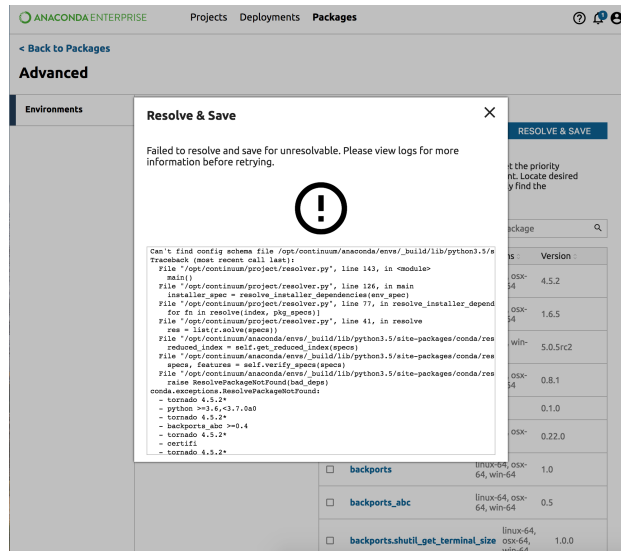
As the environment packages are being resolved, you will see a dialog with a spinner and some information about the status of the process.

You can close this dialog if you prefer to wait for a notification. If you choose to close the dialog, you can view the environment detail:

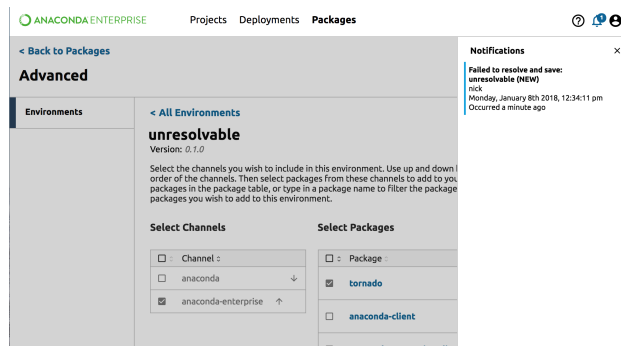


Note: If you return to the environments list you will see your new environment before it is fully saved. If you click on the name, you will also see the same detail view.

If there is an issue with the dependencies, the Resolve dialog will display the log.



If the dialog is closed, a notification provides a link to the relevant logs:



To resolve the issue, review the log, then continue adding and removing packages from your environment before trying to Resolve and Save again.

Viewing an environment

After you have successfully created an environment, it will appear in your environments list:

ANACONDA ENTERPRISE Projects Deployments Packages

< Back to Packages

Advanced

Environments

Environments

Define environment specifications that can then be used to create project templates or installers, including Ambari Management Packs and Spark Parcels.

Environment Specs	Version	Packages	Last Modified
testtr	1.2	232	January 18, 2018, 3:28 am
anaconda5_copy	0.2	230	January 17, 2018, 3:05 pm
py36	0.3	3	January 22, 2018, 2:43 pm

Click on the environment name to get to the environment detail view:

ANACONDA ENTERPRISE Projects Deployments Packages

< Back to Packages

Advanced

Environments

< All Environments

python_36 CREATE EDIT

Saved: January 19 2018, 8:52:21 am | Version: 0.1.0

Channels **Packages** Filter by Package

Channel	Package	Platforms	Version
superuser/python	python	linux-64	3.6.3

View Resolved Package List

Show Resolved List

From this view, you can edit, copy or delete the environment and inspect details of the environment - such as differences between versions and resolved packages. You can also create custom Anaconda installers, custom Anaconda management packs and custom Anaconda parcels.

View resolved package list for an environment

From the environment detail view, you can see a list of resolved packages by clicking Show Resolved List under the list of packages.

View different versions of an environment

From the environment detail view, you can view different versions of your environment by clicking on the dropdown next to the version number:

ANACONDA ENTERPRISE Projects Deployments Packages ? ? ?

[< Back to Packages](#)

Advanced

Environments

[< All Environments](#)

python_36
Saved: January 19 2018, 8:52:21 am | Version: 0.1.0

CREATE EDIT ?

Channels

Channel: superuser/python

Packages

Filter by Package

Package	Platforms	Version
python	linux-64	3.6.3

View Resolved Package List
Hide Resolved List ^

```
Channels
- superuser/python

Packages
- ca-certificates=2017.08.26-h1d4fec0_0
- certifi=2017.7.27.1-py36h8b7b7e_0
- libedit=3.1-hed3344_0
- libffi=3.2.1-h46b6e0_3
- libgcc-ng=7.2.0-h7cc462_2
- libstdc++-ng=7.2.0-h7cc462_2
- ncurses=6.0-h6874d7_1
- openssl=1.0.2m-hb7f4d05_0
- pip=9.0.1-py36h8b7b7e_0
- python=3.6.3-h8b7b7e_0
- readline=7.0-hac23f0_3
- setuptools=40.6.0-py36h8b7b7e_0
- sqlite=3.20.1-h648b0f3_1
- tk=8.6.4-h8b7b7e_0
- wheel=0.29.0-py36h8b7b7e_0
- xz=5.2.3-h8b7b7e_0
- zlib=1.2.11-hb7f4d05_1
```

ANACONDA ENTERPRISE Projects Deployments Packages ? ? ?

[< Back to Packages](#)

Advanced

Environments

[< All Environments](#)

py36
Saved: January 22 2018, 2:43:35 pm | Version: 0.3

CREATE EDIT ?

