EDP 2.3.3 — Deployment Documentation

aicas GmbH
Emmy-Noether-Straße 9
D-76131 Karlsruhe, Germany

Friday 24th January, 2025

```
aicas GmbH                      phone  +49 721 663 968-0
Emmy-Noether-Straße 9           fax    +49 721 663 968-99
76131 Karlsruhe                 email  info@aicas.com
Germany                         web    http://www.aicas.com

aicas America Limited           phone  +1 203 359 5705
4023 Kennett Pike, Suite 810
Wilmington, DE 19807            email  info@aicas.com
USA                             web    http://www.aicas.com
```

Contents

1 Preface

The purpose of this document is to provide step by step instructions on how to setup an AWS infrastructure in order to deploy and configure EDP. The default configuration discussed here is suitable for small to moderate workloads.

1.1 Intended Audience

The intended audience of the current document are developers and/or DevOps engineers, familiar with AWS cloud, who want to set up the EDP web portal in association with an own AWS cloud account.

1.2 Objective and Scope

The current document covers the steps required to deploy EDP into an AWS cloud account. It does not cover technical knowledge of AWS cloud services. Please refer to the official AWS documentation to find technical information about the usage of the AWS cloud services.

2 Introduction

The deployment of EDP 2.3.3 consists of two major parts:

- Preparation of the AWS infrastructure for the deployment of EDP.
- Configuration and deployment of EDP components.

Firstly, we will start with the AWS infrastructure configuration for EDP. We will create all necessary resources and set the configuration required for EDP to function.

Then we will switch to configuring the EDP components, after which we will deploy them to the previously prepared AWS infrastructure.

In general, provided technical knowledge in AWS and the necessary infrastructure prerequisites, it will take approximately 8 hours to complete deployment.

3 Infrastructure Requirements and Configuration

The current section gives readers a brief overview of the AWS services used by EDP, and describes the steps required to configure the AWS cloud account for the EDP deployment.

3.1 License and Distribution Artifacts

In order to use EDP, clients must set up an AWS account. In case such an account is not at hand, please contact the administrator responsible for AWS accounts at your company, and get more information at https://aws.amazon.com/getting-started.

In their customer accounts, clients will receive from aicas the following components:

- A zip file containing the four docker images that constitute EDP's solution: backend, frontend, keycloak, and bundle-pipeline. The bundle-pipeline docker image includes the file `jamaica.aicas_key`, which is needed for running the aicas tool JAR Accelerator (see sections 4.3 and 4.13 for more information).

  Additionally, this zip file will also contain a preconfigured AWS CloudFormation template and the EDP 2.3.3 deployment documentation.
- A second zip file with the EDP client bundle.
- The user manual and the license agreement to the product.

3.2 AWS Services Overview

This section gives a brief overview of the AWS services used by EDP. Please follow the supplied links to find out more information on particular AWS services.

3.2.1 CloudFormation Template

CloudFormation is an AWS Infrastructure-as-Code (IaC) service. The deployment of EDP release 2.3.3 is based on a CloudFormation Template.

Usage: Used to simplifiy provisioning and management of EDP on AWS.

https://aws.amazon.com/cloudformation/resources/templates/

3.2.2 CloudWatch

Amazon CloudWatch is a monitoring service, that collects operational data in the form of logs, metrics, and events, providing an unified view of the resources, applications, and services that run on AWS and on-premises servers.

Usage: In EDP, it is mainly used to get detail logs related to Lambda and JAR acceleration.

https://aws.amazon.com/cloudwatch/

3.2.3 Route 53

Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. Route 53 effectively connects user requests to an infrastructure running in AWS and supports many useful features.

Usage: Used as DNS provider for EDP; it hosts EDP domain name and routes requests to the EDP server.

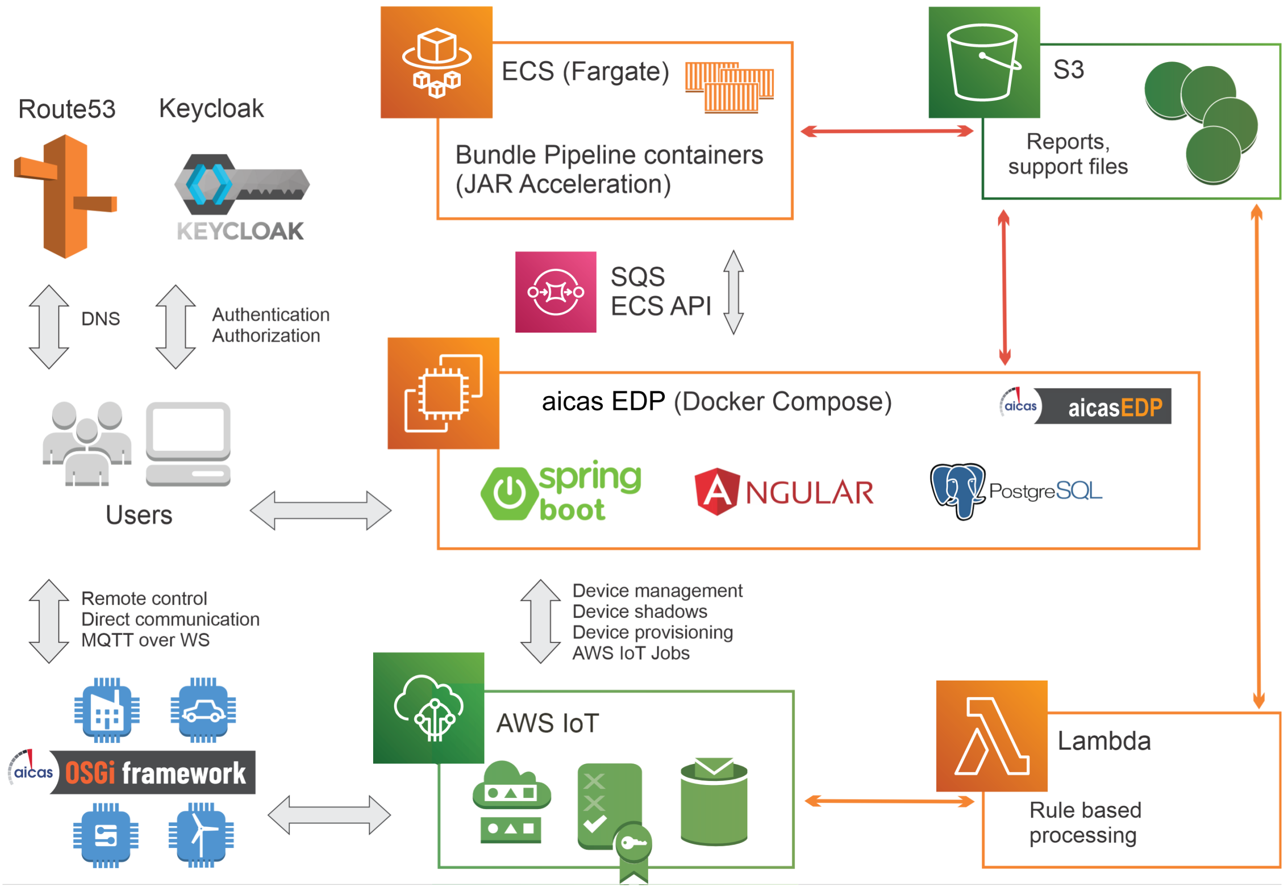


Figure 1: AWS cloud components that interact with EDP

https://aws.amazon.com/route53/

3.2.4 ECS

Amazon Elastic Container Service (ECS) is a container orchestration service that simplifies deployment, management, and scaling of containerized applications.

Usage: Each EDP component is deployed as an ECS service (EC2 instance). Each ECS service consists of one or more Docker containers.

https://aws.amazon.com/ecs/

3.2.5 ECR

Amazon Elastic Container Registry (ECR) is a fully-managed Docker container registry.

Usage: Used to store built and versioned container images of EDP components.

https://aws.amazon.com/ecr/

3.2.6 ELB

Amazon Elastic Load Balancing (ELB) automatically distributes incoming application traffic across multiple targets and virtual appliances in one or more Availability Zones (AZs).

Usage: Used to distribute incoming traffic.

https://aws.amazon.com/elasticloadbalancing/

3.2.7 DynamoDB

Amazon DynamoDB is a serverless, NoSQL document database service that offers managed backups and point-in-time recovery.

