



Objective

This document is to be used to automate a Designer Cloud v9.2 Customer Managed deployment in AWS on RHEL 8.6 using Ansible.

1: Login to AWS Console -> go to EC2

The screenshot shows the AWS Console Home page. At the top left, there's a sidebar with 'Recently visited' services: EC2 (highlighted with a red box), S3, AWS Budgets, AWS Organizations, WorkSpaces, CloudShell, and IAM. Below this is a 'View all services' link. To the right, there's a 'Welcome to AWS' section with links for 'Getting started with AWS', 'Training and certification', and 'What's new with AWS?'. In the center, there are two cards: 'AWS Health' (0 open issues, 0 scheduled changes) and 'Cost and usage' (No cost and usage). At the bottom, there are links for 'Feedback', 'Language selection', 'Privacy', 'Terms', and 'Cookie preferences'.

2. Click “Launch an instance”

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'New EC2 Experience' (selected), 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Tags', 'Limits', and 'Instances' (selected). Under 'Instances', there are links for 'Instances' (18), 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', and 'Reserved Instances'. The main area shows a table of 18 instances with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Available. A search bar at the top of the table has 'Search' entered. At the top right of the table, there are buttons for 'Launch instances' (highlighted with a red box), 'Connect', 'Instance state', 'Actions', and pagination controls. Below the table, there's a 'Select an instance' dropdown menu.



3. Under Names and Tags – add a name for your EC2 machine

EC2 > Instances > Launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name

pi-ansible-test-5

Add additional tags

4. Search for AMI: **ami-08970fb2e5767e3b8**

In the Application and OS Images Amazon Machine Image (AMI). As you can see in the screenshot below, this is for RHEL 8.6.

Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search bar: ami-08970fb2e5767e3b8

Filter categories:

- Quickstart AMIs (1) Commonly used AMIs
- My AMIs (0) Created by me
- AWS Marketplace AMIs (1) AWS & trusted third-party AMIs
- Community AMIs (1) Published by anyone

Refine results sidebar:

- Clear all filters
- Free tier only Info
- OS category
 - All Linux/Unix
 - All Windows
- Architecture
 - 64-bit (Arm)
 - 32-bit (x86)
 - 64-bit (x86)
 - 64-bit (Mac)
 - 64-bit (Mac-Arm)

Search results for "ami-08970fb2e5767e3b8":

Red Hat	Red Hat Enterprise Linux 8 (HVM), SSD Volume Type ami-08970fb2e5767e3b8 (64-bit (x86)) / ami-0bb199dd39edd7d71 (64-bit (Arm)) Red Hat Enterprise Linux version 8 (HVM), EBS General Purpose (SSD) Volume Type Platform: rhel Root device type: ebs Virtualization: hvm ENA enabled: Yes	Select
<input checked="" type="radio"/> 64-bit (x86)		<input type="radio"/> 64-bit (Arm)

The following results for "ami-08970fb2e5767e3b8" were found in other categories

- 1 results in AWS Marketplace AMIs
- 1+ results in Community AMIs

AWS Marketplace AMIs are AMIs that are published by AWS & trusted third-parties

Community AMIs are AMIs that are shared by the general AWS community



5. Pick instance type (recommend using m5.2xlarge)

▼ Instance type [Info](#)

Instance type

m5.2xlarge

Family: m5 8 vCPU 32 GiB Memory
On-Demand Linux pricing: 0.384 USD per Hour
On-Demand Windows pricing: 0.752 USD per Hour

[Compare instance types](#)

6. Use an existing keypair or create one.

This will be used to SSH into the instance as well as connect to the instance from Ansible.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

pi-keypair [▼](#) [Create new key pair](#)

7. Network

1. Select your VPC – (ie; vpc-028790bed3b4c1e33 Engineering)
2. Subnet needs to be public for the EC2 instance to get a public IP address. (For testing, this will auto-assign the correct subnet based on choosing your VPC).
3. Security Group (virtual firewall) should be configured such that the instance can be reached on port 22 (SSH) and 3005 (TCP HTTP for the UI).
*(**May need to create a new Security Group (SG))*

8. Configure Storage

Change value to 1x 100 GiB

▼ Configure storage [Info](#) [Advanced](#)

1x GiB [gp2](#) [▼](#) Root volume

i Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage [X](#)

[Add new volume](#)

0 x File systems [Edit](#)



9. Click on “**Advanced details**” to expand this section. (Admin may need to select from an existing IAM instance profile.)