Channels

Channel: superuser/python

Packages

Filter by Package

Package	Platforms	Version
asn1crypto	linux-64	0.22.0

NOTE: If there are no other versions of the environment, there will not be a dropdown.

Editing an environment

To edit an existing environment and save a new version of it, on the environment detail view, click the Edit button. While editing you must change the version number.

You can also change the channels and packages included in the environment.

To save the new version, click Resolve and Save.

The screenshot shows the 'Packages' page in the Anaconda Enterprise interface. The left sidebar has 'Environments' selected. The main content area is titled 'Advanced' and shows details for an environment named 'new_env' with version '1.0.0'. A 'RESOLVE & SAVE' button is in the top right. Below the environment name, there is a text block explaining how to select channels and packages. The 'Select Channels' section shows two channels: 'anaconda-enterprise' and 'anaconda'. The 'Select Packages' section shows a table of available packages with checkboxes for selection.

Package	Platform	Version
<input checked="" type="checkbox"/> flask	linux-64	0.12.2
<input checked="" type="checkbox"/> jupyterlab	linux-64	0.28.12
<input type="checkbox"/> alabotocore	linux-64	0.4.3
<input type="checkbox"/> aiohttp	linux-64	2.1.0
<input type="checkbox"/> aiohttp	linux-64	1.3.0
<input type="checkbox"/> alabaster	linux-64	0.7.10

After the new version has been saved, the new version will appear in your environments list:

The screenshot shows the 'Environments' page in the Anaconda Enterprise interface. The left sidebar has 'Environments' selected. The main content area is titled 'Environments' and shows a table of environment specifications. The table has columns for 'Environment Specs', 'Version', 'Packages', and 'Last Modified'. The first row shows 'new_env' with version '1.1.0' and 3 packages. Below the table, there is a section for 'Installers' which is currently empty.

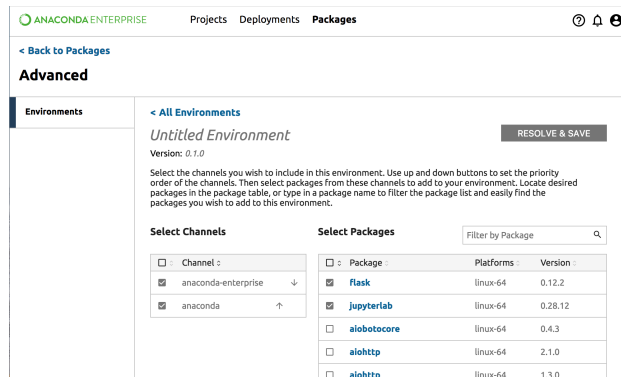
Environment Specs	Version	Packages	Last Modified
new_env	1.1.0	3	15 minutes ago

Copying an environment

To copy an environment, from the environment detail page, click the Gear icon and select Create Copy. When copying, you must choose a new name for the environment by typing in the “Untitled Environment” box. Environment names

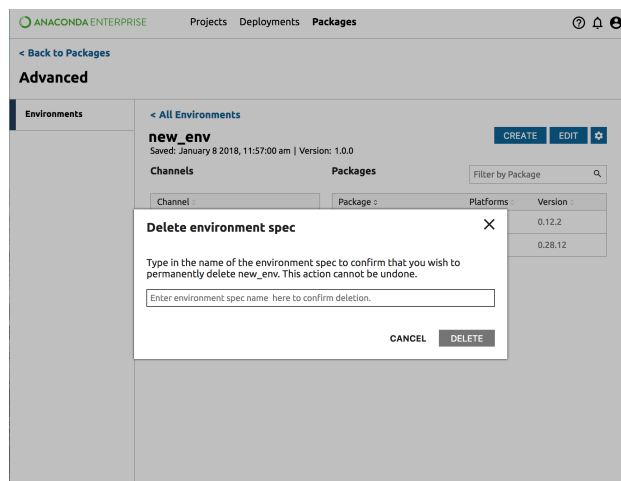
can contain alphanumeric characters and underscores only.

Choose a new version number by typing in the Version box. Select or unselect channels and packages that you wish to include.



Deleting an environment

To delete an environment, from the environment detail page click the Gear icon and select Delete environment spec. You will be asked to verify the name of the environment before deletion by typing in the environment name. You can also copy/paste the name. When you have entered the name correctly, the delete button turns red. NOTE: There is no “Are you sure?” dialog box.

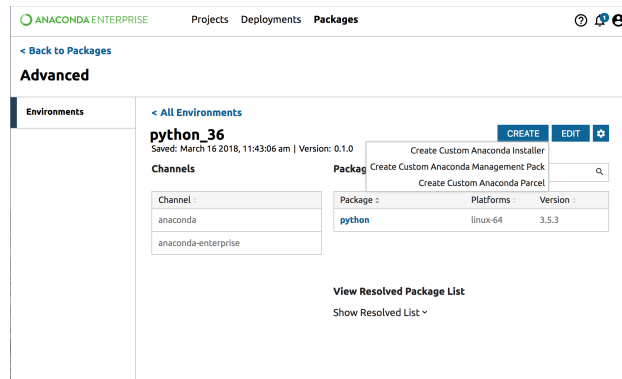


Creating custom Anaconda installers

Administrators and Superusers can create custom Anaconda installers for your users to access.