Usage: During EDP deployment, DynamoDB is used to create the configuration table.

https://aws.amazon.com/dynamodb/

3.2.8 SQS

Amazon Simple Queue Service (SQS) is a fully managed message queuing service.

Usage: Used as transport for async communication workflows inside EDP.

https://aws.amazon.com/sqs/

3.2.9 S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers scalability, data availability, security, and performance.

Usage: Used as storage for sensitive data that cannot be part of container images, such as private keys and certificates. Also used to store logs of EDP bundle processing.

https://aws.amazon.com/s3/

3.2.10 Lambda

AWS Lambda is a scalable serverless computing platform, easily integrated with AWS services.

Usage: Used to back AWS IoT rules (reacts to MQTT messages from devices) that conform to defined patterns and do some processing, like storing data in databases or generating security tokens.

https://aws.amazon.com/lambda/

3.2.11 IoT Core

AWS IoT Core is a managed cloud service that lets connected devices easily and securely interact with cloud applications and other devices.

Usage: Used as a backbone for secure device connections to EDP, device communication, device attributes management and storage.

https://aws.amazon.com/iot-core/

3.2.12 IoT Device Management

AWS IoT Device Management makes it easy to securely register, organize, monitor, and remotely manage IoT devices at scale.

Usage: Used as a backbone of EDP device management, batch execution of device tasks (jobs), device groups.

https://aws.amazon.com/iot-device-management/

3.2.13 IAM

AWS Identity and Access Management (IAM) enables you to manage access to AWS services and resources securely.

Usage: Mandatory service for any AWS cloud infrastructure.

https://aws.amazon.com/iam/

3.2.14 VPC

Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud, where you can launch AWS resources in a virtual network that you define.

Usage: Mandatory service for any AWS cloud infrastructure.

https://aws.amazon.com/vpc/

3.3 AWS Billable Services and Costs

For the deployment of EDP, the following billable services from AWS are necessary:

- EC2
- S3
- Route 53 (recommended for domain management)
- ELB
- SQS
- Lambda
- IoT Device Management

- VPC
- DynamoDB
- CloudWatch

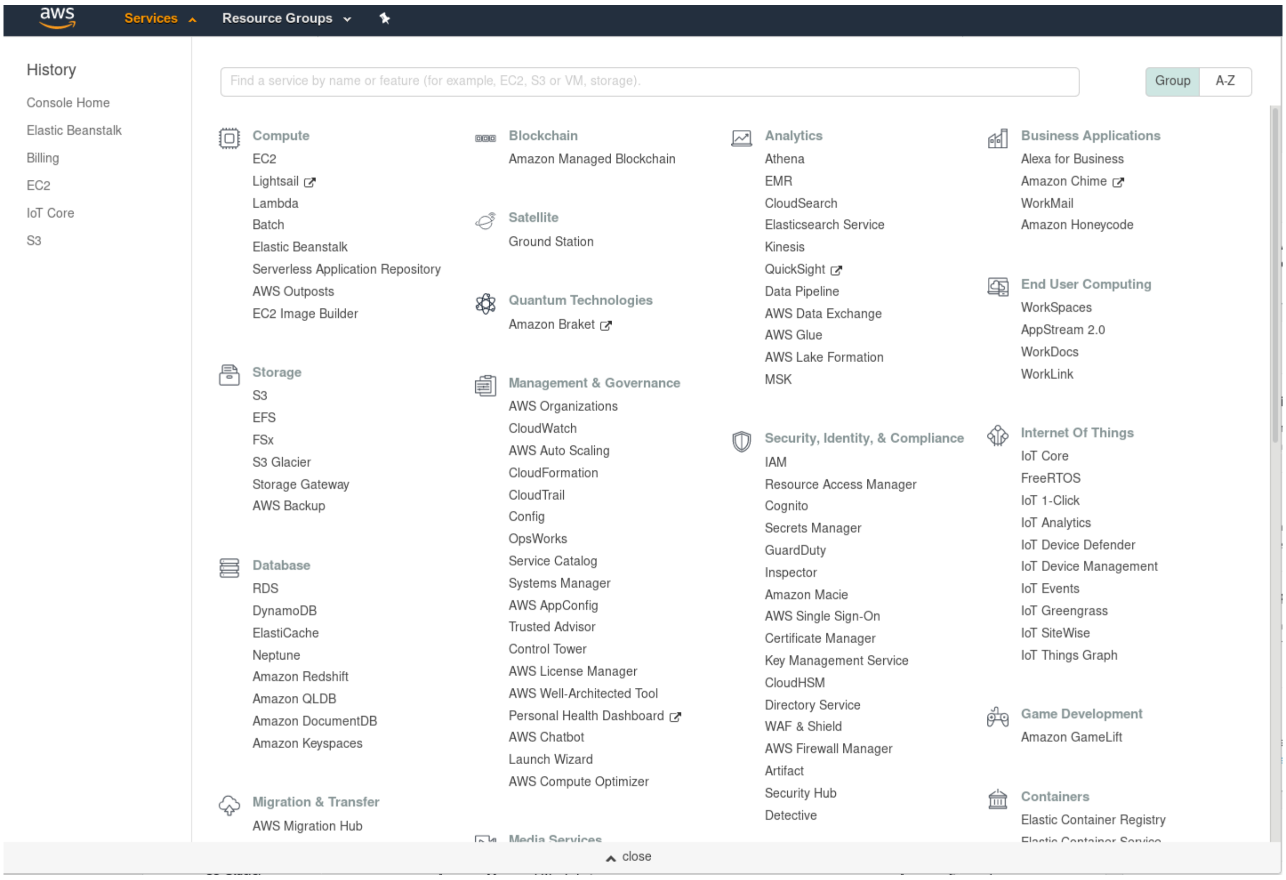
To estimate the costs of an architecture that would fit their business needs, customers can use the AWS pricing calulator in `https://calculator.aws/#/`.

3.4 Requirements

As pointed above, an AWS account is required to deploy EDP in the cloud. Please consider deploying into new AWS account because the permissions allow EDP access to read, edit and/or delete existing AWS resources in the AWS account.

3.4.1 Preparing AWS to Deploy EDP

Deploying EDP involves creating and configuring several instances of the services provided by AWS.



aws
Services
Resource Groups
Find a service by name or feature (for example, EC2, S3 or VM, storage).
Group
A-Z
close

History
Console Home
Elastic Beanstalk
Billing
EC2
IoT Core
S3

Compute
EC2
Lightsail
Lambda
Batch
Elastic Beanstalk
Serverless Application Repository
AWS Outposts
EC2 Image Builder

Storage
S3
EFS
FSx
S3 Glacier
Storage Gateway
AWS Backup

Database
RDS
DynamoDB
ElastiCache
Neptune
Amazon Redshift
Amazon QLDB
Amazon DocumentDB
Amazon Keyspaces

Migration & Transfer
AWS Migration Hub

Blockchain
Amazon Managed Blockchain

Satellite
Ground Station

Quantum Technologies
Amazon Braket

Management & Governance
AWS Organizations
CloudWatch
AWS Auto Scaling
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Systems Manager
AWS AppConfig
Trusted Advisor
Control Tower
AWS License Manager
AWS Well-Architected Tool
Personal Health Dashboard
AWS Chatbot
Launch Wizard
AWS Compute Optimizer

Analytics
Athena
EMR
CloudSearch
Elasticsearch Service
Kinesis
QuickSight
Data Pipeline
AWS Data Exchange
AWS Glue
AWS Lake Formation
MSK

Security, Identity, & Compliance
IAM
Resource Access Manager
Cognito
Secrets Manager
GuardDuty
Inspector
Amazon Macie
AWS Single Sign-On
Certificate Manager
Key Management Service
CloudHSM
Directory Service
WAF & Shield
AWS Firewall Manager
Artifact
Security Hub
Detective

Business Applications
Alexa for Business
Amazon Chime
WorkMail
Amazon Honeycode

End User Computing
WorkSpaces
AppStream 2.0
WorkDocs
WorkLink

Internet Of Things
IoT Core
FreeRTOS
IoT 1-Click
IoT Analytics
IoT Device Defender
IoT Device Management
IoT Events
IoT Greengrass
IoT SiteWise
IoT Things Graph

Game Development
Amazon GameLift

Containers
Elastic Container Registry

Figure 2: Several AWS services will be used to deploy EDP

3.4.2 Configure Work Environment

- Create an IAM user with administrative rights, if not already created. Usually this is done by the administrator of your AWS cloud account.
- Install and configure AWS CLI (https://aws.amazon.com/cli/). The created IAM user will be used to authenticate within AWS CLI.
- Choose region N. Virginia (us-east-1).

