Preparing to install Cloud Pak for Integration on TechZone AWS, Azure and IBM Cloud Environments

This document will walk you through the steps on how to prepare your environment for the purpose of demonstrating the Cloud Pak for Integration on TechZone.

This will include various steps such as:

How to request a demo environment, **Logging into your cloud of choice**, and **Step by Step guide for running the Automation**.

Goals for the Demo:

- Prepare your TechZone environment to demo Cloud Pak for Integration AWS, Azure and IBM Cloud which are supported with ROSA, ARO and ROKS OpenShift clusters.
- Deploying the Cloud Pak for Integration capabilities of your choice like Platform UI, API connect, App Connect Enterprise, MQ and Event Streams.

Prerequisites:

- A valid IBM ID that can be used to access
  - TechZone https://techzone.ibm.com/
- A valid GitHub ID that can be used to create a repository in your own organization
  - GitHub https://github.com/
- A valid Github token with permissions set to create and remove repositories
- Install a code editor, we recommend **VSCode**
  - VS Code https://code.visualstudio.com/

Licenses and Entitlements

Details on Cloud Pak for Integration licensing available at https://www.ibm.com/docs/en/cloud-paks/cp-integration/2022.2?topic=planning-licensing

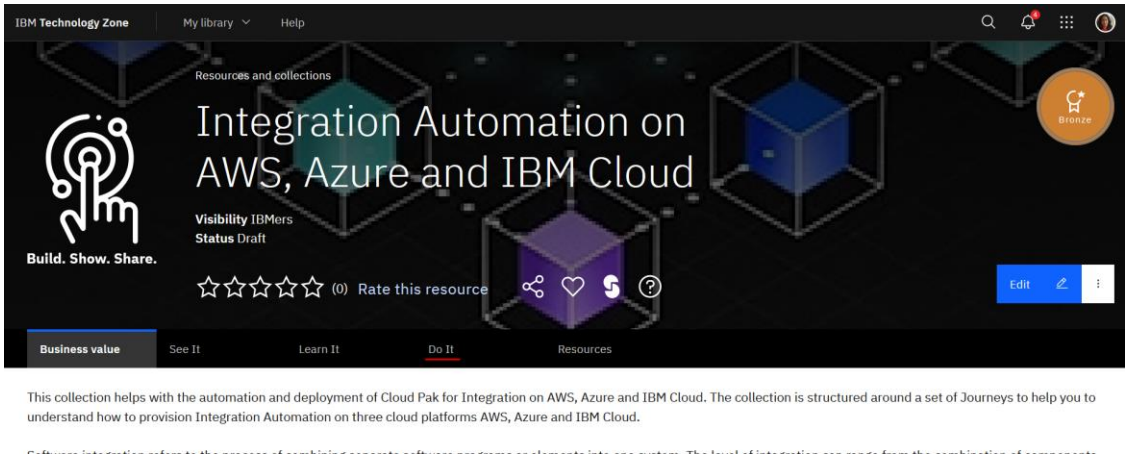
Obtaining your IBM entitlement API key

To install Cloud Pak for Integration you are required to have an entitlement key that provides access to the software components. After the necessary entitlements have been granted, use the following steps to download the entitlement key and apply it to the automation:

1. Visit the Container Software Library site - https://myibm.ibm.com/products-services/containerlibrary
2. Log in with your IBMId credentials
3. Assuming the entitlements are in place, you will be presented with an entitlement key. Click "Copy key".
4. This will be used in steps subsequently while deploying cloud pak for Integration