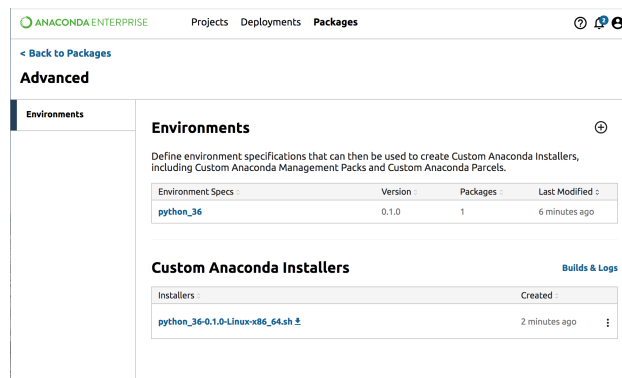
To create a custom Anaconda installer, custom Anaconda management pack, or custom Anaconda parcel for your users, first create a new environment.

Then from the environment detail view, click the Create button and select the type of installer that you would like to create:



Clicking on one of these options will create the installer. This process may take several minutes. You can stay in the dialog box, or close it and wait for a notification.

After the installer has been created, the new installer will appear in the custom Anaconda installers list:

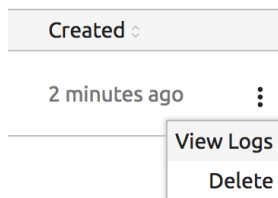


You can use the More Actions icon (three vertical dots) to delete the installer or view the relevant logs:

Viewing Builds & Logs

You can view the logs from building environments and custom Anaconda installers from the Builds & Logs link at the top right of the custom Anaconda installers list.

Builds & Logs



This link will give you access to running jobs (and the ability to terminate them), as well as failed and successful job runs.

ANACONDA ENTERPRISE Projects Deployments Packages 🔍 🔔 ⚙️

Job Runs Filter by Name and Owner 🔍

Name	Status	Owner	Started	Updated
zxc_0.1.0_env_spec_lock_job_run	finished job	superuser	24 minutes ago	23 minutes ago
zxcvbnm_0.1.0_env_spec_lock_job_run	finished job	superuser	25 minutes ago	24 minutes ago
asdf_0.1.0_env_spec_lock_job_run	finished job	superuser	29 minutes ago	28 minutes ago
asdf123_0.1.0_env_spec_lock_job_run	finished job	superuser	5 hours ago	5 hours ago

2.3.4 Backup and Restore

This page explains the backup and restore operations of Anaconda Enterprise.

Anaconda Enterprise can back up and restore the following key components of the platform:

- platform state and configuration
- platform storage (database, object storage, and project version control)

The backups are produced in the form of a pair of `gzip`-compressed `tar` files, and the restore operation requires a matching set of backup files to proceed.

NOTE: All backup and restore operations are executed on the `head` node, in the Anaconda Enterprise environment.

To run all the below operations, you must copy the `backup.sh` and `restore.sh` scripts from the installer tarball to the Anaconda Enterprise environment:

```
sudo cp backup.sh restore.sh /opt/anaconda
```

Then enter that environment to run the scripts:

```
sudo gravity enter
```

Backup

The backup script will stop the system.

Ensure all users have logged out.

In the terminal, run the script:

```
cd /opt/anaconda/
bash backup.sh
```

For help, run `bash backup.sh -h`.

This will produce a pair of backup files, in the same directory as the script was run:

- `ae5-data-backup-{timestamp}.tar.gz`
- `ae5-state-backup-{timestamp}.tar.gz`

In both cases `{{timestamp}}` is the time when the backup ran.

NOTE: Be sure to test and verify your backups. Store backups in multiple redundant locations for maximum safety.

Restore

In the terminal, run:

```
cd /opt/anaconda/
bash restore.sh data-backup-file state-backup-file
```

NOTE: Replace `data-backup-file` with the path to the data backup file generated by running the Anaconda Enterprise backup script, and replace `state-backup-file` with the path to the state backup file generated by running the Anaconda Enterprise backup script.

For help, run `bash restore.sh -h`.

2.3.5 Master node failure recovery

Other than storage-related services (database, git server, and object storage), all core Anaconda Enterprise services are resilient to master node failure.

To maintain operation of Enterprise in the event of a master node failure, `/opt/anaconda/` on the master node should be located on a redundant disk array or backed up frequently to avoid data loss. See [Backup/Restore](#) for more information.

To restore Anaconda Enterprise operations in the event of a master node failure, complete the following steps:

1. Create a new master node
2. Restore data from a backup

Create a new master node

Follow the installation process for adding a new cluster node, described in the section [Unattended installation](#).

NOTE: To create the new master node, select `--role=master` instead of `--role=worker`.

Restore data from backup

After the installation of the new master node is complete, follow the instructions in the [restore data](#) section.

2.4 Troubleshooting

2.4.1 Entering Anaconda Enterprise environment

To enter the Anaconda Enterprise environment and gain access to `kubectl` and other commands within Anaconda Enterprise, use the command:

```
sudo gravity enter
```

Moving files and data

Occasionally you may need to move files and data from the host machine to the Anaconda Enterprise environment. If so, there are two *shared mounts* to pass data back and forth between the two environments:

- host: `/opt/anaconda/` -> AE environment: `/opt/anaconda/`
- host: `/var/lib/gravity/planet/share` -> AE environment: `/ext/share`

If data is written to either of the locations, that data will be available on both the host machine and within the Anaconda Enterprise environment

2.4.2 Debugging

AWS Traffic needs to handle the public IPs and ports. You should either use a canonical security group with the proper ports opened or manually add the specific ports listed in [Network Requirements](#).

2.4.3 Failed installations

If an installation fails, you can view the failed logs as part of the support bundle in the failed installation UI.

After executing `sudo gravity enter` you can check `/var/log/messages` to troubleshoot a failed installation or these types of errors.