Please note that, although EDP has no restrictions concerning its deployment on any of the regions offered by AWS, at the moment CloudFormation is set to support region us-east-1. The choice of a different region would therefore imply considerable manual work in the creation of resources.

3.4.3 Software Requirements

In order to deploy EDP, it is necessary to have access to Docker, Keytool and AWS CLI.

4 Cloud Formation Deployment

As already mentioned, AWS CloudFormation is used as base for deploying EDP. In its version 2.3.3 EDP offers clients the option between two variants of deployment mode: a "classic" version, or an AMI-based version, meaning EDP accessed on AWS Marketplace.

Some elements of the CloudFormation template will have to be manually entered during deployment, and this could slightly differ, depending on the opted deployment mode. Section 4.6 explains all specifics.

Besides details about the distribution, this section provides step by step information on how to:

- configure S3 secret bucket,
- set DNS management,
- set a hosted zone record,
- configure the CloudFormation template,
- manage EDP-related secrets, and
- useful docker commands.

4.1 EDP Distribution

Clients of EDP "classic" will receive from aicas in their customer accounts two .zip objects. One of them will contain two pdf files with usage and deployment documentation, a preconfigured CloudFormation template (`cloudformation.yaml`), and four EDP docker images (Backend, Frontend, Keycloak, and Bundle-Pipeline), which they will need to download and unzip. This .zip file will also include a set of bootstraping scripts.

An `EDP client` zip file will also be delivered to the customers of the "classic" variant. This will have to be put into a "clients" folder, that must be previously created.

Further along the deployment process, users of EDP "classic" will also need to upload the whole content of the following directory tree structure into the S3 bucket (see section 4.3).

This tree structure is depicted below.

```
.
|-- clients
|   |-- EDP-Client-<version>.info
|   `-- EDP-Client-<version>.zip
|-- cloudformation.yaml
|-- docs
|   |-- EDP_Deployment-<version>.pdf
|   `-- EDP_Usage-<version>.pdf
|-- images
|   |-- backend.build.info
|   |-- bundle-pipeline.build.info
|   |-- edp.backend-<version>.tar.gz
|   |-- edp.bundle-pipeline-<version>.tar.gz
|   |-- edp.frontend-<version>.tar.gz
|   |-- edp.keycloak-<version>.tar.gz
|   |-- frontend.build.info
|   `-- keycloak.build.info
`-- scripts
    |-- bootstrap
    |   |-- bootstrap-provisioning-resources.sh
    |   |-- docker-compose.yml
    |   |-- env-backend.templ
    |   |-- env-frontend.templ
    |   |-- keycloak-set-up
    |   |   |-- front-realm-and-user.json
    |   |   |-- keycloak-backend-client.json
    |   |   |-- keycloak.py
    |   |   |-- password.json
    |   |   `-- super_admin_user.json
    |   |-- privilege-config.json
    |   `-- set-up.py
    `-- lambda
        |-- backuplambda.zip
        |-- bundle-url-pre-signer-v1.0.zip
        |-- fargate-anomaly-detected.zip
        `-- updatedockercompose.zip

7 directories, 29 files
```

Note that although the `EDP-Client-<version>.zip` is already present in the EDP “classic” distribution, for the users of AMI-based EDP the actual download of the EDP-Client needs to be activated by pressing the button “Get Client” in the menu area `Devices` of the EDP web portal.

For the AWS Marketplace clients, the same artifacts will be hosted in AMI, and the preconfigured `cloudformation.yaml` will already bring the AMI ID.

After the initial deployment procedures, all customers should move to the following steps.

4.2 Jarsigner Keystore File

Keytool is a command-line utility included with the Java Development Kit (JDK) that allows to manage keystores and key pairs (public and private keys).

- Download JDK 17
  Visit the official Oracle download page https://www.oracle.com/java/technologies/downloads/?er=221886#java17 and select the appropriate version for your operating system.
- Follow the installation instructions provided by Oracle
  https://docs.oracle.com/en/java/javase/17/install/overview-jdk-installation.html
- Open a terminal
- Use the “keytool” command to generate a key pair.

  ```
  keytool -genkeypair -alias 1 -keyalg RSA -keysize 2048 -validity 3650 \
  -keystore jarsigner-keystore.jks -storepass yourpassword \
  -keypass yourkeypassword -ext BC=ca:true \
  -ext KU=keyEncipherment,digitalSignature -ext EKU=serverAuth
  ```

- Press “Enter” and enter the following details
  `What is your first and last name?`
  e.g. jarsigner-ca
  `What is the name of your organizational unit?`
  e.g. all-environment
  `What is the name of your organization?`
  e.g. my-company
  `What is the name of your City or Locality?`

e.g. my-city
`What is the name of your State or Province?`
e.g. my-state
`What is the two-letter country code for this unit?`
e.g. my-country-code
`Is CN=jarsigner-ca, OU=all-environments, O=my-company, L=my-city, ST=my-state, C=my-country-code correct?`
e.g. yes

- Export the certificate from the keystore

```
keytool -exportcert -alias 1 -file jarsigner-keystore.cert \
-keystore jarsigner-keystore.jks -storepass yourpassword
```

- Convert the keystore to PKCS12 format

```
keytool -importkeystore -srckeystore jarsigner-keystore.jks \
-srcstorepass yourpassword -destkeystore ./jarsigner-keystore.p12 \
-deststoretype PKCS12 -deststorepass yourpassword
```

- Use “jarsigner-keystore.p12” in the next step.

4.3 Configure S3 Secret Bucket

- Log into AWS
- Go to Amazon S3
- Click on “Create Bucket” > “my-edp-secrets”
  `Please note that the name of the bucket must be unique.`
- Leave all default settings as they are
- Open “my-edp-secrets” bucket
- Create a folder “jarsigner”
- Upload “jarsigner-keystore.p12” into the “jarsigner” folder

For the deployment of the “classic” EDP variant, the additional steps below must be taken in order to create the necessary resources.

- Click on “Create Bucket” > “General purpose” > “aicas-edp-resources”
  `Please note that the name of the bucket must be unique.`
- Leave all default settings as they are
- Open the “aicas-edp-resources” bucket
- Click “Upload” > “Add folder”

Select and add each of the folders from the content of the extracted distribution files, so that the structure of the distribution tree is matched, as shown in section 4.1. Make sure the essential folders `clients`, `images` and `scripts` are uploaded, otherwise the deployment will fail.

4.4 Set DNS Management

- AWS web console > Route 53 > Domains > Registered domains
- Click on the "Register domains" button
- Follow the instructions provided by the "Register domains" wizard
- It will take some time for AWS to verify the provided contact information
- In order to check the status of the requested domain, please select Route 53 > Domains > Requests
- AWS Route 53 also creates a "Hosted zone" automatically
  `Successful status represents that domain is ready to use`
- AWS Route 53 > Hosted zones > "my-company-domain"
  `Please note that the console will show two record names, from types NS and SOA`

4.5 Creating a service-linked role

Admin of AWS account must create the ECS service-linked role via AWS Command Line Interface (CLI) to deploy EDP successfully.

- Open a terminal
- Configure AWS CLI with AWS account details using the following command
  `aws configure`
- Execute the following command to create the ECS service-linked role:
  `aws iam create-service-linked-role -aws-service-name ecs.amazonaws.com`

4.6 CloudFormation Template

EDP is offered in two deployment modes: "Classic" for S3-based deployment, and AMI-based deployment.

This section lists parameters which are included in the `cloudformation.yaml` file that clients receive in their customer accounts. These parameters can be entered or altered either by opening the `.yaml` file directly in an editor of choice or, more comfortably, in the AWS console.

Note that critical information like passwords can only be edited in the correspondent Cloud Formation template fields, in the AWS console.

By the configuration of the CloudFormation template, the first task will be to opt for a deployment mode. After that, just follow the steps as described below.