Purchasing option [Info](#)
 Request Spot Instances
Request Spot Instances at the Spot price, capped at the On-Demand price

IAM instance profile [Info](#)
pi-cluster-iam-role
arn:aws:iam::163305015547:instance-profile/pi-cluster-iam-role

Create new IAM profile [\[\]](#)

Hostname type [Info](#)
IP name

If you create a new IAM profile: (ie; Selection of an existing one in the screenshot above – **pi-cluster-iam-role**)

Note: If you don't create an instance role upon launching the EC2 instance, **you can't add one later**. You can however modify the attached role and associated policies after the fact.

Create or use a policy: (see screenshot below for selected radio buttons and click “Next”)



IAM > Roles > Create role

Step 1

Select trusted entity

Step 2

Add permissions

Step 3

Name, review, and create

Select trusted entity

Trusted entity type

AWS service

Allow AWS services like EC2, Lambda, or others to perform actions in this account.

AWS account

Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

Web identity

Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

SAML 2.0 federation

Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

Custom trust policy

Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

EC2

Allows EC2 instances to call AWS services on your behalf.

Lambda

Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:

Choose a service to view use case ▾

[Cancel](#)

[Next](#)

(Example Policy: note to replace items in red for your deployment)



```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "VisualEditor0",  
            "Effect": "Allow",  
            "Action": [  
                "s3>ListStorageLensConfigurations",  
                "s3>ListAccessPointsForObjectLambda",  
                "s3:GetAccessPoint",  
                "s3>PutAccountPublicAccessBlock",  
                "s3>GetAccountPublicAccessBlock",  
                "s3>ListAccessPoints",  
                "s3>PutAccessPointPublicAccessBlock",  
                "s3>ListJobs",  
                "s3>PutStorageLensConfiguration",  
                "s3>ListMultiRegionAccessPoints",  
                "s3>CreateJob",  
                "s3>ListBucket"  
            ],  
            "Resource": "arn:aws:s3:::replace-with-your-bucket"  
        },  
        {  
            "Sid": "VisualEditor1",  
            "Effect": "Allow",  
            "Action": "s3:*",  
            "Resource": [  
                "arn:aws:s3:::replace-with-your-bucket",  
                "arn:aws:s3:::replace-with-your-bucket/*"  
            ]  
        }  
    ]  
}
```



10. Click Launch Instance

The screenshot shows the 'Launch instance' wizard in the AWS Management Console. On the left, there's a sidebar with 'Specify CPU options' and several dropdown menus for 'Metadata accessible', 'Metadata version', 'Metadata response hop limit', 'Allow tags in metadata', and 'User data'. A note at the bottom says 'User data has already been base64 encoded'. On the right, under 'Summary', it shows 'Number of instances' set to 1. Below that, it lists the 'Software Image (AMI)' as Red Hat Enterprise Linux 8 (ami-08970fb2e5767e3b8), the 'Virtual server type (instance type)' as m5.large, the 'Firewall (security group)' as pi-trifacta-sg, and the 'Storage (volumes)' as 1 volume(s) - 30 GiB. At the bottom right, there are 'Cancel' and 'Launch instance' buttons, with the 'Launch instance' button being highlighted with a red box.

11. Create or use an existing S3 bucket:

1. Create bucket

The screenshot shows the AWS S3 Buckets list. At the top, there's a header with 'Buckets (77)' and a 'Create bucket' button. Below that is a search bar with 'Find buckets by name'. The main table has columns for 'Name', 'AWS Region', 'Access', and 'Creation date'. One row is visible, showing a bucket named 'sandbox-tfstate' in the US West (Oregon) region (us-west-2), with access set to 'Bucket and objects not public' and created on June 6, 2022, at 15:24:08 (UTC-04:00).

Name	AWS Region	Access	Creation date
sandbox-tfstate	US West (Oregon) us-west-2	Bucket and objects not public	June 6, 2022, 15:24:08 (UTC-04:00)

2. Give a name (globally unique) and select the same region as the deployed EC2 instance.



Create bucket Info

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name
 Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.
[Choose bucket](#)

3. Click “Create Bucket”

We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose [Provide feedback](#).