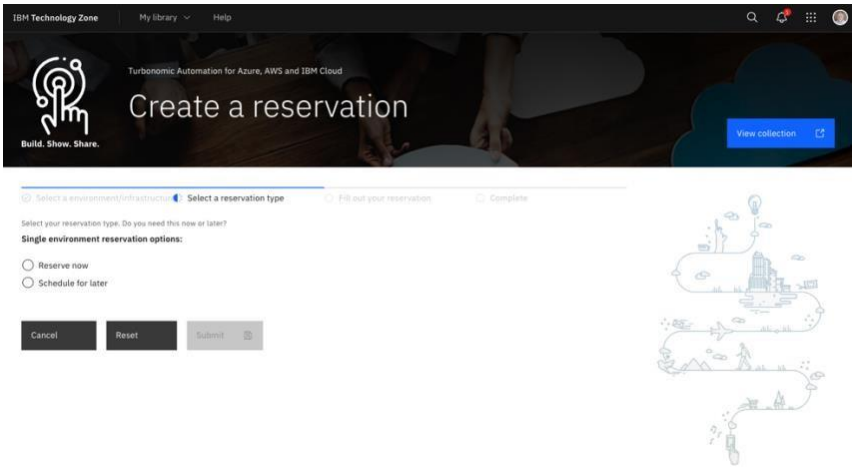
Requesting TechZone Environment

1. If you have not already done so, access the Tech Zone collection for Cloud Pak for Integration Automation for AWS, Azure, and IBM Cloud https://techzone.ibm.com/collection/integration-automation-on-aws-azure-and-ibm-cloud