4.7 AMI-based Deployment

The first step in this process is for the admins of the AWS account to create an EC2ConnectKeyPairName via AWS Web Console. This is an important component, used for accessing the AWS instance running EDP, to check on potential problems. It is needed during the AMI-based deployment and below are the steps to be followed to generate it.

- AWS web console > EC2 > Network & Security > Key Pairs
- Click on the "Create key pairs" button
- Follow the instructions provided by the "Create key pair" wizard
- Select Key pair type as "RSA"
- Select Pricate key file format as ".pem"
- Click on the "Create key pair" button
  The newly created key pairs will be automatically downloaded and ready for immediate use. Be sure to save this key pair securely, as it cannot be re-downloaded or reattached to an EC2 instance later.

4.7.1 AMI Parameters

- *Environment:* `For deployment, environment is set by default to EDP-Production`
- *AccountID:* AccountID where CloudFormation stack will be created.
- *EC2ConnectKeyPairName:* EC2 key pair name which will be used to access EC2 instance via AWS CLI.
- *Region:* AWS region where CloudFormation stack will be created. Currently set for `us-east-1`.
- *HostedZoneId:* Hosted zone ID in AWS account where all DNS records will be created, from Route 53 service
  `AWS web console > Route 53 > Hosted zones > Hosted zone name > Hosted zone details > Hosted zone ID`
- *HostedZoneName:* Hosted zone name in AWS account where all DNS records will be created, from Route 53 service
  `AWS web console > Route 53 > Hosted zones > Hosted zone name > Hosted zone details > Hosted zone name`

- *InstanceType:* Possible selection for EC2 instance size
- *FrontendPrefix:* Prefix of the website URL. Must end with a dot "." symbol. A full URL consists of FrontendPrefix+HostedZoneName
  `For example "anycompany."` Please note that AWS imposes a limit, per AWS account per region, of 10 CA certificates with the same subject field. As FrontendPrefix is used in the naming scheme of the certificates, after 10 redeployments with the same prefix, it is expected that users will be affected by this restriction. A workaround in this case would be to modify the value of FrontendPrefix.
- *KeycloakPrefix:* Prefix of ther Keycloak site URL. Must end with a dot "." symbol. The full URL to the Keyckloak service will consist of KeycloakPrefix + HostedZoneName + /auth
  `For example "id."`
- *JARSignerKeyBucket:* Name of a bucket with Jarsigner key and JARA. The bucket must be in the same account.
  `AWS web console > S3 > Buckets > my-edp-secrets`
- *JARSignerKeyPath:* Path for Jarsigner key file in a bucket
  `Example: jarsigner/jarsigner-keystore.p12`, or just `jarsigner-keystore.p12`, if this file is in the root folder.
- *JARSignerKeyPassword:* Password for Jarsigner key file
  `This password is the one that was provided while creating the jarigner key-file (see in step 4.2).`
- *KeycloakPassword:* Password for admin keycloak user
  `Provide any secure password`
- *DBMasterPassword:* Root password of PostgreSQL database in EC2 instance
  `Provide any secure password`
- *EDPAccessCIDR:* CIDR range for EDP access. Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *SNSNotificationEmail:* E-mail address for SNS notification. Must be confirmed after the creation of the SNS topic. This email address is sent to and stored within your AWS account. This information will be used solely to send notifications about the start, stop or restart of EDP services.
  `For example "group@anycompany.com"`

4.7.2 Deployment via Marketplace

Please follow the steps below:

- AWS web console > AWS Marketplace > Manage Subscriptions

- Click on the “aicas Edge Device Portal” product from your subscribed product list.
- Click on the “Action” button > “Launch CloudFormation stack”
- Configure the Software accordingly
- Click on the “Continue to Launch” button
- Click “Next” to proceed
- Specify the “Stack details” accordingly
- Click “Next” to proceed
- In the “Configure stack details” step, leave all the settings as default and click “Next”
- In the “Review and Create” step, leave all the settings as default and check all the checkboxes under the “Capabilities and transforms” section.
- Click “Next” to proceed
- It will take several minutes to prepare and initialize all the necessary resources. Wait until the status of “stack-name” is “CREATE_COMPLETE”
- Cloudformation template will also create several IAM roles which are listed below.

Please skip the following section, which is dedicated to users who opted for the “classic” deployment mode.

4.8 Classic Deployment

As Deployment mode, select “Classic”. Then enter the parameters as follows.

- *S3BucketName:* `S3 Bucket containing the images, bootstrap scripts, documents, and client. Please leave this field empty if in AMI deployment mode.`
- *AMI ID:* `Please leave this field empty if in Classic deployment mode.`
- *Environment:* `For deployment, environment is set by default to EDP-Production`
- *Region:* AWS region where CloudFormation stack will be created. Currently set for `us-east-1`.
- *HostedZoneId:* Hosted zone ID in AWS account where all DNS records will be created, from Route 53 service
  `AWS web console > Route 53 > Hosted zones > Hosted zone name > Hosted zone details > Hosted zone ID`
- *HostedZoneName:* Hosted zone name in AWS account where all DNS records will be created, from Route 53 service
  `AWS web console > Route 53 > Hosted zones > Hosted zone name > Hosted zone details > Hosted zone name`

- *InstanceType:* Possible selection for EC2 instance size
- *FrontendPrefix:* Prefix of the website URL. Must end with a dot "." symbol. A full URL consists of FrontendPrefix+HostedZoneName
  `For example "anycompany."`
- *KeycloakPrefix:* Prefix of ther Keycloak site URL. Must end with a dot "." symbol. The full URL to the Keyckloak service will consist of KeycloakPrefix + HostedZoneName + /auth
  `For example "id."`
- *JARSignerKeyBucket:* Name of a bucket with Jarsigner key and JARA. The bucket must be in the same account.
  `AWS web console > S3 > Buckets > my-edp-secrets`
- *JARSignerKeyPath:* Path for Jarsigner key file in a bucket
  `Example: jarsigner/jarsigner-keystore.p12`, or just `jarsigner-keystore.p12`, if this file is in the root folder.
- *JARSignerKeyPassword:* Password for Jarsigner key file
  `This password is the one that was provided while creating the jarigner key-file (see in step 4.2).`
- *KeycloakPassword:* Password for admin keycloak user
  `Provide any secure password`
- *DBMasterPassword:* Root password of PostgreSQL database in EC2 instance
  `Provide any secure password`
- *SSHAccessCIDR:* CIDR range for SSH access (Port 22). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *HTTPAccessCIDR:* CIDR range for HTTP access (Port 80). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *HTTPSAccessCIDR:* CIDR range for HTTPS access (Port 443). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *PostgresAccessCIDR:* CIDR range for PostgreSQL access (Port 5432). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *CustomAppAccessCIDR:* CIDR range for Custom App (Port 8085). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *CustomApplicationAccessCIDR:* CIDR range for Custom Application (Port 8080). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.

- *RedisAccessCIDR:* CIDR range for Redis access (Port 6379). Set to x.x.x.x/32 to allow one specific IP address, 0.0.0.0/0 to allow all IP addresses, or another CIDR range.
- *SNSNotificationEmail:* E-mail address for SNS notification. Must be confirmed after the creation of the SNS topic. This email address is sent to and stored within your AWS account. This information will be used solely to send notifications about the start, stop or restart of EDP services.
  `For example "group@anycompany.com"`

Listed below are the steps that must be followed, either after entering the parameters directly in the configuration file or after editing them in the AWS web console.

- Log into AWS
- Set account region to "us-east-1"
- Go to "Cloud formation" service
- Click the "Create stack" button
- In the "Prepare template" section, select "Choose an existing template"
- In the "Specify template" section, choose "Upload a template file"
- Click on "Choose file" and choose `cloudformation.yaml`. Click "Next" to proceed
- In the "Configure stack details" step, leave all the settings as default and click "Next"
- In the "Review and Create" step, leave all the settings as default and check all the checkboxes under the "Capabilities and transforms" section.
- Click "Next" to proceed
- It will take several minutes to prepare and initialize all the necessary resources. Wait until the status of "stack-name" is "CREATE_COMPLETE"
- Cloudformation template will also create several IAM roles which are listed below.