Disable
 Enable

Tags (0) - optional
Track storage cost or other criteria by tagging your bucket. [Learn more](#)

No tags associated with this bucket.
[Add tag](#)

Default encryption
Automatically encrypt new objects stored in this bucket. [Learn more](#)

Server-side encryption
 Disable
 Enable

Advanced settings

After creating the bucket you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) **Create bucket**

4. Then find the Bucket you created:

Buckets (78) Info

Buckets are containers for data stored in S3. [Learn more](#)

[Empty](#) [Delete](#) **Create bucket**

X 1 match < 1 >

Name	AWS Region	Access	Creation date
my-silly-example-bucket	US West (Oregon) us-west-2	Bucket and objects not public	July 11, 2022, 15:39:50 (UTC-04:00)



5. Click “Permissions”

my-silly-example-bucket [Info](#)

Objects (0)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[C](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions ▾](#)

[Create folder](#) [Upload](#)

Find objects by prefix

< 1 > [Filter](#)

	Name	Type	Last modified	Size	Storage class
No objects					
You don't have any objects in this bucket.					
Upload					

6. Click “Edit” under Bucket Policy:

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

[Edit](#) [Delete](#)

i Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using [Amazon S3 Block Public Access](#)

```
{  
  "Version": "2012-10-17",  
  "Id": "Policy1578435424058",  
  "Statement": [  
    {  
      "Sid": "Stmt1578435417646",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::163305015547:role/pi-cluster-iam-role"  
      },  
      "Action": [  
        "s3:ListBucket"  
      ]  
    }  
  ]  
}
```

[Copy](#)



Example: note that the ARN is what we created earlier (the IAM Role for the Trifacta instance)

The bucket should be replaced with the bucket you created. You'll want to replace the red text with the arn number (**63305015547**), IAM Instance Profile (**cluster-iam-role**), and S3 bucket (**test-bucket**).

```
{  
    "Version": "2012-10-17",  
    "Id": "Policy1578435424058",  
    "Statement": [  
        {  
            "Sid": "Stmt1578435417646",  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::163305015547:role/ cluster-iam-role"  
            },  
            "Action": [  
                "s3>ListBucket",  
                "s3:GetBucketLocation"  
            ],  
            "Resource": "arn:aws:s3::: test-bucket"  
        },  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::163305015547:role/ cluster-iam-role"  
            },  
            "Action": "s3:*",  
            "Resource": [  
                "arn:aws:s3::: test-bucket",  
                "arn:aws:s3::: test-bucket/*"  
            ]  
        }  
    ]  
}
```



-----[CHECKPOINT]-----

So, at this point you have an EC2 instance, with an instance role, an S3 bucket and a policy that allows the role we created the ability to read/write to S3... but now we need to install Designer Cloud and configure it.

Install Ansible, it will help make a vanilla install and configuration easier. If you don't have Python, PIP, or Ansible on your machine then [click here](#) to follow setup instructions for these.

What you will need:

1. Public IP address (from AWS Console EC2)

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation links like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances, Images, and Elastic Block Store. The Instances link is expanded, showing sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, and Elastic Block Store. The main content area shows a table of instances. One instance, 'demo.aws.se.trifacta.net', is selected and highlighted with a blue border. Below the table, there's a detailed view for this specific instance. The 'Details' tab is active, and under the 'Instance summary' section, the 'Public IPv4 address' field is highlighted with a red box and contains the value '34.220.230.194 | open address'. Other fields in this section include 'Instance ID' (demo.aws.se.trifacta.net), 'IPv6 address' (empty), 'Hostname type' (empty), 'Private IP DNS name (IPv4 only)' (empty), and 'Instance type' (m5.4xlarge). To the right of the summary table, there are sections for 'Private IPv4 addresses', 'Public IPv4 DNS', and 'Elastic IP addresses'.

2. Pem key (created earlier) – remember where this was saved locally
3. Name of the S3 bucket Designer Cloud will be using
4. A license.json file (place the Designer Cloud license file in the ansible playbook directory - i.e. same location as the playbook.yml file location)



1. Edit the hosts file (change the IP and pem file location)

```
[trifacta]
trifacta_node ansible_host=34.220.37.104
~  

[trifacta:vars]
ansible_ssh_private_key_file=/Users/ppayandeh/Trifacta/keys/pi-keypair.pem
ansible_user=ec2-user
~  

~
```

