**Table of Contents**

[**1.** **Install Docker** 2](#_Toc7412853)

[**2.** **Installing Go** 2](#_Toc7412854)

[**2.1** **Download the tarball** 2](#_Toc7412855)

[**2.2** **Verify the tarball** 2](#_Toc7412856)

[**2.3** **Extract the tarball** 2](#_Toc7412857)

[**2.4** **Setting Paths for Go** 2](#_Toc7412858)

[**2.5** **Test Go installation** 3](#_Toc7412859)

[**3.** **Astronomer CLI** 4](#_Toc7412860)

[**3.1** **Installation** 4](#_Toc7412861)

[**3.2** **Create project and running locally** 5](#_Toc7412862)

[**3.3** **Issues during CLI Installation** 5](#_Toc7412863)

[**3.3.1** **Dockerfile version issue** 6](#_Toc7412864)

[**3.3.2** **Postgres panic runtime error** 6](#_Toc7412865)

# **Install Docker**

Astronomer requires Docker to be installed as a prerequisite. This section gives the installation process of Docker on Amazon Linux

sudo yum update -y

sudo yum install -y docker

sudo service docker start

sudo usermod -a -G docker <your-username>

***Note****: Log out and log in to pick up the added group*

# **Installing Go**

Astronomer requires Go to be installed as a prerequisite. This section gives the installation process of Go on Amazon Linux

## **Download the tarball**

wget https://dl.google.com/go/go1.10.3.linux-amd64.tar.gz

## **Verify the tarball**

Once the download is completed verify the tarball checksum with the sha256sum command

sha256sum go1.10.3.linux-amd64.tar.gz

## **Extract the tarball**

Use the tar command to extract the tarball to the /usr/local directory:

sudo tar -C /usr/local -xzf go1.10.3.linux-amd64.tar.gz

## **Setting Paths for Go**

To execute Go like any other command, we need to append its install location to the $PATH variable. Go was installed in a system directory, which is why we will set the environment variable globally.

Create a path.sh script in the /etc/profile.d directory using the vi editor:

sudo vi /etc/profile.d/path.sh

Add the following to the file /etc/profile.d/path.sh save and exit:

export PATH=$PATH:/usr/local/go/bin

***Note****: If Go was installed in a different location, then adjust the path accordingly in the above* export

Additionally, define the GOPATH and GOBIN Go environment variables in your user’s .bash\_profile file to point to the recently created workspace. The GOPATH variable tells Go the location of your source files, while the GOBIN variable instructs it where to create the compiled binary files.

Open the .bash\_profile file:

vi ~/.bash\_profile

Add the following to the end of the file, save and exit:

export GOBIN="$HOME/projects/bin"

export GOPATH="$HOME/projects/src"

***Note****: if Go was not installed in the* /usr/local *directory, then add the* GOROOT *variable as well to the above 2 exports as shown*

export GOROOT="add/your/path/to/Go/here"

To apply the changes to your current BASH session, use the source command to reload the updated profiles:

source /etc/profile && source ~/.bash\_profile

## **Test Go installation**

Writing our first program will ensure that our environment is working. To get started, create a new .go file:

vi ~/projects/src/hello.go

The code below uses the main Go package, imports the formatted IO content component, and sets a new function to print the string “Hello, World!”. Add the following to the file

package main

import "fmt"

func main() {

fmt.Printf("Hello, World!\n")

}

Then, save and exit the file. Next, compile the hello.go source file with the go install command:

go install $GOPATH/hello.go

We are now ready to run our program:

$GOBIN/hello

The hello.go program should produce a “Hello, World!” message, confirming a successful installation of Go.

# **Astronomer CLI**

This section gives the installation process of Astronomer CLI. Ensure Docker and Go are installed before proceeding with this installation *(refer section 1 and 2)*.

## **Installation**

Install Astro CLI by running the following command

If you are a Cloud customer, run:

curl -sSL https://install.astronomer.io | sudo bash -s -- 0.7.5

If you are an Enterprise customer, run:

curl -sSL https://install.astronomer.io | sudo bash

Confirm Astro installation by typing the following command

astro

If you're set up properly, you should see the following:

astro is a command line interface for working with the Astronomer Platform.

Usage:

astro [command]

Available Commands:

airflow Manage airflow projects and deployments

auth Mangage astronomer identity

cluster Manage Astronomer EE clusters

config Manage astro project configurations

deployment Manage airflow deployments

help Help about any command

upgrade Check for newer version of Astronomer CLI

user Manage astronomer user

version Astronomer CLI version

workspace Manage Astronomer workspaces

Flags:

-h, --help help for astro

## **Create project and running locally**

Now create a sample project hello-astro using following commands

mkdir hello-astro && cd hello-astro

astro airflow init

astro airflow init will build a base image from Astronomer's fork of Apache-Airflow using Alpine Linux. The build process will include everything in your project directory, which makes it easy to include any shell scripts, static files, or anything else you want to include in your code.