4.9 Created IAM roles

As mentioned before, the Cloudformation template creates several IAM roles. Those are:

- AICAS-EDP-BootstrapRole:
  This role is intended to be used by a Lambda function. When the Lambda function executes, it needs permissions to perform certain actions, like writing logs to CloudWatch. This role grants the necessary permissions for logging and allows the Lambda service to assume this role.
- AICAS-EDP-RepublishMessageRole:
  This role is essential for enabling AWS IoT to effectively manage and route messages

within an IoT environment by providing the necessary permissions to republish messages to other topics.

- AICAS-EDP-SnapshotRole:
  This role is essential for enabling AWS Lambda to perform automated management of EC2 snapshots, ensuring that backups are regularly created, properly tagged, and cleaned up when no longer needed.
- AICAS-EDPIotRole:
  This role is crucial for Lambda functions that are part of an IoT solution, providing them with the necessary permissions to interact with IoT devices, process IoT data, and retrieve supplementary data from S3.
- DeleteECRRepositoryLambdaExecutionRole:
  This role is essential for automating tasks related to ECR repository management and ensures that the Lambda function has the necessary permissions to perform its duties securely and efficiently.
- ExecutionRole:
  This role needed by ECS.Purpose is to facilitate the secure and efficient execution of ECS tasks within a pipeline. The role ensures that these tasks can interact with SQS for messaging and queue management, pull necessary resources, and log their activities while adhering to the principle of least privilege
- FargateAnomalyDetectorRole:
  This role that grants a Lambda function the necessary permissions to execute and read data from Amazon S3. This role is specifically designed for a Lambda function involved in anomaly detection, potentially within an AWS Fargate environment. The role ensures that the function can log its activities and access the data it needs from S3, which is critical for its operation.
- LambdaUpdateDockerComposeServiceRole:
  This role that equips an AWS Lambda function with the necessary permissions to manage Docker Compose services running on EC2 instances, as well as to interact with other AWS services like CloudFormation and SSM for comprehensive service management and automation.
- ParametersValidationLambdaExecutionRole:
  This role is intended to be used by a Lambda function. When the Lambda function executes, it needs permissions to perform certain actions, like writing logs to CloudWatch. This role grants the necessary permissions for logging and allows the Lambda service to assume this role.
- RolePolicies:
  The RolePolicies resource attaches an IAM policy to the FargateAnomalyDetectorRole role. This policy allows the role to publish messages to an SNS topic.

- TaskRole:
  This role is for the containers. This enable ECS tasks to fully interact with Amazon S3 and Amazon SQS as part of a pipeline or workflow step. The role adheres to AWS best practices by defining specific permissions needed for the tasks to function correctly within the EdpBundlePipelineStep.

4.10 Things Registration Policies

In case they are not already available in the AWS account, the next step will be to create the following three policies:

- ThingsRegistrationLogginPolicy
  - AWS > IAM > Policies > “Create policy” > Select “JSON” tab
  - Copy the text from Figure 3 and paste it into the policy editor

```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents",
            "logs:PutMetricFilter",
            "logs:PutRetentionPolicy",
            "logs:GetLogEvents",
            "logs:DeleteLogStream"
         ],
         "Resource": "*",
         "Effect": "Allow"
    }
]
}
```

Figure 3: ThingsRegistrationLogginPolicy

  - Click Next, name this policy as “ThingsRegistrationLogginPolicy” and press the “Create policy” button
- ThingRegistrationPolicy
  - AWS > IAM > Policies > “Create policy” > Select “JSON” tab
  - Copy the text from Figure 4 and paste it into the policy editor
  - Click Next, name this policy as “ThingRegistrationPolicy” and press the “Create policy” button
- ThingRegistrationRuleActionPolicy
  - AWS > IAM > Policies > “Create policy” > Select “JSON” tab

```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Action": [
            "iot:AddThingToThingGroup",
            "iot:AttachPrincipalPolicy",
            "iot:AttachThingPrincipal",
            "iot:CreateCertificateFromCsr",
            "iot:CreatePolicy",
            "iot:CreateThing",
            "iot:DescribeCertificate",
            "iot:DescribeThing",
            "iot:DescribeThingGroup",
            "iot:DescribeThingType",
            "iot:DetachThingPrincipal",
            "iot:GetPolicy",
            "iot:ListPolicyPrincipals",
            "iot:ListPrincipalPolicies",
            "iot:ListPrincipalThings",
            "iot:ListThingGroupsForThing",
            "iot:ListThingPrincipals",
            "iot:RegisterCertificate",
            "iot:RegisterThing",
            "iot:RemoveThingFromThingGroup",
            "iot:UpdateCertificate",
            "iot:UpdateThing",
            "iot:UpdateThingGroupsForThing"
        ],
        "Resource": "*",
        "Effect": "Allow"
    }
    ]
}
```

Figure 4: ThingsRegistrationPolicy

- Copy the text from Figure 5 and paste it into the policy editor
- Click Next, name this policy as “ThingRegistrationRuleActionPolicy” and press the “Create policy” button

Add the three policies created above into edp.jitp.role. This role is mandatory to generate the CA certificate for an organization. Please consider the following instruction to create the “edp.jitp.role” role

- AWS > IAM > Roles > Create role
- Select “AWS service” as a Trusted entity type
- Choose “IoT” from a drop-down list of the “Use case” section
- Click “Next” then “Next” again
- Provide “edp.jitp.role” as a Role name and press the “Create Role” button
- Click on the ”edp.jitp.role” to add the previously created policies

```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Action": [
            "dynamodb:PutItem",
            "kinesis:PutRecord",
            "iot:Publish",
            "s3:PutObject",
            "sns:Publish",
            "sqs:SendMessage*",
            "cloudwatch:SetAlarmState",
            "cloudwatch:PutMetricData",
            "es:ESHttpPut",
            "firehose:PutRecord"
        ],
        "Resource": "*",
        "Effect": "Allow"
    }
]
}
```

Figure 5: ThingsRegistrationRuleActionPolicy

- Select the “Permissions” tab
- Click on “Add permissions” and select “Attach policies” from the drop-down list
- Search for “ThingsRegistrationLogginPolicy” , “ThingRegistrationPolicy” and “ThingRegistrationRuleActionPolicy” and check the boxes of these permissions
- Press the “Add permissions” button.

4.11 Enable Download Feature of Bundle Artifacts

- AWS > S3 > Buckets > jedp.resources-<replace this with AWS account ID>
- Go to “Permissions” tab
- Scroll down to “CORS” section, add or edit the rules that follow
- Copy the text from Figure 6 and paste it into the policy editor
- Click on the “Save changes” button

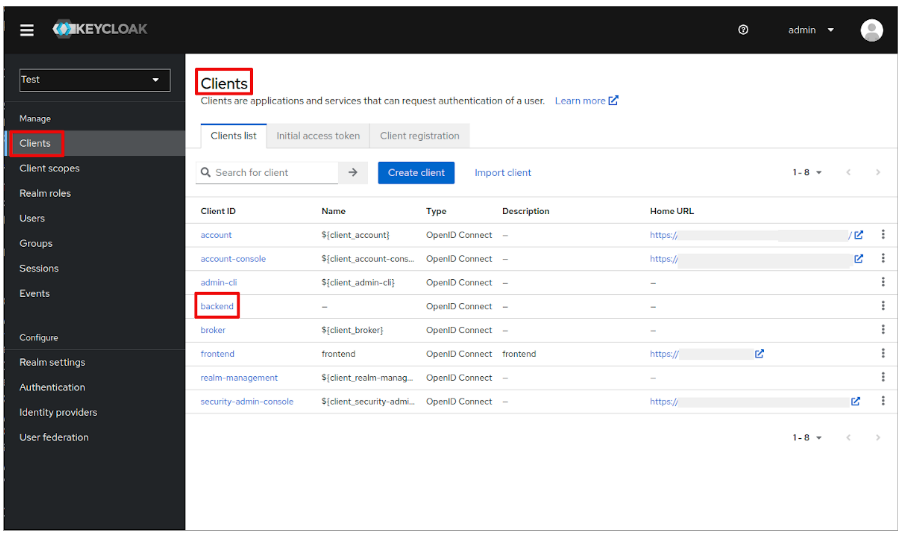
```
    [
        {
          "AllowedHeaders": [
             "Authorization"
          ],
           "AllowedMethods": [
              "GET",
               "HEAD"
           ],
            "AllowedOrigins": [
            "<the domain of the EDP instance, e.g., https://my-company.stage.edp.de for EDP Staging>"
           ],
             "ExposeHeaders": [
             "Access-Control-Allow-Origin"
           ]
        }
   ]
```

Figure 6: Enable Download Feature of Bundle Artifacts

4.12 Turn into a Superuser

Further steps in settings request the deployer to assign the role of "superuser" to the account received from the AWS SNS email. The email contains the admin credentials and URL for the EDP web portal and Keycloak. Log into Keycloak and follow the steps below.