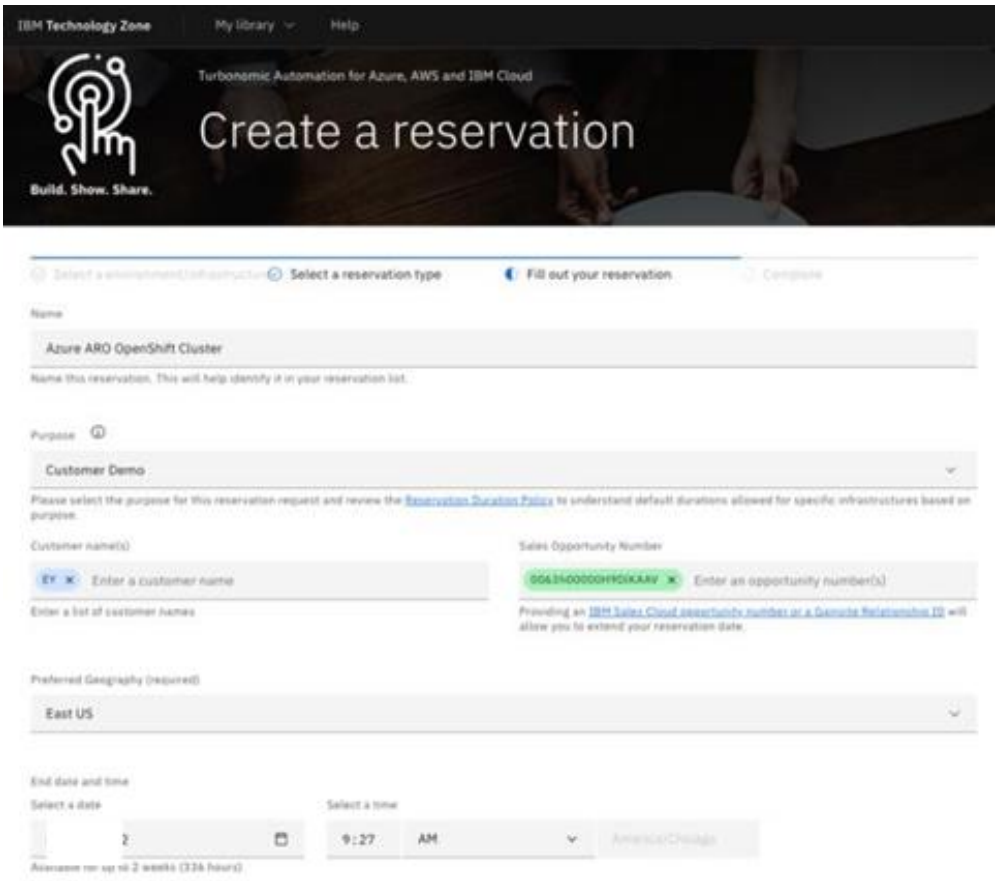


2. The first thing you need to do is to request an access to the demo environment. Click on **Do It** Tab

3. Click on the cloud type of your choice. For example **Azure ARO OpenShift Cluster** environment

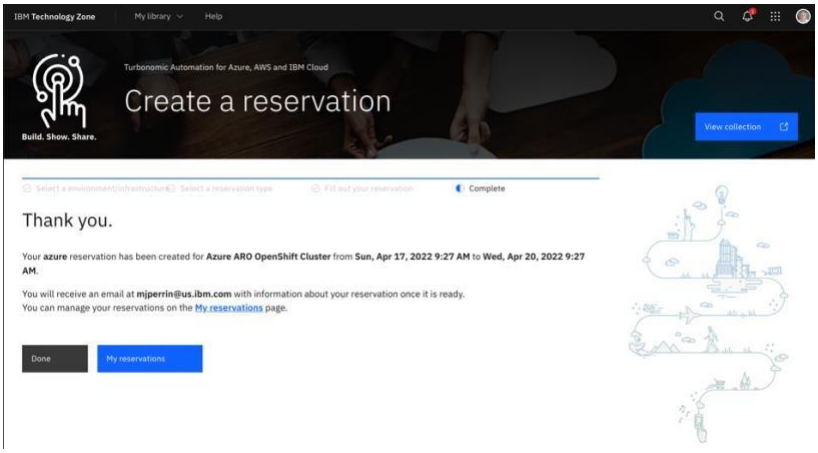


4. Click on the **Reserve now** radio button



5. Complete the Reservation Type form. Please make sure all the fields are complete, you may want to select **Customer Demo** as the Purpose of your request
    a. As these are real AWS, Azure, and IBM Cloud environment, you will need to use a real customer opportunity value from **IBM Sales Cloud** https://w3.ibm.com/w3publisher/ibm-sales-cloud

6. Enter the **Sales Opportunity Number** as you will not be able to proceed without a valid and live opportunity
7. Select your duration for the environment, and for the size for Cloud Pak for Integration based on the capabilities to be deployed https://www.ibm.com/docs/en/cloud-paks/cp-integration/2022.2?topic=requirements-compute-resources-development-environments
8. Click on **Submit** once all the fields are entered correctly
9. Once the **Submit** has completed you will see the following screen with a THANK YOU message to confirm that your reservation is being processed

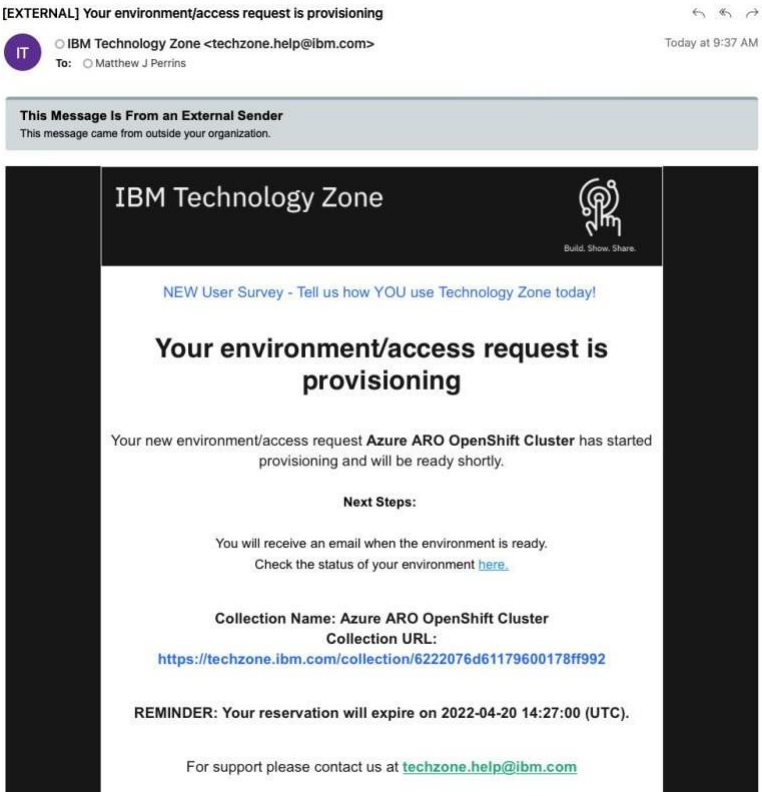


Check your email for confirmation.