Once that command is run, you'll see the following skeleton project generated inside the directory

.

├── dags *# Where your DAGs go*

│ ├── example-dag.py

├── Dockerfile *# For runtime overrides*

├── include *# For any other files you'd like to include*

├── packages.txt *# For OS-level packages*

├── plugins *# For any custom or community Airflow plugins*

└── requirements.txt *# For any python packages*

Now you can run astro airflow start and see Airflow running on localhost:8080/admin

All changes made to the dags and plugins directory will be picked up automatically - any changes made to any of the other files will need the image to be rebuilt using astro airflow stop and astro airflow start

## **Issues during CLI Installation**

This section describes the issues faced during the Astro CLI installation process and steps taken to resolve these issue.

### **Dockerfile version issue**

After executing the astro airflow init command, a skeleton project will be generated in the directory. Dockerfile is one of the generated files. The initial contents of Dockerfile is given below:

FROM astronomerinc/ap-airflow:0.7.5-2-1.9.0-onbuild

The above line specifies the container image version 0.7.5-2-1.9.0-onbuild that needs to be pulled from at runtime from DockerHub, but version specified in the generated file is incorrect and results in an following error

Env file ".env" found. Loading...

Sending build context to Docker daemon 11.78kB

Step 1/1 : FROM astronomerinc/ap-airflow:0.7.5-2-1.9.0-onbuild

manifest for astronomerinc/ap-airflow:0.7.5-2-1.9.0-onbuild not found

Error: command 'docker build -t del-this-2/airflow:latest failed: failed to execute cmd: exit status 1

We managed to solve this by specifying the container image that is available on DockerHub and that is compatible with the CLI version being used. We changed the initial contents as given below:

FROM astronomerinc/ap-airflow:0.7.5-1.9.0-onbuild

### **Postgres panic runtime error**

On resolving the Dockerfile issue, we ran into Postgres panic runtime error error which occurs on execution of astro airflow start. The error is given below

Env file ".env" found. Loading...

Sending build context to Docker daemon 11.78kB

Step 1/1 : FROM astronomerinc/ap-airflow:0.7.5-1.9.0-onbuild

# Executing 5 build triggers

---> Using cache

---> Using cache

---> Using cache

---> Using cache

---> Using cache

---> d6924fac8a92

Successfully built d6924fac8a92

Successfully tagged del-this-2/airflow:latest

Creating network "delthis2\_airflow" with driver "bridge"

Creating volume "delthis2\_airflow\_logs" with driver "local"

Creating volume "delthis2\_postgres\_data" with driver "local"

INFO[0015] [0/3] [postgres]: Starting

Pulling postgres (postgres:10.1-alpine)...

**panic: runtime error: index out of range**

.

.

.

created by github.com/astronomer/astro-cli/vendor/github.com/docker/libcompose/project.(\*Project).startService

/go/src/github.com/astronomer/astro-cli/vendor/github.com/docker/libcompose/project/project.go:448 +0x93e

We solved this issue by running the following command inside the project directory

docker pull postgres:10.1-alpine

Then again run the command astro airflow start, you should be able to see the following

Env file ".env" found. Loading...

Sending build context to Docker daemon 11.78kB

Step 1/1 : FROM astronomerinc/ap-airflow:0.7.5-1.9.0-onbuild

# Executing 5 build triggers

---> Using cache

---> Using cache

---> Using cache

---> Using cache

---> Using cache

---> d6924fac8a92

Successfully built d6924fac8a92

Successfully tagged del-this-2/airflow:latest

INFO[0000] [0/3] [postgres]: Starting

INFO[0001] [1/3] [postgres]: Started

INFO[0001] [1/3] [scheduler]: Starting

INFO[0001] [2/3] [scheduler]: Started

INFO[0001] [2/3] [webserver]: Starting

INFO[0002] [3/3] [webserver]: Started

Airflow Webserver: http://localhost:8080/admin/

Postgres Database: localhost:5432/postgres

Now navigate to localhost:8080/admin to access the airflow UI

**Reference link for solution**

<https://forum.astronomer.io/t/i-get-a-pulling-postgres-postgres-10-1-alpine-panic-runtime-error-index-out-of-range-when-i-try-to-astro-airflow-start/133>