CrateDB

Real-time Demo Workshop

Table of Contents

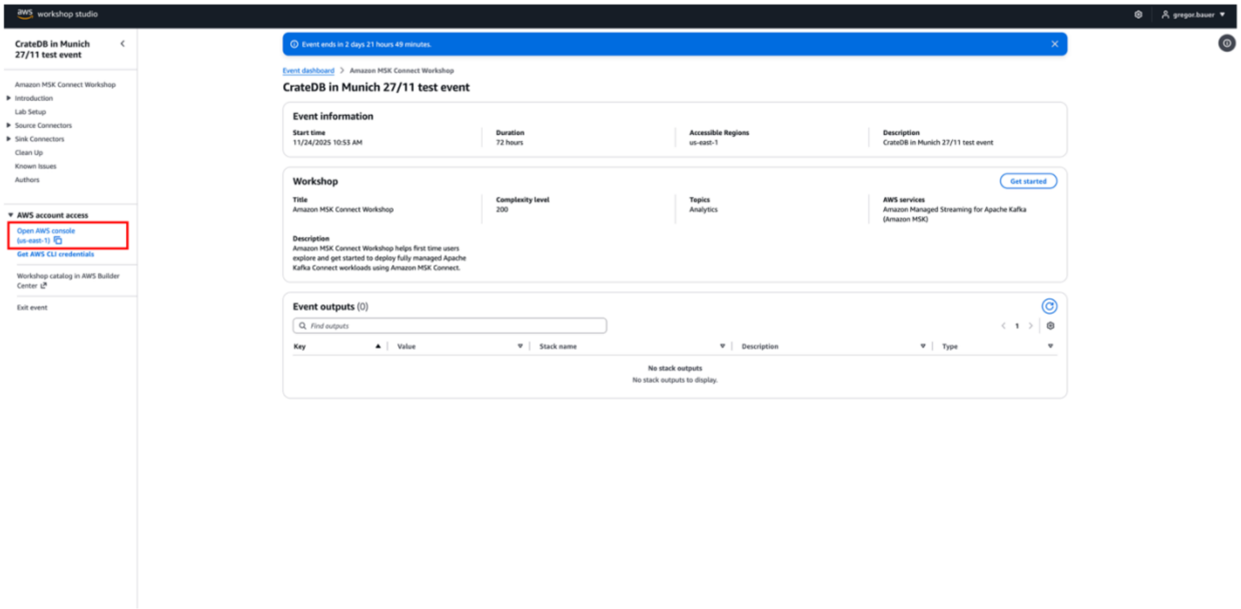
Workshop User Instructions

Creating AWS Workshop Pre-Requisites

Note: Please use the exact names provided in this guide for naming instances otherwise build steps might fail. Use copy paste to avoid errors.

Note: If you already have AWS console it's best to use another browser

1. Click on the provided link and add the email you registered for the event to get an OTP
2. After entering the OTP, you received go to the Amazon Console



Event ends in 2 days 21 hours 49 minutes.

CrateDB in Munich 27/11 test event

Event information

Start time: 11/24/2025 10:53 AM
Duration: 72 hours
Accessible Regions: us-east-1
Description: CrateDB in Munich 27/11 test event

Workshop

Title: Amazon MSK Connect Workshop
Complexity level: 200
Topics: Analytics
AWS services: Amazon Managed Streaming for Apache Kafka (Amazon MSK)

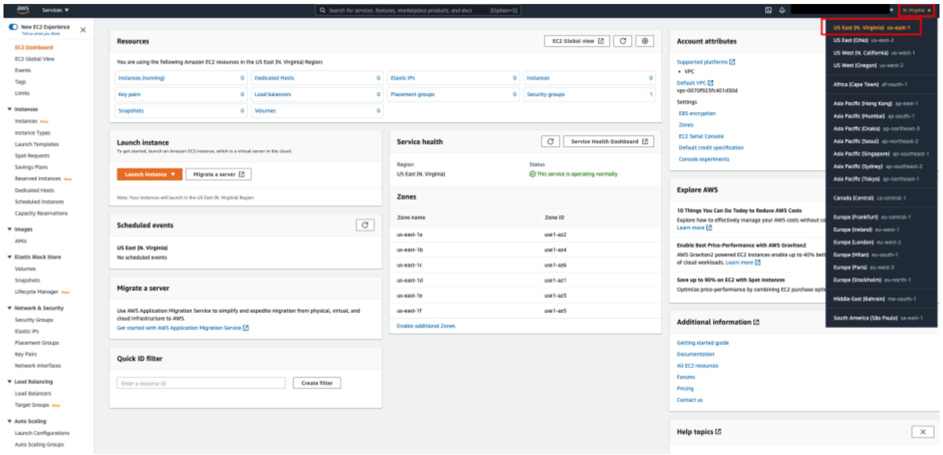
Description: Amazon MSK Connect Workshop helps first time users explore and get started to deploy fully managed Apache Kafka Connect workloads using Amazon MSK Connect.