After executing `sudo gravity enter` you can run `journalctl` to look at logs to troubleshoot a failed installation or these types of errors:

```
journalctl -u gravity-23423lkqjfefqpfh2.service
```

NOTE: Replace `gravity-23423lkqjfefqpfh2.service` with the name of your gravity service.

You may see messages in `/var/log/messages` related to errors such as “etcd cluster is misconfigured” and “etcd has no leader” from one of the installation jobs, particularly `gravity-site`. This usually indicates that `etcd` needs more compute power, needs more space or is on a slow disk.

Anaconda Enterprise is very sensitive to disk latency, so we usually recommend using a better disk for `/var/lib/gravity` on target machines and/or putting `etcd` data on a separate disk. For example, you can mount `etcd` under `/var/lib/gravity/planet/etcd` on the hosts.

After a failed installation, you can *uninstall Anaconda Enterprise* and start over with a fresh installation.

2.4.4 Failed on pulling gravitational/rbac

If the node refuses to install and fails on pulling `gravitational/rbac`, create a new directory `TMPDIR` before installing and provide *write access* to user 1000.

2.4.5 Problems during air gap project migration

The command `anaconda-project lock` over-specifies the channel list resulting in a conda bug where it adds defaults from the internet to the list of channels.

Solution:

Add to the `.condarc`: “default_channels”. This way, when conda adds “defaults” to the command it is adding the internal repo server and not the `repo.continuum.io` URLs.

EXAMPLE:

```
default_channels:
- anaconda
channels:
- our-internal
- out-partners
- rdkit
- bioconda
- defaults
- r-channel
- conda-forge
channel_alias: https://:8086/conda
auto_update_conda: false
ssl_verify: /etc/ssl/certs/ca.2048.cer
```

2.4.6 Problems during post-install or post-upgrade steps

Post-install and post-upgrade steps run as Kubernetes jobs. When they complete running the pods used to run them are **not** removed. These and other stopped pods can be found using:

```
kubectl get pods -a
```

The logs in each of these three pods will be helpful for diagnosing issues in the following steps:

Pod	Issues in this step
ae-wagonwheel	post-install UI
install	installation step
postupdate	post-update steps

2.4.7 Problems after post-install configuration does not work

In order to reinitialize the post-install configuration UI, which can come in handy for regenerating temporary (self-signed) SSL certificates or reconfiguring the platform based on your domain name, you must re-create and re-expose the service on a new port.

First, recreate the `ap-wagonwheel` deployment:

```
kubectl create -f /var/lib/gravity/site/packages/unpacked/gravitational.io/
↪AnacondaEnterprise/5.X.X/resources/wagonwheel.yaml -n kube-system
```

NOTE: Replace `5.X.X` with your actual version number.

Then execute `sudo gravity enter` and run:

```
kubectl get deploy -n kube-system
```

to check the services running in the system namespace. One of these should be `ae-wagonwheel`, the post-install configuration UI. To make this visible to the outside world, run:

```
kubectl expose deploy ae-wagonwheel --port=8000 --type=NodePort --name=post-install -  
↪n kube-system
```

This will run the UI on a new port, allocated by Kubernetes, under the name `post-install`. Run:

```
kubectl get svc -n kube-system | grep post-install
```

to find out which port it is listening under, then navigate to `http://<your domain>:<this port>` to see the post-install UI again.

2.4.8 LDAP error in ap-auth

```
[LDAP: error code 12 - Unavailable Critical Extension]; remaining name  
'dc=acme, dc=com'
```

This error can be caused when pagination is turned on. Pagination is a server side extension and is not supported by some LDAP servers, notably the Sun Directory server.

Frequently asked questions

- *General*
 - *When was the general availability release of Anaconda Enterprise v5?*
 - *Which notebooks or editors does Anaconda Enterprise support?*
 - *Can I deploy multiple data science applications to Anaconda Enterprise?*
 - *Does Anaconda Enterprise support high availability deployments?*
 - *Which identity management and authentication protocols does Anaconda Enterprise support?*
 - *Does Anaconda Enterprise support two-factor authentication (including one-time passwords)?*
- *System requirements*
 - *What operating systems are supported for Anaconda Enterprise?*
 - *What are the minimum system requirements for Anaconda Enterprise nodes?*
 - *Which browsers are supported for Anaconda Enterprise?*
 - *Does Anaconda Enterprise come with a version control system?*
 - *Can Anaconda Enterprise integrate with my own Git server?*
- *Installation*
 - *How do I install Anaconda Enterprise?*
 - *Can Anaconda Enterprise be installed on-premises?*
 - *Can Anaconda Enterprise be installed on cloud environments?*
 - *Does Anaconda Enterprise support air gapped (off-line) environments?*
 - *Can I build Docker images for the install of Anaconda Enterprise?*
 - *Can I install Anaconda Enterprise on my own instance of Kubernetes?*

- *Can I get the AE installer packaged as a virtual machine (VM), Amazon Machine Image (AMI) or other installation package?*
- *Which ports are externally accessible from Anaconda Enterprise?*
- *Can I use Anaconda Enterprise to connect to my Hadoop/Spark cluster?*
- *How can I manage Anaconda packages on my Hadoop/Spark cluster?*
- *On how many nodes can I install Anaconda Enterprise?*
- *Anaconda Project*
 - *What operating systems and Python versions are supported for Anaconda Project?*
 - *How is encapsulation with Anaconda Project different from creating a workspace or project in Spyder, PyCharm, or other IDEs?*
 - *What types of projects can I deploy?*
 - *Does Anaconda Enterprise include Docker images for my data science projects?*
- *Notebooks*
 - *Are the deployed, self-service notebooks read-only?*
 - *What happens when other people run the notebook? Does it overwrite any file, if notebook is writing to a file?*
 - *Can I define environment variables as part of my data science project?*
 - *How are Anaconda Project and Anaconda Enterprise available?*
 - *Where can I find example projects for Anaconda Enterprise?*
 - *Does Anaconda Enterprise support batch scoring with REST APIs?*
 - *Does Anaconda Enterprise provide tools to help define and implement REST APIs?*
- *Help and training*
 - *Do you offer support for Anaconda Enterprise?*
 - *Do you offer training for Anaconda Enterprise?*
 - *Do you have a question not answered here?*

3.1 General

3.1.1 When was the general availability release of Anaconda Enterprise v5?

Our GA release was August 31, 2017 (Version 5.0.3).

