

# Deploying Metric Insights in an Orchestrated Environment

**i** Beginning in version 6.0, Metric Insights can be deployed using a container orchestration platform. Container orchestration allows for *horizontal scaling* (versus vertical) as well as a *highly-available* architecture. Orchestration platforms include:

- Kubernetes
- Amazon ECS (CloudFormation, Terraform)
- Docker Swarm
- OpenShift

Metric Insights consists of services running inside of containers:

- Web
- Data Processor
- Seed
- Data Analyzer
- Monitoring

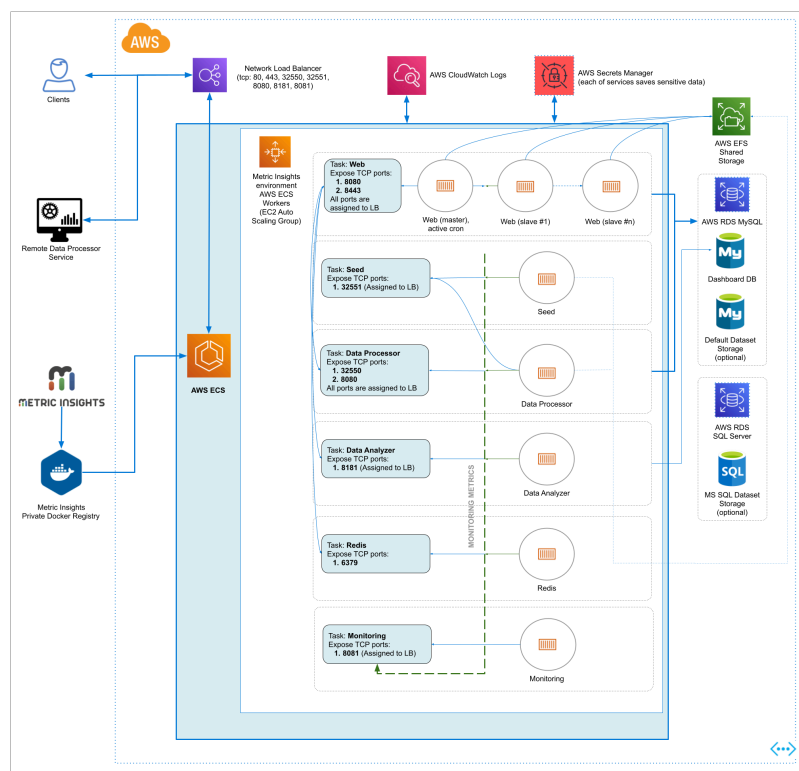
Deployment process:

## [Metric Insights Application Architecture in AWS ECS](#)

1. [Prerequisites](#)
2. [Obtain Docker Registry Credentials](#)
3. [Choose a Deployment Method \(Kubernetes, Amazon ECS, Docker Swarm\)](#)
4. [Generate the configuration file to deploy to Amazon ECS](#)
5. [Create the ECS Stack with AWS CloudFormation](#)
6. [Accessing the Metric Insights Deployment](#)
7. [Resources involved in running Metric Insights in ECS](#)

**⚠** Starting v6.4.1, services inside containers are run by one unprivileged user. The ownership for the network shared volume must be manually set to `www-data`, `uid: 33` before updating to v6.4.1.

# Metric Insights Application Architecture in AWS ECS




## 1. Prerequisites

**i** Ensure that the [system requirements for a Metric Insights server](#) are met.


To deploy Metric Insights across the different orchestration platforms, the following architectural pieces are required:

- **kubectl** command-line tool to manage a Kubernetes deployment (not required for ECS and Docker Swarm)
- **oc** command-line tool to manage an OpenShift deployment
- **Remote Database Server** to host the application database:
  - MySQL v8+
  - (MySQL/MariaDB v5.5+ is supported only in MI versions *prior to* v6.2.0)
- **Persistent shared storage** to store the application file system
  - e.g., NFS, Portworx, EFS, etc.
- Specific ports open on the network:
  - 80, 443: HTTP and HTTPS ports for the UI Application Service (by default redirection to 443)
  - 2550: TCP port for the Data Processor cluster within the kubernetes namespace

- 2551: TCP port for the Seed service within the kubernetes namespace
- 32550: TCP port for external access to the Data Processor cluster
- 32551: TCP port for external access to the Seed service
- 3306: MySQL port to get access from outside
- 8080,8443: HTTP and HTTPS ports for the REST API Data Processor Service (only one port is enabled at one time)
- 8081: TCP port for Monitoring Tool

 For non-orchestrated deployments, see the help article on using [Simple Installer](#).


## 2. Obtain Docker Registry Credentials


 Contact MI Support for access to the official Metric Insights Docker Registry. Credentials are needed to pull docker images for each Metric Insights' service.

- Note: the default MI Docker Registry address ([docker.metricinsights.com](https://docker.metricinsights.com)) is specified in the deployment configuration file for each orchestration type.