Event outputs (0)

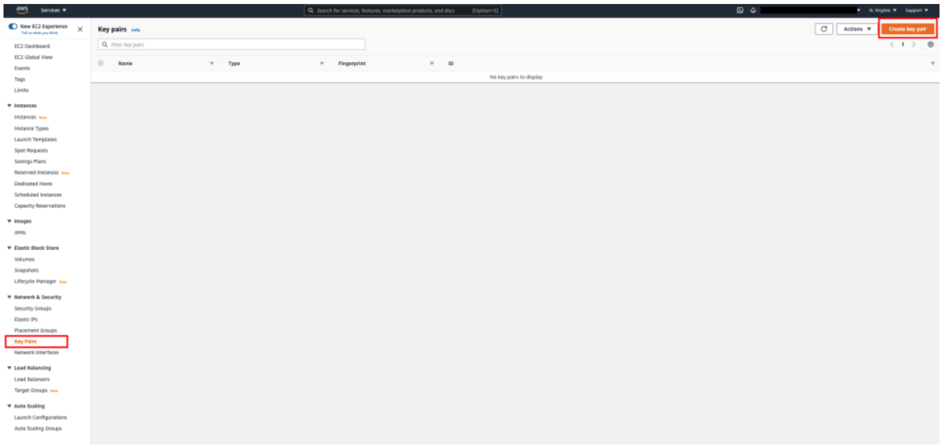
No stack outputs
No stack outputs to display.

3. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/

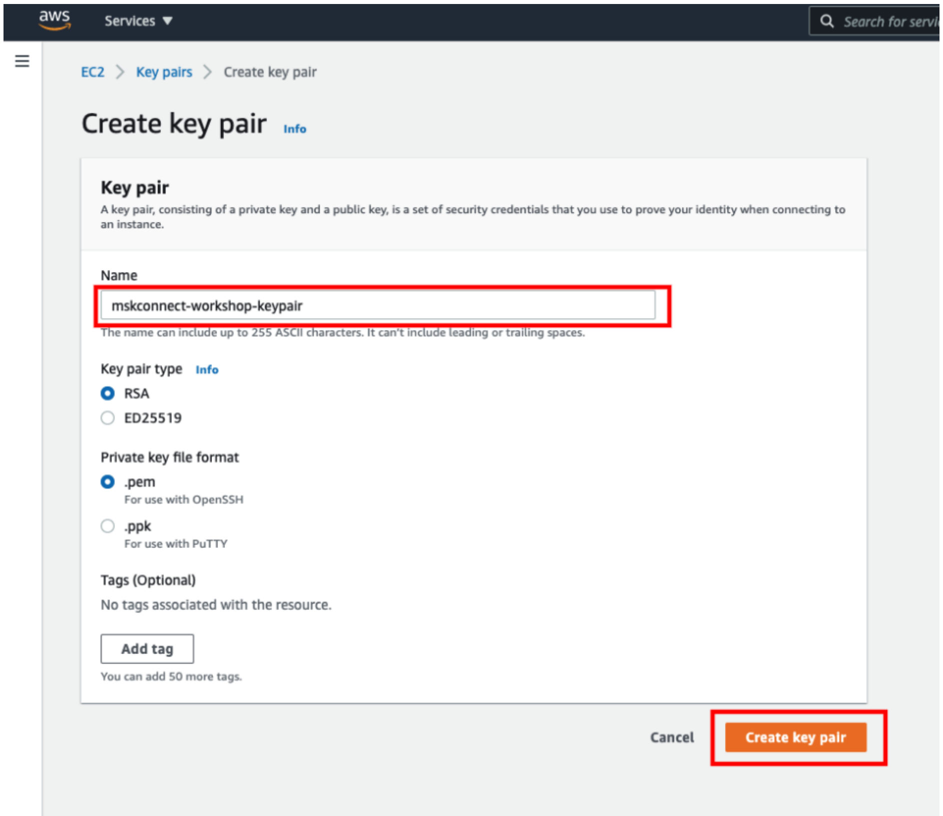
Choose region **us-east-1** this will be the region that you are using for the workshop.