3.1.2 Which notebooks or editors does Anaconda Enterprise support?

Anaconda Enterprise supports the use of Jupyter Notebooks and JupyterLab, which are the most popular integrated data science environments for working with Python and R notebooks.

3.1.3 Can I deploy multiple data science applications to Anaconda Enterprise?

Yes, you can deploy multiple data science applications and languages across an Anaconda Enterprise cluster. Each data science application runs in a secure and isolated environment with all of the dependencies from Anaconda that it requires.

A single node can run multiple applications based on the amount of compute resources (CPU and RAM) available on a given node. Anaconda Enterprise handles all of the resource allocation and application scheduling for you.

3.1.4 Does Anaconda Enterprise support high availability deployments?

Partially. Some of the Anaconda Enterprise services and user-deployed apps will be automatically configured when installed to three or more nodes. Anaconda Enterprise provides several automatic mechanisms for fault tolerance and service continuity, including automatic restarts, health checks, and service migration.

For more information, see [Fault Tolerance](#).

3.1.5 Which identity management and authentication protocols does Anaconda Enterprise support?

- LDAP / AD
- SAML
- Kerberos
- [Identity Brokering](#)

3.1.6 Does Anaconda Enterprise support two-factor authentication (including one-time passwords)?

Yes, Anaconda Enterprise supports single sign-on (SSO) and two-factor authentication (2FA) using FreeOTP, Google Authenticator or Google Authenticator compatible 2FA.

You can configure one-time password policies in Anaconda Enterprise by navigating to the authentication center and clicking on Authentication and then OTP Policy.

3.2 System requirements

3.2.1 What operating systems are supported for Anaconda Enterprise?

Please see [operating system requirements](#).

NOTE: Linux distributions other than those listed in the documentation can be supported on request.

3.2.2 What are the minimum system requirements for Anaconda Enterprise nodes?

Please see [system requirements](#).

3.2.3 Which browsers are supported for Anaconda Enterprise?

Please see *browser requirements*.

3.2.4 Does Anaconda Enterprise come with a version control system?

Yes, Anaconda Enterprise includes an internal git server, which allows users to save and commit versions of their projects.

3.2.5 Can Anaconda Enterprise integrate with my own Git server?

In a future version of Anaconda Enterprise 5.x, you will be able to connect to external git servers.

3.3 Installation

3.3.1 How do I install Anaconda Enterprise?

The Anaconda Enterprise installer is a single tarball that includes Docker, Kubernetes, system dependencies, and all of the components and images necessary to run Anaconda Enterprise. The system administrator runs one command on each node.

3.3.2 Can Anaconda Enterprise be installed on-premises?

Yes, including airgapped environments.

3.3.3 Can Anaconda Enterprise be installed on cloud environments?

Yes, including Amazon AWS, Microsoft Azure, and Google Cloud Platform.

3.3.4 Does Anaconda Enterprise support air gapped (off-line) environments?

Yes, the Anaconda Enterprise installer includes Docker, Kubernetes, system dependencies, and all of the components and images necessary to run Anaconda Enterprise on-premises or on a private cloud, with or without internet connectivity. We can deliver the installer to you on a USB drive.

3.3.5 Can I build Docker images for the install of Anaconda Enterprise?

No. The installation of Anaconda Enterprise is supported only by using the single-file installer. The Anaconda Enterprise installer includes Docker, Kubernetes, system dependencies, and all of the components and images necessary for Anaconda Enterprise.

3.3.6 Can I install Anaconda Enterprise on my own instance of Kubernetes?

No. The Anaconda Enterprise installer already includes Kubernetes.

3.3.7 Can I get the AE installer packaged as a virtual machine (VM), Amazon Machine Image (AMI) or other installation package?

No. The installation of Anaconda Enterprise is supported only by using the single-file installer.

3.3.8 Which ports are externally accessible from Anaconda Enterprise?

Please see *network requirements*.

3.3.9 Can I use Anaconda Enterprise to connect to my Hadoop/Spark cluster?

Yes. Anaconda Enterprise supports connectivity from notebooks to local or remote Spark clusters by using the Sparkmagic client and a Livy REST API server. Anaconda Enterprise provides Sparkmagic, which includes Spark, PySpark, and SparkR notebook kernels for deployment.

3.3.10 How can I manage Anaconda packages on my Hadoop/Spark cluster?

An administrator can generate custom Anaconda parcels for Cloudera CDH or custom Anaconda management packs for Hortonworks HDP using Anaconda Enterprise. A data scientist can use these Anaconda libraries from a notebook as part of a Spark job.

3.3.11 On how many nodes can I install Anaconda Enterprise?

You can install Anaconda Enterprise in the following configurations during the initial installation:

- One node (one master node)
- Two nodes (one master node, one worker node)
- Three nodes (one master node, two worker nodes)
- Four nodes (one master node, three worker nodes)

After the initial installation, you can add or remove worker nodes from the Anaconda Enterprise cluster at any time.