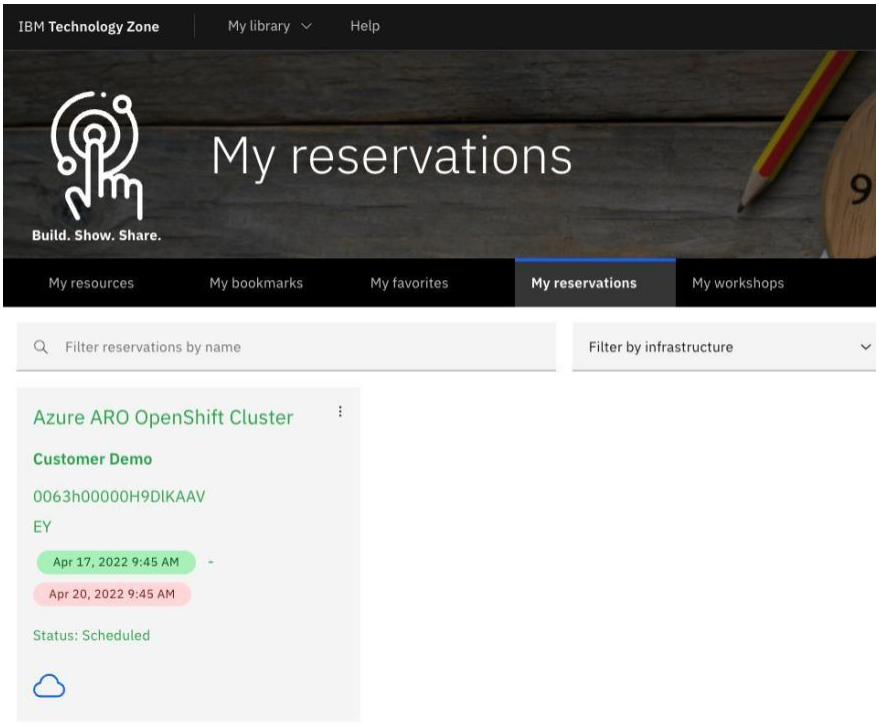
Receive Confirmation Email Log in for the First Time

Steps:

1. Once you have reserved the demo environment, you will receive the confirmation email.



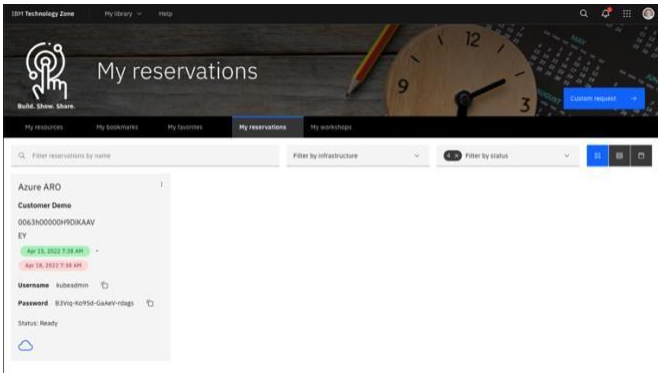
2. It is important that you read the email to follow the instructions on how to log into the Tech Zone Demo Account.
3. You will receive a second email once your reservation has been processed; Click on **My reservations** to see the status of your reservation.



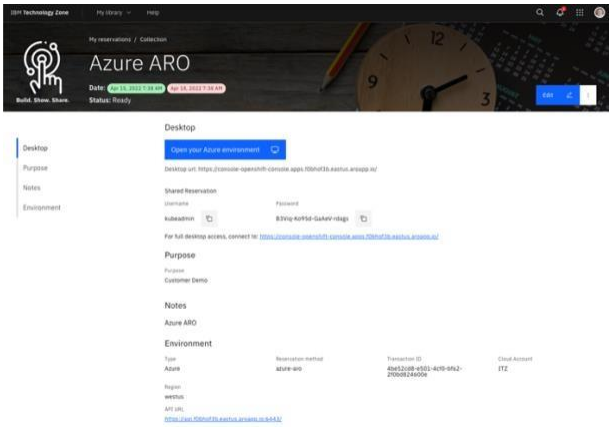
Log into your TechZone Environment

Steps:

1. Navigate to **TechZone** and **Your Reservations**, click on the reservation you created in the previous steps.

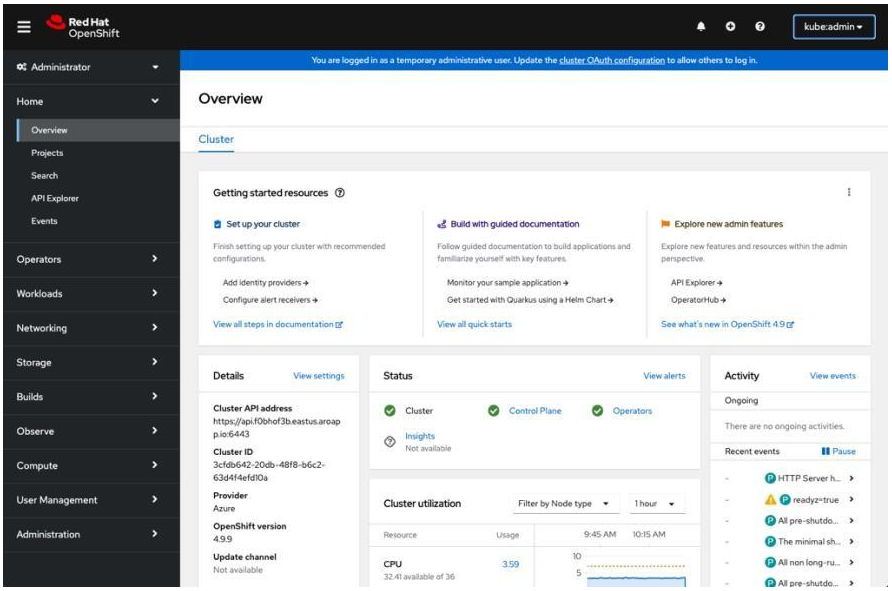


2. You will see the reservation view in detail, copy the **password** and click on the **Open your Azure environment** button.



3. This will display the OpenShift login screen for the cluster, enter the user ID and password details from the previous reservation screen. Typically, it is **kubeadmin** for the user id and the system generated password.

4. Once you have logged in you will see the main OpenShift administration screen



5. You have successfully logged into your TechZone environment. You can now start the installation process

Installing Cloud Pak for Integration into your TechZone environment

The installation process will use standard Terraform git repository that has been built using the modules. You need to make Cloud Pak for Integration installation consistent across the three cloud environments AWS, Azure, and IBM Cloud.

Set up the runtime environment

At this time the most reliable way of running this automation is with Terraform on your local machine either through a bootstrapped docker image or Virtual Machine. We provide both a container image and a virtual machine cloud-init script that have all the common SRE tools installed.

Supported Runtimes

There are two supported runtimes where the automation is expected to be executed inside of:

1. Docker Desktop (Container engine)
2. Multipass (VM)

The Terraform automation can be run from the local operating system, but it is recommended to use either of the runtimes listed above, which provide a consistent and controlled environment, with all dependencies pre-installed. Detailed instructions for downloading and configuring both Docker Desktop and Multipass can be found in RUNTIMES.md

Set up environment credentials

1. First step is to clone the automation code to your local machine. Run this git command in your favorite command line shell.

   $ git clone git@github.com:IBM/automation-integration-platform.git

2. Navigate into the **automation-integration-platform** folder using your command line.
   a. The **README.md** has a comprehensive instruction on how to install this into cloud environments other than TechZone. This document focuses on running in a TechZone requested environment.

   $ cd automation-integration-platform

3. The first step is to setup your **credentials.properties** file. This will enable a secure access to your cluster.

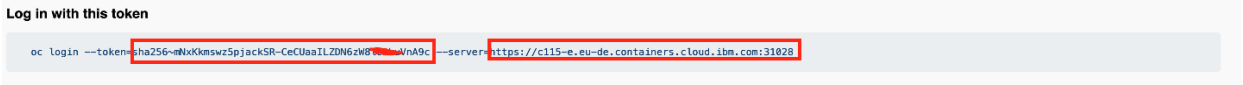
   $ cp credentials.template credentials.properties
   $ code credentials.properties