Services
New EC2 Experience
EC2 Dashboard
EC2 Global View
Events
Tags
Limits
Instances
Images
Elastic Block Store
Network & Security
Load Balancing
Auto Scaling
Resources
EC2 Global view
Launch instance
Migrate a server
Service health
Service Health Dashboard
Zones
Scheduled events
Quick ID filter
Create filter
Account attributes
Explore AWS
Additional information
Help topics
US East (N. Virginia) us-east-1
US East (Ohio) us-east-2
US West (N. California) us-west-1
US West (Oregon) us-west-2
Africa (Cape Town) af-south-1
Asia Pacific (Hong Kong) ap-east-1
Asia Pacific (Mumbai) ap-south-1
Asia Pacific (Osaka) ap-northeast-3
Asia Pacific (Seoul) ap-northeast-2
Asia Pacific (Singapore) ap-southeast-1
Asia Pacific (Sydney) ap-southeast-2
Asia Pacific (Tokyo) ap-northeast-1
Canada (Central) ca-central-1
Europe (Frankfurt) eu-central-1
Europe (Ireland) eu-west-1
Europe (London) eu-west-2
Europe (Milan) eu-south-1
Europe (Paris) eu-west-3
Europe (Stockholm) eu-north-1
Middle East (Bahrain) me-south-1
South America (São Paulo) sa-east-1

2. Choose **Key Pairs** in the navigation bar on the left and click on **Create key pair**.



3. Provide a name for the key pair (e.g. **mskconnect-workshop-keypair**) and click on **Create key pair**.



Create key pair Info

Key pair

A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name

mskconnect-workshop-keypair

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type Info

● RSA

○ ED25519

Private key file format

● .pem
For use with OpenSSH

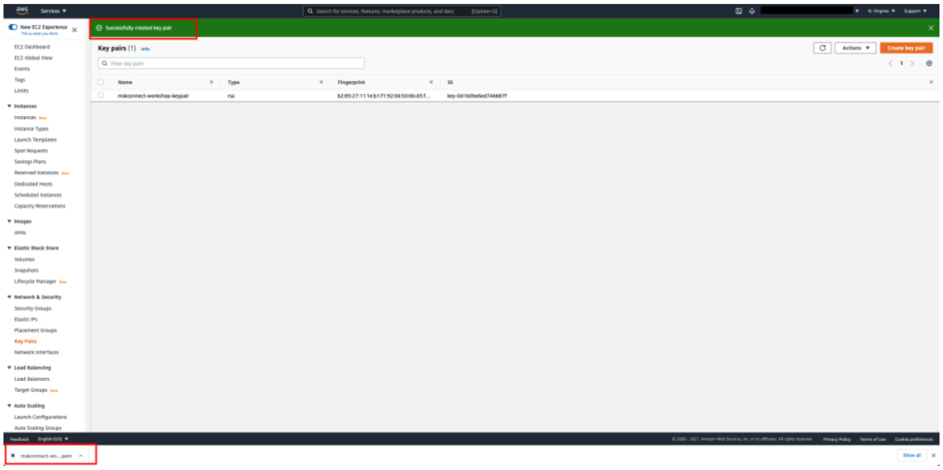
○ .ppk
For use with PuTTY

Tags (Optional)

No tags associated with the resource.

You can add 50 more tags.

4. You should see the message **Successfully created key pair**, and confirm the download of the generated **.pem** file to your local machine.