- Open the Keycloak management console
- Select your realm from the dropdown menu
- Click on "Clients"
- Click on "backend" (Figure 7)



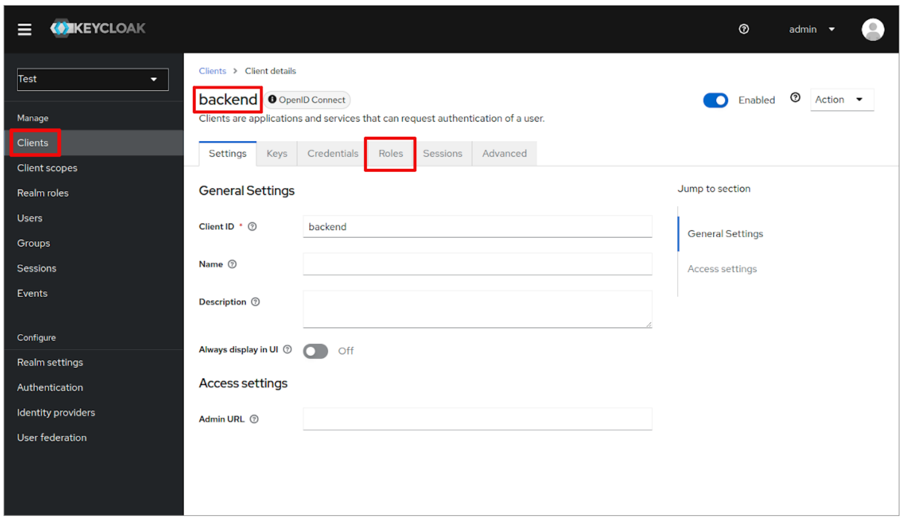
Clients

Clients are applications and services that can request authentication of a user. Learn more

Client ID	Name	Type	Description	Home URL
account	${client_account}	OpenID Connect	–	https://[redacted]/
account-console	${client_account-cons...	OpenID Connect	–	https://[redacted]
admin-cli	${client_admin-cli}	OpenID Connect	–	–
backend	–	OpenID Connect	–	–
broker	${client_broker}	OpenID Connect	–	–
frontend	frontend	OpenID Connect	frontend	https://[redacted]
realm-management	${client_realm-manag...	OpenID Connect	–	–
security-admin-console	${client_security-admi...	OpenID Connect	–	https://[redacted]

Figure 7: First step is to choose "backend" under "Clients"

- Click on the tab "Roles" choose "Create role" (Figure 8)



backend

Clients are applications and services that can request authentication of a user.

General Settings

Client ID *: backend

Name:

Description:

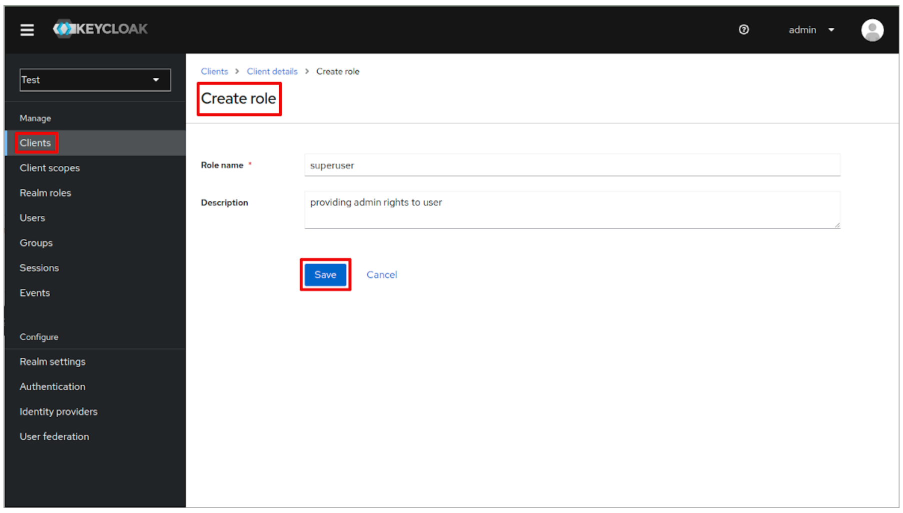
Always display in UI: Off

Access settings

Admin URL:

Figure 8: Once in “backend”, choose “Roles”

- Provide “superuser” as a “Role name” field (Figure 9)



Create role

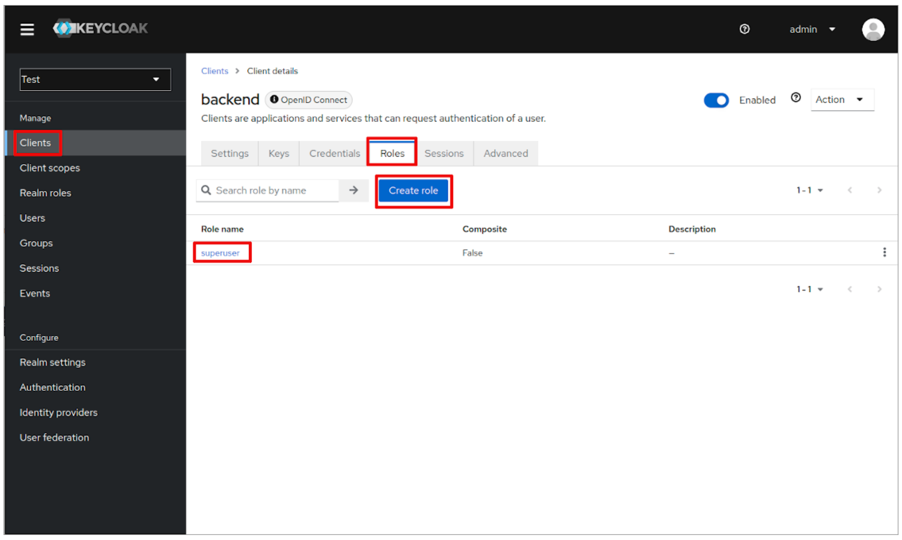
Role name *: superuser

Description: providing admin rights to user

Figure 9: Create a superuser role and click on “Save”

- Verify that the new role appears on the list (Figure 10)
- Click on “Users” and select the name to become the superuser (Figure 11)
- Click on “Role Mapping” and choose “Assign role” (Figure 12)
- Choose “superuser” (Figure 13). The “Filter by clients” function can facilitate the search.
- Click on “Assign”

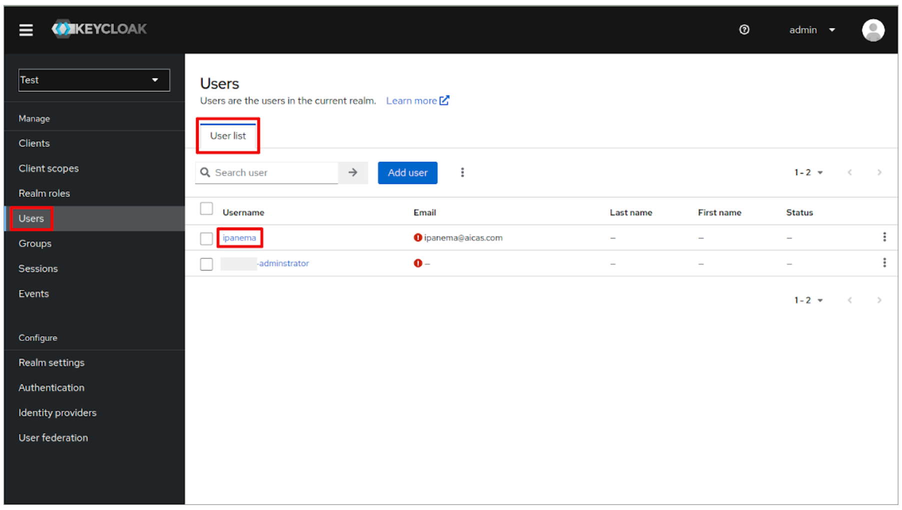
Please note that further realm configurations, like setting the EDP environments and the



Clients are applications and services that can request authentication of a user.