   ```
   # Add the values for the Credentials to access the IBM
   Cloud
   # Instructions to access this information can be found in
   the README.MD
   # This is a template file and the ./launch.sh script looks for
   a file based on this template named credentials.properties

   ## gitops_repo_username: The username of the user with
   ```

```
access to the repository
export TF_VAR_gitops_repo_username=

## gitops_repo_token: The personal access token used to
access the repository
export TF_VAR_gitops_repo_token=

## TF_VAR_cluster_login_token: Token used for
authentication to the api server
export TF_VAR_cluster_login_token=

## TF_VAR_server_url: The url for the OpenShift api server
export TF_VAR_server_url=

## TF_VAR_entitlement_key: The entitlement key used to
access the IBM software images in the container registry.
Visit https://myibm.ibm.com/products-
services/containerlibrary to get the key
export TF_VAR_entitlement_key=


## TF_VAR_ibmcloud_api_key: IBM Cloud API Key required
to provision storage on IBM Cloud
export TF_VAR_ibmcloud_api_key=

# Only needed if targeting AWS Deployment
export TF_VAR_access_key=
export TF_VAR_secret_key=

##
## Azure credentials
## Credentials are required to install Portworx on an Azure
account. These credentials must have
## particular permissions in order to interact with the
account and the OpenShift cluster. Use the
## provided `azure-portworx-credentials.sh` script to
retrieve/generate these credentials.
##
# Only needed if targeting Azure Deployment

## TF_VAR_azure_subscription_id: The subscription id for
the Azure account. This is required if Azure portworx is
used
export TF_VAR_azure_subscription_id=
## TF_VAR_azure_tenant_id: The tenant id for the Azure
account. This is required if Azure portworx is used
export TF_VAR_azure_tenant_id=
## TF_VAR_azure_client_id: The client id of the user for
the Azure account. This is required if Azure portworx is
used
export TF_VAR_azure_client_id=
## TF_VAR_azure_client_secret: The client id of the user
for the Azure account. This is required if Azure portworx is
used
export TF_VAR_azure_client_secret=
```

4. You will need to populate these values. Add your **GitHub** username and your Personal Access Token to **TF_VAR_gitops_repo_username** and **TF_VAR_gitops_repo_token**
5. From you **OpenShift console** click on top right menu and select **Copy login command** and click on **Display Token**

6. Copy the API Token value into the **TF_VAR_cluster_login_token** value and Server URL into the **TF_VAR_server_url** value, only the part starting with https



7. Set the **TF_VAR_gitops_repo_username** value to your GitHub User Name
8. Copy the entitlement key, this can be obtained from visiting the IBM Container Library and place it in the **TF_VAR_entitlement_key** variable

Configure Storage (Skip, if you already have the required storage configured in the OpenShift cluster)

Deploying RookNFS Server on IBM Cloud, Azure & AWS

In Cloud Pak for Integration v2022.2.1, IBM introduced the support for deploying Rook NFS server which would facilitate the deployment of PlatformNavigator using the underlying RWO (ReadWriteOnce) storage. The instruction provided in https://www.ibm.com/docs/en/cloud-paks/cp-integration/2022.2?topic=ui-deploying-platform-rwo-storage is automated in this integration-automation asset.

"**IMPORTANT**: Please NOTE. Use this storage in case of PoC/PoTs/Demos. Leverage IBM ODF or Portworx for real customer engagement."

If you decided to use this, move on to `Set up the Runtime Environment` section.

Deploying on IBM Cloud (Portworx or ODF)

1. Provide the IBM Cloud API Key for the target IBM Cloud account as the value for TF_VAR_ibmcloud_api_key

Deploying on Azure (Portworx)

If Cloud Pak for Integration will be deployed on OpenShift deployed on Azure, the credentials for the Azure account need to be provided. Several clis are required for these steps:

- az cli - https://docs.microsoft.com/en-us/cli/azure/install-azure-cli
- jq cli - https://stedolan.github.io/jq/download/

You can install these clis on your local machine **OR** run the following commands within the provided container image by running launch.sh

1. Log into your Azure account

   az login

2. Run the azure-portworx-credentials.sh script to gather/create the credentials:

   ./azure-portworx-credentials.sh -t {cluster type} -g {resource group name} -n {cluster name} [-s {subscription id}]

   where:

   - **cluster type** is the type of OpenShift cluster (aro or ipi).
   - **resource group name** is the name of the Azure resource group where the cluster has been provisioned.
   - **cluster name** is the name of the OpenShift cluster.
   - **subscription id** is the subscription id of the Azure account. If a value is not provided it will be looked up.
3. Update credentials.properties with the values output from the script.

```
{
  "azure_client_id": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
  "azure_client_secret": "XXXXXXX",
  "azure_tenant_id": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
  "azure_subscription_id": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
}
```

4. If you used the container image to run the script, type exit to close the container shell then re-rung launch.sh to pick up the changes to the environment variables.