Successfully created key pair

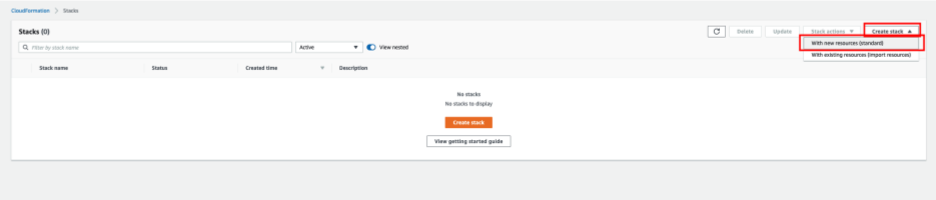
Key pairs (1)

Name	Type	Fingerprint	ID
mskconnect-workshop-keypair	rsa	b2:85:27:11:1e:b1:f1:92:0d:50:6b:d3:f...	key-0d1b0be6ed746687f

mskconnect-wo....pem

[Deploy required AWS resources via CloudFormation]

5. Download the CloudFormation template here: msk-connect-workshop-3.6.0.yml
6. Open the AWS CloudFormation console at https://console.aws.amazon.com/cloudformation/
7. Click on **Create stack** and select **With new resources (standard)**.



CloudFormation > Stacks

Stacks (0)

Filter by stack name | Active ▾ | View nested

Delete | Update | Stack actions ▾ | Create stack ▴

With new resources (standard)

With existing resources (import resources)

Stack name	Status	Created time	Description

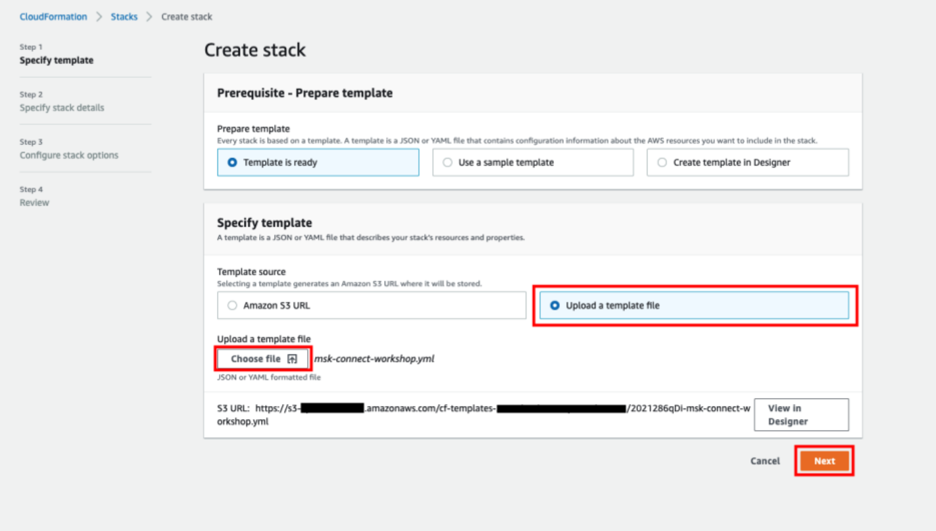
No stacks

No stacks to display

Create stack

View getting started guide

8. Choose **Upload a template file** option and upload the CloudFormation template from your local machine, then click **Next**.