2. The configuration step:

```
.  
  config.txt ← copy the example/reference templates for the desired environment  
  connect.sh  
  hosts  
  license.json  
  triconf_templates  
    └── AWS_S3_only_config.template  
      └── Azure_ADLS_only_config.template  
  trifact_92_rhel7_playbook.yml  
  trifact_92_rhel8_playbook.yml  
  update_conf.py  
  
1 directory, 9 files
```

The purpose of having a triconf_templates folder with a series of template files, is to create an inventory of different deployment scenarios. The goal is to minimize the scope of what needs to change in order to have trifacta-conf.json which is 1000+ lines of configuration and reduce it to: “what minimally needs to change for my deployment scenario to work?”. In our case with this document, we want to configure an AWS deployment with S3, so we copy the AWS_S3_only_config.template into config.txt (screenshot above). The file being named config.txt is just the convention and is expected from the Ansible playbook.

```
# IMPORTANT: Don't edit the file directly
#
# If you are doing an AWS deployment with S3 only, copy this file to config.txt
# cp AWS_S3_only_config.template config.txt
#
# Then change only the bucket you would like to configure Cloud Designer to use.
# If a line starts with #, it is a comment and will not alter the trifacta configuration.
# the JSON path is case sensitive so AWS.S3.BUCKET.NAME is not the same as aws.s3.bucket.name

# user config section

aws.s3.bucket.name = "bucket_name"

# END user config section

# SYSTEM CONFIG: do not edit
hdbs.enabled = False
webapp.storageProtocol = "s3"
aws.credentialProvider = "instance"
```



The only thing that needs to change for our simple AWS + S3 deployment is the S3 bucket name (aws.s3.bucket.name).

3. Place the license file in the directory where you unpacked the ansible configuration. It should look like this:

```
Trifacta/ansible/install_92 via v3.8.7 on eu-west-1 on ppayandeh@trifacta.com
> ls
config.txt           hosts          trifconf_templates/
connect.sh*          license.json   trifact_92_rhel7_playbook.yml  update_conf.py
```

4. Then run the playbook:

For Centos 8:

```
ansible-playbook -i hosts trifacta_92_rhel8_playbook.yml
```

Centos 7:

```
ansible-playbook -I hosts trifacta_92_rhel7_playbook.yml
```

Which will output something like the following:



```
> ansible-playbook -i hosts playbook.yml

PLAY [all] ****
TASK [Gathering Facts] ****
The authenticity of host '34.220.37.104 (34.220.37.104)' can't be established.
ECDSA key fingerprint is SHA256:ZJLbnTNMAB=cN5NgwDwRuyk1Mmd1u1VreysKGUL6o.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
ok: [trifacta_node]

TASK [Create /root/trifacta_temp if it does not exist] ****
changed: [trifacta_node]

TASK [Download dependencies file (trifacta-server-deps-9.2.0-centos8.tar.gz)] ****
changed: [trifacta_node]

TASK [Download trifacta RPM (trifacta-server-9.2.0-184.centos8.x86_64.rpm)] ****
changed: [trifacta_node]

TASK [Unarchive (trifacta-server-deps-9.2.0-centos8.tar.gz)] ****
changed: [trifacta_node]

TASK [copy local.repo] ****
changed: [trifacta_node]

TASK [trifacta repo directory status] ****
ok: [trifacta_node]

TASK [trifacta repo directory status] ****
ok: [trifacta_node]

TASK [Move trifacta repo directory] ****
changed: [trifacta_node]

TASK [install python3] ****
changed: [trifacta_node]

TASK [install java 8 from local repo] ****
changed: [trifacta_node]

TASK [postgress dnf install] ****
changed: [trifacta_node]

TASK [postgress client dnf install] ****
ok: [trifacta_node]

TASK [install nodejs, nginx] ****
changed: [trifacta_node]

TASK [install trifacta] ****
changed: [trifacta_node]

TASK [copy license file] ****
changed: [trifacta_node]

TASK [encrypt key] ****
changed: [trifacta_node]

TASK [Check if PostgreSQL database is initialized.] ****
ok: [trifacta_node]

TASK [init db] ****
changed: [trifacta_node]

TASK [backup /var/lib/pgsql/12/data/pg_hba.conf /var/lib/pgsql/12/data/pg_hba.conf.bak] ****
changed: [trifacta_node]

TASK [cp /opt/trifacta/bin/setup-utils/db/pg_hba.conf.SAMPLE /var/lib/pgsql/12/data/pg_hba.conf] ****
changed: [trifacta_node]

TASK [Start service postgresql-12, if not running] ****
changed: [trifacta_node]

TASK [initialize the database] ****
changed: [trifacta_node]

TASK [Start service postgresql-12, if not running] ****
skipping: [trifacta_node]

TASK [change permissions to fix issue [JIRA]] ****
changed: [trifacta_node]

TASK [copy update_conf.py] ****
changed: [trifacta_node]

TASK [run update conf] ****
changed: [trifacta_node]

TASK [Start trifacta service, if not running] ****
changed: [trifacta_node]

PLAY RECAP ****
trifacta_node      : ok=26   changed=22   unreachable=0   failed=0    skipped=1    rescued=0    ignored=0
```