Role name	Composite	Description
superuser	False	–

Figure 10: Superuser is now an available role



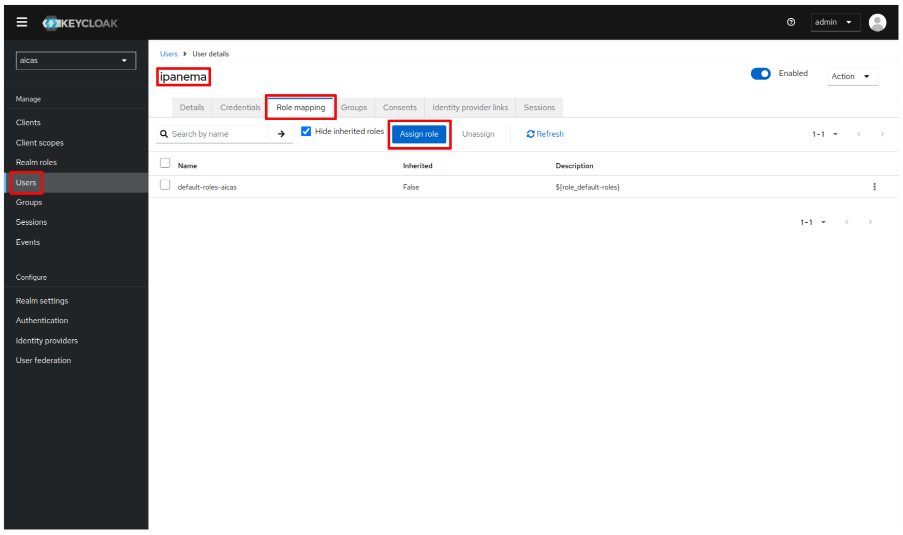
Users are the users in the current realm.

Username	Email	Last name	First name	Status
ipanema	ipanema@aicas.com	–	–	–
[redacted]-adminstrator	–	–	–	–

Figure 11: Select an user for role assignment

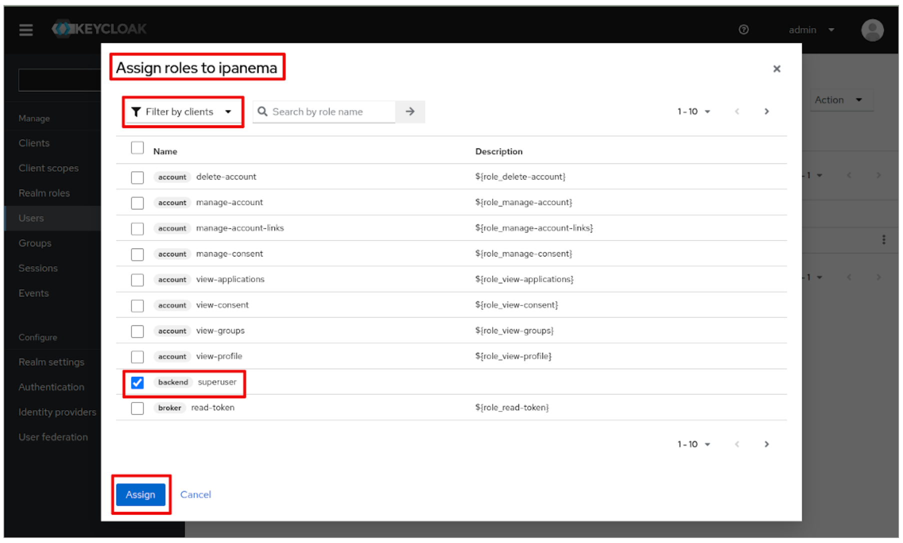
organization, as well as the steps to create EDP users and assign them EDP roles and permissions, are described in EDP's usage documentation.

Access to the user manual is given by the "Help" button, placed at the bottom of the main menu in the EDP portal. It triggers the download of its pdf file (Figure 14).



Name	Inherited	Description
default-roles-aicas	False	${role_default-roles}

Figure 12: Start to map the role to the user



	Name	Description
☐	account delete-account	${role_delete-account}
☐	account manage-account	${role_manage-account}
☐	account manage-account-links	${role_manage-account-links}
☐	account manage-consent	${role_manage-consent}
☐	account view-applications	${role_view-applications}
☐	account view-consent	${role_view-consent}
☐	account view-groups	${role_view-groups}
☐	account view-profile	${role_view-profile}
✓	backend superuser	
☐	broker read-token	${role_read-token}

Figure 13: Choose from the list the role “superuser” and assign it to the user

4.13 Setting up JARA Target

For JAR acceleration, it is necessary to set up a JARA target in the EDP web portal, under the menu “Settings”. That is only possible for users with admin rights and that have been assigned the role of “superuser”, like seen above.

- Log into the EDP web portal
- Go to “Settings”
- Click on “Jara Targets”

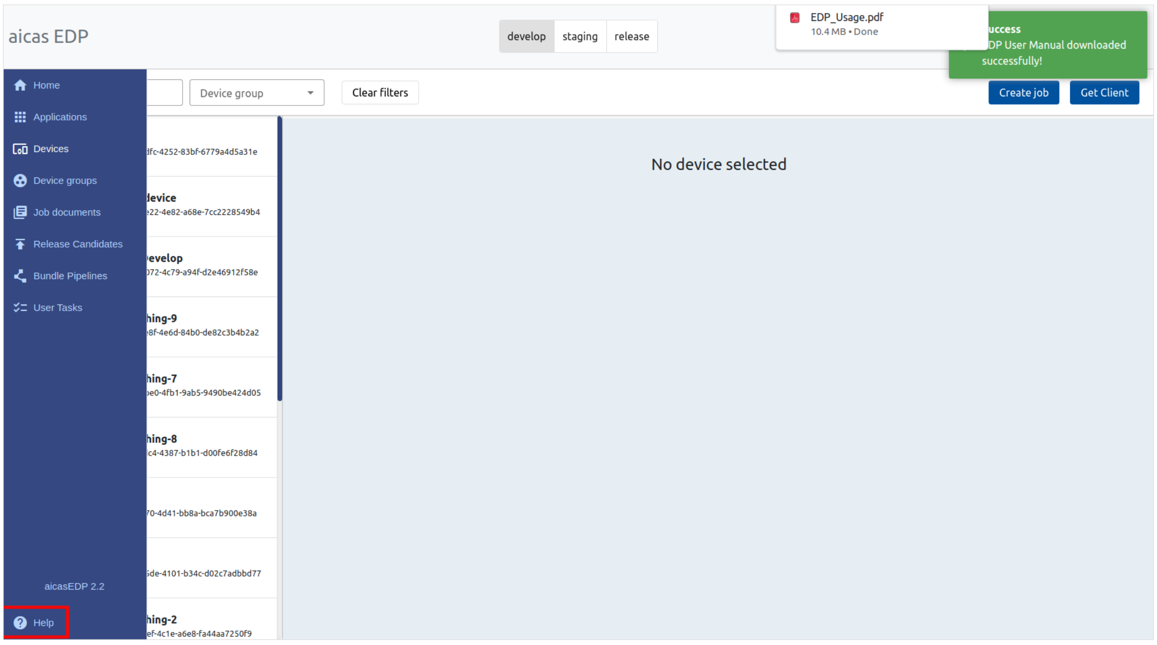


Figure 14: A Help button provides access to the user manual

- Click on "Add Fargate Task"
- Add the necessary input:
  - *Friendly Name:* Provide the friendly name of the Jara target
    `linux-x86_64`
  - *Classifier:* Provide the Jara target classifier
    `linux_x86_64`
  - *Family:* Provide the family name of the Fargate task definition
    `bp-jara-linux_x86_64_m1`
  - *Revision:* Provide the revision of the Fargate task definition
    `AWS web console > ECS > Task Definition > bp-jara-linux_x86_64_m1 > bp-jara-linux_x86_64_m1:revision`
  - *Timeout:* Provide the timeout duration in the ISO 8601 duration format. The "P" indicates the start of the duration, "T" indicates the time part, and "10M" specifies that the duration is 10 minutes.
    `PT10M`
  - *Jar sign keystore S3 bucket:* Name of a bucket with the Jara signer key. The bucket must be in the same account
    `my-edp-secrets`
  - *Jar sign keystore S3 key:* Path for the JARA key file in a bucket

`jarsigner/jarsigner-keystore.p12`

  – *Jar sign keystore password:* Password for the Jara signer key file
    `jar signer password`

Note: The usage of the JAR Accelerator requires a valid license, which is already integrated in the bundle pipeline image.