Configure the automation

Get the Portworx configuration spec (for AWS or Azure deployments)

5. Follow the steps to download the portworx configuration spec
6. Copy the downloaded file into the root directory of the cloned automation-integration-platform repository

Set up the automation workspace

1. Launch the automation runtime. Ensure the current working directory is 'automation-integration-platform'
   1.1. If using *Docker Desktop*, run *./launch.sh*. This will start a container image with the prompt opened in the /terraform directory.
   1.2. If using *Multipass*, run *mutlipass shell cli-tools* to start the interactive shell, and cd into the /automation/{template} directory, where {template} is the folder you've cloned this repo. Be sure to run source credentials.properties once in the shell.
2. Next, we need to set up the working directory for the automation: setup-workspace.sh script is responsible for choosing the required module to be deployed on Openshift Cluster. The module we refer here is cater to "GitOps, Storage & Cloud Pak capabilities (PlatformNavigator, APIC, MQ, ACE and EventStreams)".

   2.1. If your decision was to use RookNFS storage

   ./setup-workspace-with-rook-NFS.sh [-p {cloud provider}] [-n {prefix name}]

   where:

   - **cloud provider** - the target cloud provider for the deployment (aws, azure, or ibm)
   - **portworx spec file** (optional) - the name of the file containing the Portworx configuration spec yaml

   At this stage, We assume you have Openshift Cluster up and running. Following info will help the user in setting up the right workspace.

   if [OpenShift Cluster is Provisoned on IBM Cloud]
   ./setup-workspace-with-rook-NFS.sh -p ibm

   if [OpenShift Cluster is Provisoned on AWS]
   ./setup-workspace-with-rook-NFS.sh -p aws

   if [OpenShift Cluster is Provisoned on Azure]
   ./setup-workspace-with-rook-NFS.sh -p azure

   2.2. If you decided to go with 'odf' or 'portworx' storage

   ./setup-workspace.sh [-p {cloud provider}] [-s {storage}] [-r {region}] [-x {portworx spec file}] [-n {prefix name}]

   where:

   - **cloud provider** - the target cloud provider for the deployment (aws, azure, or ibm)
   - **storage** - the intended storage provider (portworx or odf)
   - **region** (mandatory in case of `azure` & `aws`) - the region where OpenShift Cluster is deployed
   - **prefix name** (optional) - the name prefix that will be used for the gitops repo
   - **portworx spec file** (mandatory in case of `azure` & `aws`) - the name of the file containing the Portworx configuration spec yaml

At this stage, We assume you have Openshift Cluster up and running. Following info will help the user in setting up the right workspace.

if [OpenShift Cluster is Provisoned on IBM Cloud && No Storage Provisioned]
In IBM Cloud you have a choice to provision 'Open Data Foundation' or 'Portworx'
In case
'OpenData Foundation' :
./setup-workspace-with-odf-or-portworx.sh -p ibm -s odf
'portworx' :
./setup-workspace-with-odf-or-portworx.sh -p ibm -s portworx -r [{ region where OCP is provisioned }] [-x {portworx spec file}]

if [OpenShift Cluster is Provisoned on Azure && No Storage Provisioned]
In Azure, at this point of time you can only go with 'portworx'
./setup-workspace-with-odf-or-portworx.sh -p azure -s portworx -r [{ region where OCP is provisioned }] [-x {portworx spec file}]

if [OpenShift Cluster is Provisoned on aws && No Storage Provisioned]
In aws, at this point of time you can only go with 'portworx'
./setup-workspace-with-odf-or-portworx.sh -p aws -s portworx -r [{ region where OCP is provisioned }] [-x {portworx spec file}]

3. The setup-workspace.sh script configures the terraform.tfvars file with reasonable defaults.

   Note: The default terraform.tfvars file is symbolically linked to the new workspaces/current folder so to enables you to edit the file in your native operating system using editor of choice.

The following are variables that you will be updating. Here are some suggested values.