If you must use a **Private Docker Registry** instead, see [Uploading Metric Insights Docker Images to a Private Registry](#) about how to download our docker images and upload them to your private registry.

## 3. Choose Deployment Method (Kubernetes, Amazon ECS, Docker Swarm)

 If deploying to **Kubernetes**, please see [Deploying Metric Insights on Kubernetes](#).

 If deploying to **Docker Swarm**, please see [Deploying Container Orchestration with Docker Swarm](#).

 If deploying to **OpenShift**, please see [Deploying Metric Insights on OpenShift v3](#).

 If deploying to **Amazon ECS**, continue below.


### Amazon ECS Prerequisites:

1. Database (RDS or EC2 instance with custom database deployment)
2. EFS or custom NFS shared storage
3. **Optional:** If utilizing a private registry; i.e., non-Metric Insights, ensure that you have those credentials available.

## 4. Generate Configuration File to Deploy to Amazon ECS

The configuration file can be generated using the Metric Insights installer package:

1. Download the installer package to a Linux system and unpack
2. Change into the installer directory, then run the installer with the **ecs** command and specify a target filename to generate the configuration file:
  - If the remote DB server has the same timezone as MI app: `./installer.py ecs --timezone <MI app timezone> -o <manifest filename>.json`
  - If the remote DB server has a different timezone than MI app: `./installer.py ecs --timezone <MI app timezone> --mysql-timezone <remote database server timezone> -o <filename>.json`
3. The configuration file can now be used as a template with AWS CloudFormation to create and deploy the Metric Insights environment

 **Note:** Run `./installer.py ecs --help` to see the list of available installer options. See [Basic Console Commands](#) section.

## 5. Create the ECS Stack with AWS CloudFormation

Prepare the following:

1. RDS address with root credentials
2. EFS address to connect to Metric Insights application

Apply the configuration file through the CloudFormation UI:

1. Upload the generated json file as a template

CloudFormation > Stacks > Create stack

Step 1  
Specify template

Step 2  
Specify stack details

Step 3  
Configure stack options

Step 4  
Review

## Create stack

### Prerequisite - Prepare template

**Prepare template**  
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

☒ Template is ready ☐ Use a sample template ☐ Create template in Designer

### Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

**Template source**  
Selecting a template generates an Amazon S3 URL where it will be stored.

☐ Amazon S3 URL 1. ☒ Upload a template file

2. Upload a template file  
 `metricinsights-6.1.0.json`  
 JSON or YAML formatted file

S3 URL: `https://s3.us-east-2.amazonaws.com/cf-templates-20k6o5ky29hn-us-east-2/2019296d75-metricinsights-6.1.0.json`

3.

2. Complete each field then click [Next] at the bottom of the page. Some key notes:

- To generate passwords for each service, you can either run `echo -n '<pwd>' | base64` to encode a password of your choice, or run something like `openssl rand -base64 8` to auto generate a password for you.
- Use the full RDS address for the field "DBHostName"
- Enter the RDS root user in the field "DBRootUserName"
- Enter the full EFS address in the field "NFSServerAddress"
- Select all Subnet IDs available in the field "SubnetIDs"
- The field "WebReplicationsCount" represents the number of web slave containers (secondary to web master).

CloudFormation > Stacks > Create stack

Step 1  
Specify template

Step 2  
**Specify stack details**

Step 3  
Configure stack options

Step 4  
Review

### Specify stack details

**Stack name**

Stack name

miECS2

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

**Parameters**

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

**AllowAccessForSubnet**  
Specify the network that Metric Insights application has to listen.

0.0.0.0/0 1

**DBDataProcessorPassword**  
\*\*\*\*\*

**DBDatasetReadPassword**  
\*\*\*\*\*

**DBGeneratorPassword**  
\*\*\*\*\*

**DBHostName**  
Specify remote database server to connect Metric Insights Application.

dashboard.csbdzi7yeiqd.us-east-2.rds.amazonaws.com

3. Click **[Next]** to skip through the subsequent pages until you reach the window shown below.
  - Click the checkbox to acknowledge that IAM resources might be created on deployment
  - Click **[Update stack]**

**Capabilities**

ⓘ The following resource(s) require capabilities: [AWS::IAM::Role]  
 This template contains Identity and Access Management (IAM) resources that might provide entities access to make changes to your AWS account. Check that you want to create each of these resources and that they have the minimum required permissions. [Learn more](#)

☒ I acknowledge that AWS CloudFormation might create IAM resources.