4.14 Verifying the Logs of the Docker Containers

In case, after deploying EDP, users verify that some instances are not running as expected, it is recommended to check the logs of the docker containers before contacting `support@aicas.com`.

- As a first step, start a terminal session via web console
  `AWS web console > EC2 > Instances > Instance ID > Connect > EC2 Instance Connect > Connect`
- For a status overview, list all docker containers with:
  ```
  docker ps -a
  ```
- To show general or specific log outputs from the containers, enter:
  ```
  docker-compose logs
  docker-compose logs keycloak
  docker compose logs backend
  docker compose logs frontend
  ```
- The command to check the Postgres status is:
  ```
  systemctl status postgresql.service
  ```

4.15 Management of EDP Related Secrets

Following are the keys and password which need to be secured.

- DB Master Password
- Keycloak Master Password
- JAR Signer Key Password

These passwords can be stored in any of the available Password Managers on the market. Always choose strong passwords and avoid sharing them.

It is ususally recommended as good practice to change passwords of the used services, and also of user accounts, periodically. However, there is no absolute agreement on this: The North American institute NIST (National Institute of Standards and Technology), for example, advises against doing so, recommending instead to just force changes in case there is evidence that the password has been compromised.

4.16 Instance Metadata Service

Instance Metadata Service (IMDS) is a component in the EC2 instance, used for accessing metadata that relates to the EC2 instance. Consider that IMDS is the place where the temporary credentials for the instance's role are stored. Many libraries and tools automatically reach out to IMDS to retrieve those credentials and then access AWS services in accordance with the permissions assigned by the instance role. If access to this metadata was blocked or unavailable, then all of those things would fail.

Several known attack types can access the metadata server from a remote location and access the AWS account by utilizing an EC2 instance of the account. EDP is currently using version 2.17.100 of AWS SDK which by default enables IMDSv2. In IMDSv2 a token is required when requesting the metadata. Version 2 is also session-based as the token will expire after a period of time.

IMDSv2 is the recommended security best practice to enable on your instances. It provides another layer of security to access the instance metadata.

In order to get more details about this service by AWS, please refer to https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/configuring-instance-metadata-service.html

- Execute
  `aws ec2 describe-instances -instance-id <instance-id> -region us-east-1`
- If "HttpTokens" is equal to required, then it means IMDSv2 is enabled. This can be checked by executing the following command:

  ```
  curl -w "\%{http_code}\n" http://169.254.169.254/
  ```

  A 401 status code after that would mean unauthorized. Note that this would not occur if the instance metadata service were in its version 1.
- To get instance metadata using version 2, execute the following command. You will be requesting a token and storing it into a variable.

  ```
  TOKEN=’curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
  metadata-token-ttl-seconds: 21600"’
  curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-
  data/
  ```

- You will be able to get the response to the requests with a provided token.

5 Routine Maintenance

This section will provide information about the following topics:

- Rotation of programmatic credentials
- Software patches and upgrades
- AWS service limits

5.1 Programmatic Credentials Rotation

IAM access key rotation is a healthy security practice as it ensures that any keys that may have been leaked, either due to reuse or breach, become irrelevant. aicas recommends following the AWS best practices for AWS IAM user account access key rotation. Please do not delete the old access keys without reading the following official documents.

A guide for automatic rotation of IAM user acess key can be found at:
https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/automatically-rotate-iam-user-access-keys-at-scale-with-aws-organizations-and-aws-secrets-manager.html

A guide for manual rotation of IAM user access key can be found at:
https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html#Using_RotateAccessKey

5.2 Software Patches and Upgrades

aicas will contact EDP's customers to inform them about updates and new patches. For specific details about versions of EDP, please contact support@aicas.com.

5.3 AWS Service Limits

Every AWS service currently used for EDP runs within their default values, and service limits do not need to be altered during deployment.

6 Operational Guidance

This section will provide instructions on the following topics:

- Health check
- Emergency maintenance
- Backup and software recovery

6.1 Health Check

Product functionality can be checked by monitoring the status of docker containers. If any of the containers is not running, EDP will not run.

- To verify that, start a terminal session via web console
  `AWS web console > EC2 > Instances > Instance ID > Connect > EC2 Instance Connect > Connect`
- List all four docker containers (keycloak, frontend, backend, redis).
  ```
  docker ps -a
  ```
- If all four docker containers status are "Up", the application's state is "healthy"; otherwise actions need to be taken according to errors. Logs can be retrieved via the following commands:
  ```
  docker-compose logs
  docker-compose logs keycloak
  docker-compose logs frontend
  docker-compose logs backend
  docker-compose logs redis
  ```
- Command to check Postgres status
  ```
  systemctl status postgresql.service
  ```

6.2 Emergency Maintenance

Emergency maintenance provides a guidance for actions to be taken in the event of fault conditions, such as a transient or even permanent, failure of an AWS service.

6.2.1 Handling Fault Conditions

- In case of faulted EC2 instances, please refer to the official AWS documentation https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-troubleshoot.html
- In case an EC2 scheduled maintenance event has been received, please refer to https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-instances-status-check_sched.html
- If availability of EC2 in a particular availability zone is degraded, please refer to https://docs.aws.amazon.com/sap/latest/general/arch-guide-failure-scenarios.html
- In case of failure of EC2 instances with system check error, please refer to https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-system-instance-status-check.html

6.3 Backup and Software Recovery

Lambda creates an EC2 snapshot which is basically used as a backup for the EC2 instance with the database. The EC2 instance can be restored from the snapshot. The first step would be to create an AMI image from one of the snapshots and then launch an EC2 instance from this

AMI. More details on this type of backup strategy can be found at https://docs.aws.amazon.com/prescriptive-guidance/latest/backup-recovery/ec2-backup.html

- Snapshots can be found at
  `AWS web console > EC2 > Elastic Block Store > Snapshots`
- Creation of AMI image
  `AWS web console > EC2 > Elastic Block Store > Snapshots > Snapshot ID > Actions > Create image from snapshot`
- AMI image can be found at
  `AWS web console > EC2 > AMIs`
- Launch instance from AMI
  `AWS web console > EC2 > AMIs > AMI ID > Launch instance from AMI`

7 Support

For general questions about support and the different possible levels of service provided by aicas, please contact `support@aicas.com`.

7.1 Troubleshooting

- How to log into my AWS account?
  - Please have a look into https://docs.aws.amazon.com/signin/latest/userguide/how-to-sign-in.html#user-types-defined
- I am not able to configure AWS CLI.
  - Please have a look into https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html
- How to find my Account ID?
  - Please have a look into https://docs.aws.amazon.com/IAM/latest/UserGuide/console_account-alias.html
- I cannot see the deployed resources.
  - Please make sure that you have selected the correct region on your AWS account. For details please have a look into https://docs.aws.amazon.com/awsconsolehelpdocs/latest/gsg/select-region.html
- How to verify if an EDP instance is in running state?
  - Please follow the instructions provided in the "Health Check" section
- How to restart an EDP instance?

- Log into the AWS account
- Go to Services > EC2 Instance > Instance ID > Connect > EC2 Instance Connect > Connect
- Execute `sudo bash /bin/aicas-edp/aicas-edp.sh`
- Click "Action" button
- Click "Restart app server(s)"

- How to restart docker containers?
  - Log into the AWS account
  - Go to Services > EC2 Instance > Instance ID > Connect > EC2 Instance Connect > Connect
  - Execute `sudo bash /bin/aicas-edp/aicas-edp.sh`
    This command will first remove the docker containers and then start
- How to access an EC2 instance by using AWS CLI.
  - Please have a look into https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-connect-methods.html