One node serves as the master node and writes storage to disk, and the other nodes serve as worker nodes. Anaconda Enterprise services and user-deployed applications run seamlessly on the master and worker nodes.

3.4 Anaconda Project

3.4.1 What operating systems and Python versions are supported for Anaconda Project?

Anaconda Project supports Windows, macOS and Linux, and tracks the latest Anaconda releases with Python 2.7, 3.5 and 3.6.

3.4.2 How is encapsulation with Anaconda Project different from creating a workspace or project in Spyder, PyCharm, or other IDEs?

A workspace or project in an IDE is a directory of files on your desktop. Anaconda Project encapsulates those files, but also includes additional parameters to describe how to run a project with its dependencies. Anaconda Project is portable and allows users to run, share, and deploy applications across different operating systems.

3.4.3 What types of projects can I deploy?

Anaconda Project is very flexible and can deploy many types of projects with conda or pip dependencies. Deployable projects include:

- Notebooks (Python and R)
- Bokeh applications and dashboards
- REST APIs in Python and R (including machine learning scoring and predictions)
- Python and R scripts
- Third-party apps, web frameworks, and visualization tools such as Tensorboard, Flask, Falcon, deck.gl, plot.ly Dash, and more.

Any generic Python and R script or webapp can be configured to serve on port 8086, which will show the app in Anaconda Enterprise when deployed.

3.4.4 Does Anaconda Enterprise include Docker images for my data science projects?

Anaconda Enterprise includes data science application images for the editor and deployments. You can install additional packages in either environment using Anaconda Project. Anaconda Project includes the information required to reproduce the project environment with Anaconda, including Python, R, or any other conda package or pip dependencies.

3.5 Notebooks

3.5.1 Are the deployed, self-service notebooks read-only?

Yes, the deployed versions of self-service notebooks are read-only, but they can be executed by collaborators or viewers. Owners of the project that contain the notebooks can edit the notebook and deploy (or re-deploy) them.

3.5.2 What happens when other people run the notebook? Does it overwrite any file, if notebook is writing to a file?

A deployed, self-service notebook is read-only but can be executed by other collaborators or viewers. If multiple users are running a notebook that writes to a file, the file will be overwritten unless the notebook is configured to write data based on a username or other environment variable.

3.5.3 Can I define environment variables as part of my data science project?

Yes, Anaconda Project supports environment variables that can be defined when deploying a data science application. Only project collaborators can view or edit environment variables, and they cannot be accessed by viewers.

3.5.4 How are Anaconda Project and Anaconda Enterprise available?

Anaconda Project is free and open-source. Anaconda Enterprise is a commercial product.

3.5.5 Where can I find example projects for Anaconda Enterprise?

Sample projects are included as part of the Anaconda Enterprise installation, which include sample workflows and notebooks for Python and R such as financial modeling, natural language processing, machine learning models with REST APIs, interactive Bokeh applications and dashboards, image classification, and more.

The sample projects include examples with visualization tools (Bokeh, deck.gl), pandas, scipy, Shiny, Tensorflow, Tensorboard, xgboost, and many other libraries. Users can save the sample projects to their Anaconda Enterprise account or download the sample projects to their local machine.

3.5.6 Does Anaconda Enterprise support batch scoring with REST APIs?

Yes, Anaconda Enterprise can be used to deploy machine learning models with REST APIs (including Python and R) that can be queried for batch scoring workflows. The REST APIs can be made available to other users and accessed with an API token.

3.5.7 Does Anaconda Enterprise provide tools to help define and implement REST APIs?

Yes, a data scientist can basically create a model without much work for the API development. Anaconda Enterprise includes an API wrapper for Python frameworks that builds on top of existing web frameworks in Anaconda, making it easy to expose your existing data science models with minimal code. You can also deploy REST APIs using existing API frameworks for Python and R.

3.6 Help and training

3.6.1 Do you offer support for Anaconda Enterprise?

Yes, we offer full support with Anaconda Enterprise.

3.6.2 Do you offer training for Anaconda Enterprise?

Yes, we offer product training for collaborative, end-to-end data science workflows with Anaconda Enterprise.

3.6.3 Do you have a question not answered here?

[Please](#) for more information.

4.1 Anaconda Enterprise 5.1.2

Released: March 16, 2018

4.1.1 Administrator-facing changes

- Fixed issue with image/version tags when upgrading AE

4.1.2 Backend improvements (non-visible changes)

- Updated to Kubernetes 1.7.14

4.2 Anaconda Enterprise 5.1.1

Released: March 12, 2018

4.2.1 Administrator-facing changes

- Ability to specify custom UID for service account at install-time (default UID: 1000)
- Added pre-flight checks for kernel modules, kernel settings, and filesystem options when installing or adding nodes
- Improved initial startup time of project creation, sessions, and deployments after installation. Note that all services will be in the `ContainerCreating` state for 5 to 10 minutes while all AE images are being pre-pulled, after which the AE user interface will become available.
- Improved upgrade process to automatically handle upgrading AE core services