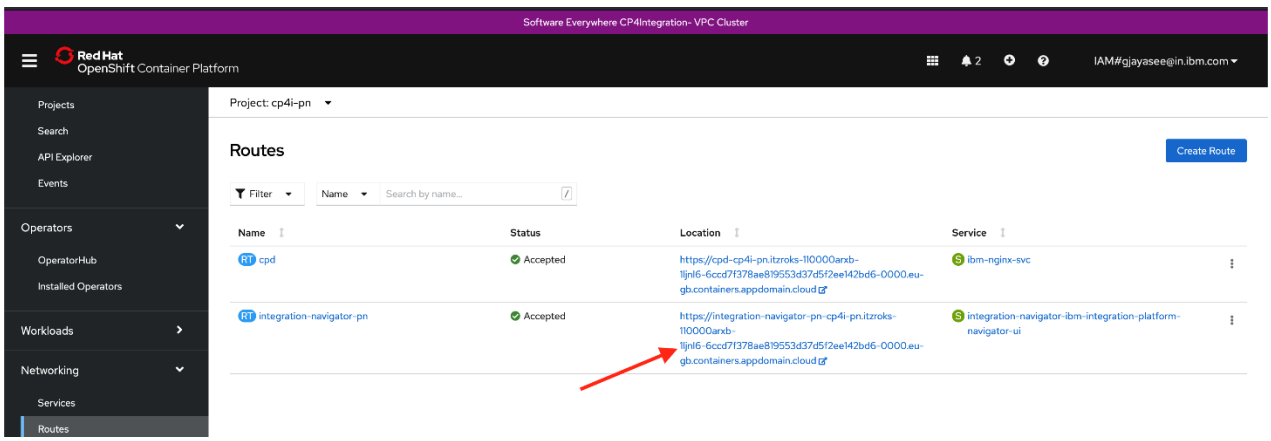
Variable	Description	Suggested Value
rwx_storage_class	The storage class to use for Read-Write-Many volumes.	In case of 'ibm cloud' & 'odf' storage it is 'ocs-storagecluster-cephfs'. In case of 'aws' cloud & 'portworx' storage it is 'portworx-gp3-sc'. In case of 'azure' cloud & 'portworx' storage it is 'portworx-gp3-sc'. In case of 'aws' or 'aws' or 'azure' cloud & 'RookNFS' storage it is 'integration-storage'
rwo_storage_class	The storage class to use for Read-Write-One volumes. on aws: gp2, on azure: managed-premium, on ibm: ibmc-vpc-block-10iops-tier	In case of 'ibm cloud' & 'odf' storage it is 'ibmc-vpc-block-10iops-tie'. In case of 'aws' cloud & 'portworx' storage it is 'gp2'. In case of 'azure' cloud & 'portworx' storage it is 'managed-premium'
gitops_repo_host	The host for the git repository.	github.com
gitops_repo_type	The type of the hosted git repository (github or gitlab).	github
gitops_repo_org	The org/group where the git repository will be created. If the value is left blank then it will default to the username	github userid or org

Variable	Description	Suggested Value
gitops_repo_repo	The short name of the repository to create	cp4i-gitops
gitops-cluster-config_banner_text	Banner text for the cluster console	Cloud Pak for Integration
portworx_spec_file	The content of Portworx yaml in base64 encoded format	Will be setup by setup-workspace-with-odf-or-portworx

1. Scan through the `terraform.tfvars` file thoroughly and double check the values based on your environment.
2. Save the terraform.tfvars file

Apply the automation

1. We are now ready to start installing Cloud Pak for Integration. Ensure you are inside the running container or Multipass VM.
2. Within the container terminal, change directory to the /workspaces/current folder. This folder was populated by the setup-workspaces.sh script in the previous step. (The launch.sh command configures a named volume to preserve the contents of the /workspaces directory between container images so you don't need to re-run setup-workspaces.sh again unless you want to configure a different environment.)
3. Run ./apply-all.sh to kick off the automation. The script will apply each layer in order. Cloud Pak fo Integration deployment will run asynchronously in the background and may require up to 90 to 100 minutes to complete.
4. You can check the progress by looking at two places, first look in your github repository. You will see the git repository has been created based on the name you have provided. The Cloud Pak for Integration install will populate this with information to let OpenShift GitOps install the software. The second place is to look at the OpenShift console, Click Workloads->Pods and you will see the GitOps operator being installed. You can also check the progress of the deployment by opening up Argo CD (OpenShift GitOps). From the OpenShift user interface, click on the Application menu 3x3 Icon on the header and select **Cluster Argo CD** menu item.)
5. Once deployment is complete, go back into the OpenShift cluster user interface and navigate to view Routes for the cp4i-pn namespace. Here you can see the URL to the deployed Platform Navigator instance. Open this url in a new browser window.



6. Navigate to Secrets in the ibm-common-services namespace, and find the platform-auth-idp-credentials secret. Copy the value of password key inside of that secret.
7. Go back to the PlatformNavigator instance that you opened in a separate window. Log in using the username admin with the password copied in the previous step.

THIS CONCLUDES THE CLOUD PAK FOR INTEGRATION INSTALLATION STEPS