Now you need to open a browser and point it at: **http:<designer-cloud ip address>:3005**

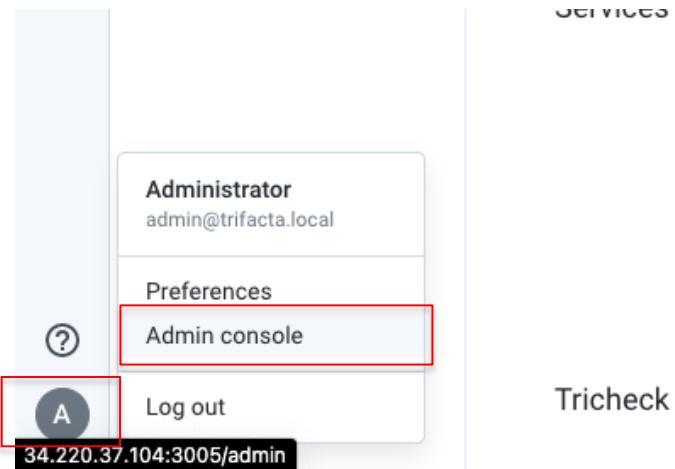
NOTE: The Designer Cloud IP Address is the Public IP address from AWS Console EC2.

** The default username and password is:

Username: admin@trifacta.local

password : admin

1. In the Designer Cloud UI, click the “A” on the very bottom left side of the UI -> then click “Admin console”





2. Then click “Admin settings”

The screenshot shows the Designer Cloud Admin console interface. On the left, there is a sidebar with various icons and links: Admin console, Users, Roles, Workspace settings, Admin settings (which is highlighted with a red box), Environment parameters, and OAuth 2.0 clients. The main panel is titled "Users" and shows a table with columns for All, Enabled, and Disabled. A single user entry is visible: "Administrator".

3. In the search bar under Platform Settings, search for: aws.credential (to filter)
Change The aws.credentialProvider from “default” to “instance” and click “Save”.
This will take a few minutes to restart the Designer Cloud service. When the service comes back online, the issue should be fixed and you should have access to S3.

The screenshot shows the Platform Settings page. A search bar at the top contains the query "aws.credential". Below it, there are two configuration items: "aws.emr.forceInstanceRoleForSTS" and "aws.emr.forceInstanceRole". The third item, "aws.credentialProvider", is highlighted with a red box. Its dropdown menu shows three options: "default", "instance" (which is selected and highlighted with a red box), and "temporary". To the right of the dropdown, there is a detailed description of each option. At the bottom right of the page is a "Save" button.



4. If you go to Library (little icon on the far left side of the UI) and click on “Import Data”

The screenshot shows the Designer Cloud Library interface. On the left is a sidebar with icons for Library, All Data, Imported Datasets, References, Macros (which is highlighted with a red box), and other unlabelled icons. The main area has a title "All Data" with a sub-section "Import Data" highlighted with a red box. Below this are tabs for "All", "Owned by me", and "Shared with me". A table header "Name" is followed by columns "Owner" and "In flows". A message at the bottom says "You don't have any data yet".

5. You should see your S3 bucket like the following:

The screenshot shows the "Import Data" screen. On the left is a sidebar with icons for Import Data, All Data (highlighted with a red box), Imported Datasets, References, Macros, and other unlabelled icons. The main area has a search bar "Search..." and an "Upload" button. A section titled "Choose a file or folder" shows an "S3" entry with a red box around it. Below this is a search bar "Search...". A table lists datasets with columns "NAME", "SIZE", and "LAST UPDATED". One entry is shown: "pi-trifacta-test-bucket". To the right, a panel shows "0 New Datasets" and a message "Choose data to import.".