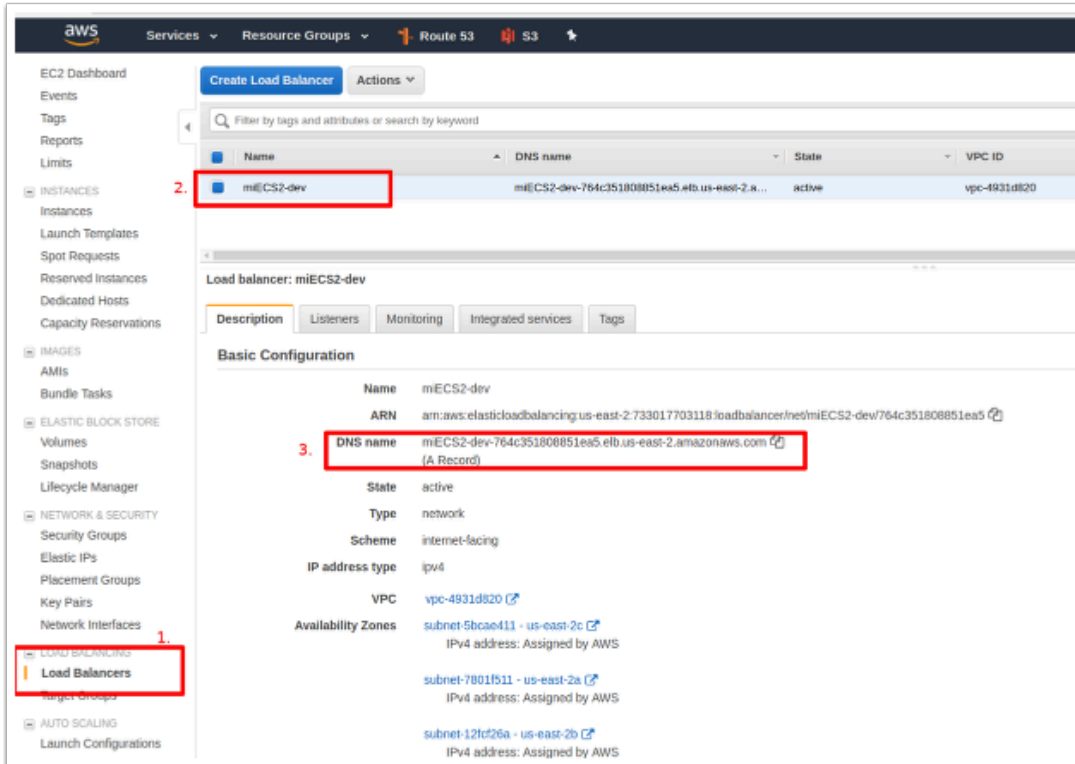
Cancel Previous View change set **Update stack**

4. Allow incoming connections to RDS for new ECS/EC2 security group to complete the deployment
  1. As the new ECS Stack is being deployed, go to the EC2 Console and select one of the new EC2s created for ECS
  2. Go to the Security Group field and select on the new security group name
  3. Copy the Group ID; e.g., "sg-name"
  4. Switch to the RDS Console and select the RDS instance being used for ECS
  5. Go to the VPC Security Group field and select the security group name
  6. Switch to the "Inbound" tab and click **[Edit]**
  7. Add the new EC2 security group to the list and then **[Save]**:
    - Add Rule > All Traffic > Paste Group ID

On adding the group, switch back to CloudFormation to monitor the ECS Stack deployment. The deploy should complete in 5-10 minutes.

**i** If you need to customize some processes after the application deployment, see [Configuring Custom Components](#) article.

## 6. Accessing Metric Insights Deployment



Once the ECS Stack is deployed:

1. Switch back to the EC2 Console and select "Load Balancers" in the left menu pane.
2. Identify the Load Balancer DNS name to access the Metric Insights application in a browser.
3. For the best user experience, map the Load Balancer DNS name to a user-friendly name in Amazon Route 53.

Metric Insights is now deployed in ECS and browser ready.

## 7. Resources Involved in Running Metric Insights in ECS

- [AWS ECS Task Definitions](#)

- AWS ECS Cluster
- AWS ECS Services
- AWS EC2 Auto Scaling group
- AWS EC2 Launch Configuration
- AWS EC2 Security Groups
- AWS Target Groups
- AWS Network Load Balancer
- IAM Roles
- AWS Secret Manager
- AWS Cloud Formation (only for deployment and updates)

Non-ECS resources in AWS needed for deployment include:

1. AWS RDS instance based on MySQL 8.0.20
  - Requires a custom Parameter Group with *log\_bin\_function\_creators* enabled
  - See [this KB](#) for a list of mysql parameters to adjust
2. AWS EFS Shared Storage

## 8. Basic Console Commands

Basic console commands can be checked by running `./installer.py ecs --help`.



The following list of utilities are available to use on the host.

Note, all of these tools become available only if the Web Component is installed.

Optional Parameters	
<b>--registry REGISTRY</b>	Docker registry URL, that will be used for deploying MI components. Example: <hostname> or <hostname>:<port>. (default: None)
<b>--hostname HOSTNAME</b>	Web service additional hostname (default: None). The option is used to assign 127.0.0.1 to the Web container. It helps the Metric Insights application to interact with its internal components faster inside the container.