- Improved consistency between GUI- and CLI-based installation paths
- Improved security and isolation between internal database from user sessions and deployments
- Added capability to configure a custom trust store and LDAPS certificate validation
- Simplified installer packaging using a single tarball and consistent naming
- Updated documentation for system requirements, including XFS filesystem requirements and kernel modules/settings
- Updated documentation for mirroring packages from channels
- Added documentation for configuring AE to point to online Anaconda repositories
- Added documentation for securing the internal database
- Added documentation for configuring RBAC, role mapping, and access control
- Added documentation for LDAP federation and identity management
- Improved documentation for backup/restore process
- Fixed issue when deleting related versions of custom Anaconda parcels
- Added command to remove channel permissions
- Fixed issue related to Ops Center user creation in post-install configuration
- Silenced warnings when using `verify_ssl` setting with `anaconda-enterprise-cli`
- Fixed issue related to default admin role (`ae-admin`)
- Fixed issue when generating TLS/SSL certificates with FQDNs greater than 64 characters
- Fixed issue when using special characters with AE Ops Center accounts/passwords
- Fixed bug related to Administrator Console link in menu

4.2.2 User-facing changes

- Improvements to collaborative workflow: Added notification when collaborators make changes to a project, ability to pull changes into a project, and ability to resolve conflicting changes when saving or pulling changes into a project.
- Additional documentation and examples for connecting to remote data and compute sources: Spark, Hive, Impala, and HDFS
- Optimized startup time for Spark and SAS project templates
- Improved initial startup time of project creation, sessions, and deployments by pre-pulling images after installation.
- Increased upload limit of projects from 100 MB to 1 GB
- Added capability to `sudo yum install` system packages from within project sessions
- Fixed issue when uploading projects that caused them to fail during partial import
- Fixed R kernel in R project template
- Fixed issue when loading `sparklyr` in Spark Project
- Fixed issue related to displaying kernel names and Spark project icons
- Improved performance when rendering large number of projects, packages, etc.
- Improved rendering of long version names in environments and projects

- Render full names when sharing projects and deployments with collaborators
- Fixed issue when sorting collaborators and package versions
- Fixed issue when saving new environments
- Fixed issues when viewing installer logs in IE 11 and Safari

4.3 Anaconda Enterprise 5.1.0

Released: January 19, 2018

4.3.1 Administrator-facing changes

- New post-installation administration GUI with automated configuration of TLS/SSL certificates, administrator account, and DNS/FQDN settings; significantly reduces manual steps required during post-installation configuration process
- New functionality for administrators to generate custom Anaconda installers, parcels for Cloudera CDH, and management packs for Hortonworks HDP
- Improved backup and restore process with included scripts
- Switched from groups to roles for role-based access control (RBAC) for Administrator and superuser access to AE services
- Clarified system requirements related to system modules and IOPS in documentation
- Added ability to specify fractional CPUs/cores in global container resource limits
- Fixed consistency of TLS/SSL certificate names in configuration and during creation of self-signed certificates
- Changed use of `verify_ssl` to `ssl_verify` throughout AE CLI for consistency with `conda`
- Fixed configuration issue with licenses, including field names and online/offline licensing documentation

4.3.2 User changes

- Updated default project environments to Anaconda Distribution 5.0.1
- Improved configuration and documentation on using Sparkmagic and Livy with Kerberos to connect to remote Spark clusters
- Fixed R environment used in sample projects and project template
- Fixed UI rendering issue on package detail view of channels, downloads, and versions
- Fix multiple browser compatibility issues with Microsoft Edge and Internet Explorer 11
- Fixed multiple UI issues with Anaconda Project JupyterLab extension

4.3.3 Backend improvements (non-visible changes)

- Updated to Kubernetes 1.7.12
- Updated to conda 4.3.32
- Added SUSE 12 SP2/SP3, and RHEL/CentOS 7.4 to supported platform matrix

- Implemented TLS 1.2 as default TLS protocol; added support for configurable TLS protocol versions and ciphers
- Fixed default superuser roles for repository service, which is used for initial/internal package configuration step
- Implemented secure flag attribute on all session cookies containing session tokens
- Fixed issue during upgrade process that failed to vendor updated images
- Fixed `DiskNodeUnderPressure` and cluster stability issues
- Fixed Quality of Service (QoS) issue with core AE services on under-resourced nodes
- Fixed issue when using access token instead of ID token when fetching roles from authentication service
- Fixed issue with authentication proxy and session cookies

4.3.4 Known issues

- IE 11 compatibility issue when using Bokeh in notebooks (including sample projects)
- IE 11 compatibility issue when downloading custom installers

4.4 Anaconda Enterprise 5.0.6

Released: November 9, 2017

4.5 Anaconda Enterprise 5.0.5

Released: November 7, 2017

4.6 Anaconda Enterprise 5.0.4

Released: September 12, 2017

4.7 Anaconda Enterprise 5.0.3

Released: August 31, 2017 (General Availability Release)

4.8 Anaconda Enterprise 5.0.2

Released: August 15, 2017 (Early Adopter Release)

4.9 Anaconda Enterprise 5.0.1

Released: March 8, 2017 (Early Adopter Release)

Features:

- Simplified, one-click deployment of data science projects and deployments, including live Python and R notebooks, interactive data visualizations and REST APIs.
- End-to-end secure workflows with SSL/TLS encryption.
- Seamlessly managed scalability of the entire platform
- Industry-grade productionization, encapsulation, and containerization of data science projects and applications.

- *Updating a package from the Anaconda metapackage*
- *File size limit when uploading files*
- *IE 11 compatibility issue when using Bokeh in projects (including sample projects)*
- *IE 11 compatibility issue when downloading custom Anaconda installers*
- *Project names over 40 characters may prevent JupyterLab launch*

5.1 Updating a package from the Anaconda metapackage

When updating a package dependency of a project, if that dependency is part of the Anaconda metapackage the package will be installed once but a subsequent `anaconda-project` call will uninstall the upgraded package.

5.1.1 Workaround

When updating a package dependency remove the `anaconda metapackage` from the list of dependencies at the same time add the new version of the dependency that you want to update.

5.1.2 Affected Versions

5.1.0, 5.1.1

5.2 File size limit when uploading files

Unable to upload new files inside of a project that are larger than the current restrictions:

- The limit of file uploads in JupyterLab is 15 MB

5.2.1 Affected Versions

5.1.0, 5.1.1

5.3 IE 11 compatibility issue when using Bokeh in projects (including sample projects)

Bokeh plots and applications have had a number of issues with Internet Explorer 11, which typically result in the user seeing a blank screen.

5.3.1 Workaround

Upgrade to the latest version of Bokeh available. On Anaconda 4.4 the latest is 0.12.7. On Anaconda 5.0 the latest version of Bokeh is 0.12.13. If you are still having issues, consult the Bokeh team or support.

5.3.2 Affected Versions

5.1.0, 5.1.1

5.4 IE 11 compatibility issue when downloading custom Anaconda installers

Unable to download a custom Anaconda installer from the browser when using Internet Explorer 11 on Windows 7. Attempting to download a custom installer with this setup will result in an error that “This page can’t be displayed”.

5.4.1 Workaround

Custom installers can be downloaded by refreshing the page with the error message, clicking the “Fix Connection Error” button, or using a different browser.

5.4.2 Affected Versions

5.1.0, 5.1.1

5.5 Project names over 40 characters may prevent JupyterLab launch

If a project name is more than 40 characters long, launching the project in JupyterLab may fail.

5.5.1 Workaround

Rename the project to a name less than 40 characters long and launch the project in JupyterLab again.

5.5.2 Affected Versions

5.1.1

SEE ALSO: *Concepts*.

Anaconda Project An encapsulation of your data science assets to make it easily portable. Anaconda Project automates setup, so you can quickly share and execute projects. All Project setup can be done from the Enterprise web interface.

Anaconda Project CLI A command-line interface included with Anaconda Enterprise that allows data scientists to create and share channels and packages.

Anaconda Repository A centralized location on your network for storing over 1,000 professionally built software packages for data science, from which the packages can be retrieved and installed.

Channel A location in the repository where Anaconda Enterprise looks for packages.

Classic notebook Refers to Jupyter Notebook, the previous version of the browser-based interactive development environment available in Anaconda Enterprise. It combines the notebook, file browser, text editor, terminal and outputs in one software product. See also the next-generation product, JupyterLab.

Commit Making a set of local changes permanent by copying them to the remote server. Anaconda Enterprise checks to see if your work will conflict with the commits that your colleagues have made on the same project, so the files cannot be overwritten unless you so choose.

Deploy

Deployment A deployed Anaconda Project. When you deploy a project, Enterprise finds and builds all of the software dependencies—the programs on which the Project depends in order to run—and encapsulates them so they are completely self-contained. This allows you to easily share the application with others.

Interactive data applications Visualizations with sliders, drop-downs and other widgets that allow users to interact with them. Interactive data applications can drive new computations, update plots and connect to other programmatic functionality.

Interactive development environment (IDE) A suite of software tools that combines everything a developer needs to write and test software. It typically includes a code editor, a compiler or interpreter and a debugger that the developer accesses through a single graphical user interface (GUI). An IDE may be installed locally, or it may be included as part of one or more existing and compatible applications accessed through a web browser.

JupyterLab The next-generation browser-based interactive development environment with flexible building blocks for interactive and collaborative computing. JupyterLab still contains the notebook, file browser, text editor, terminal and outputs in the same product. For classic Jupyter Notebook users, the interface for JupyterLab is familiar, but even more enjoyable and productive to work with.

Live notebooks Together, JupyterLab and the classic Jupyter Notebooks are web-based IDE applications that allows you to create and share documents that contain live code, equations, visualizations and explanatory text.

Package Software files and information about the software, such as its name, the specific version and a description, that are bundled into a file that can be installed and managed by a package manager. Packages can then be encapsulated into Anaconda Projects for easy portability.

Project templates When creating a new project, you can select a comprehensive project template that contains a set of packages and their dependencies. Choices are Anaconda 3.6 or 3.5, Anaconda 2, R language, SAS or Spark.

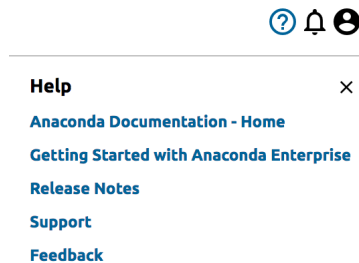
REST APIs A common way to operationalize machine learning models is through REST APIs. REST APIs are callable URLs which provide results based on a query. This allows developers to make their applications intelligent without having to write models themselves.

CHAPTER 7

Help and support

To obtain help and support from within Anaconda Enterprise:

1. In the top right navigation bar, click the **Help** icon:



2. Select any of the following:

- Documentation—A link to this documentation.
- Getting Started with Anaconda Enterprise—a tutorial.
- Release Notes—Detailed information about the latest version.
- Support—A link to our support system.
- Feedback—A link to our feedback system.
- Get email support from enterprise_innovator@anaconda.com.
- Provide feedback by completing this [5-minute survey](#).

A

Anaconda Project, [205](#)
Anaconda Project CLI, [205](#)
Anaconda Repository, [205](#)

C

Channel, [205](#)
Classic notebook, [205](#)
Commit, [205](#)

D

Deploy, [205](#)
Deployment, [205](#)

I

Interactive data applications, [205](#)
Interactive development environment (IDE), [205](#)

J

JupyterLab, [206](#)

L

Live notebooks, [206](#)

P

Package, [206](#)
Project templates, [206](#)

R

REST APIs, [206](#)