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 ● Upload a template file

Upload a template file

Choose file *msk-connect-workshop.yml*

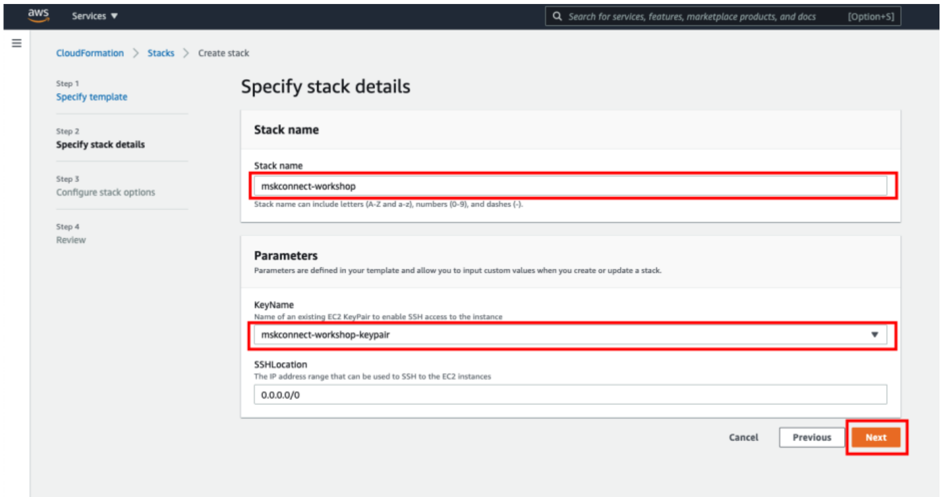
JSON or YAML formatted file

S3 URL: https://s3-[redacted].amazonaws.com/cf-templates-[redacted]/2021286qDi-msk-connect-workshop.yml

View in Designer

Cancel | Next

9. Enter **mskconnect-workshop** as the stack name, choose the **mskconnect-workshop-keypair** under **KeyName**, and then click **Next**.



aws Services ▾ Search for services, features, marketplace products, and docs [Option+S]

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: mskconnect-workshop

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.

KeyName

Name of an existing EC2 KeyPair to enable SSH access to the instance

mskconnect-workshop-keypair ▾

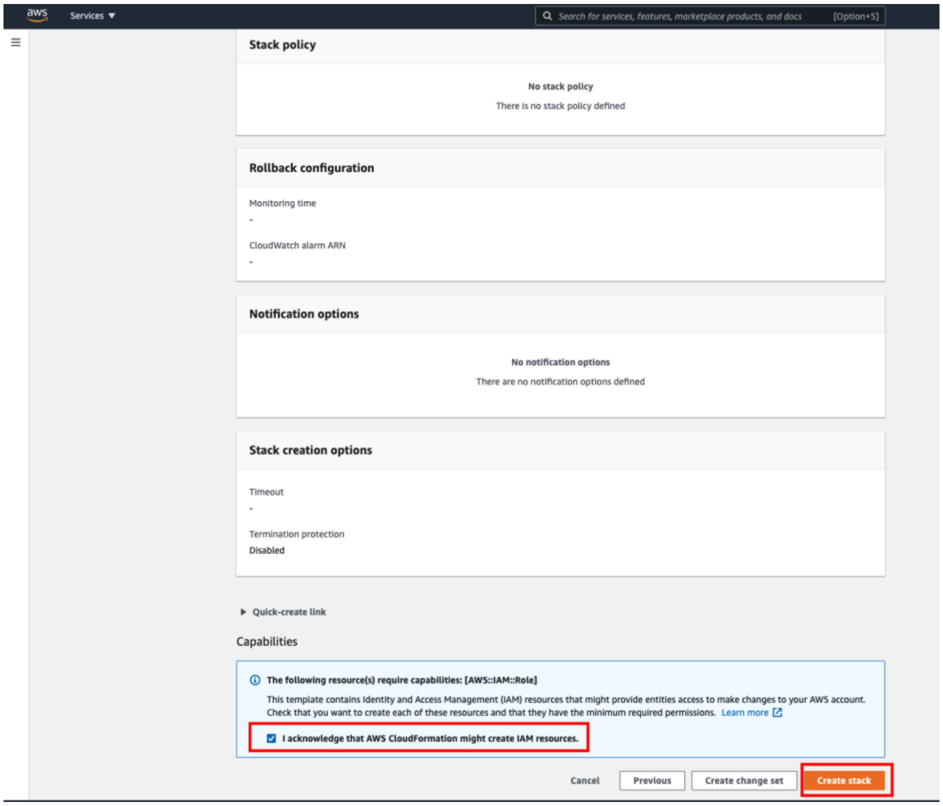
SSHLocation

The IP address range that can be used to SSH to the EC2 instances

0.0.0.0/0

Cancel | Previous | Next

10. Leave the default settings on the step **Configure stack options** and click **Next**.
11. Scroll down to the bottom of the **Review mskconnect-workshop** screen and check the box to acknowledge the creation of IAM resources. Click **Create stack** to trigger the creation of the CloudFormation stack.



Stack policy

No stack policy

There is no stack policy defined

Rollback configuration

Monitoring time
-

CloudWatch alarm ARN
-

Notification options

No notification options

There are no notification options defined

Stack creation options

Timeout
-

Termination protection
Disabled

▸ Quick-create link

Capabilities

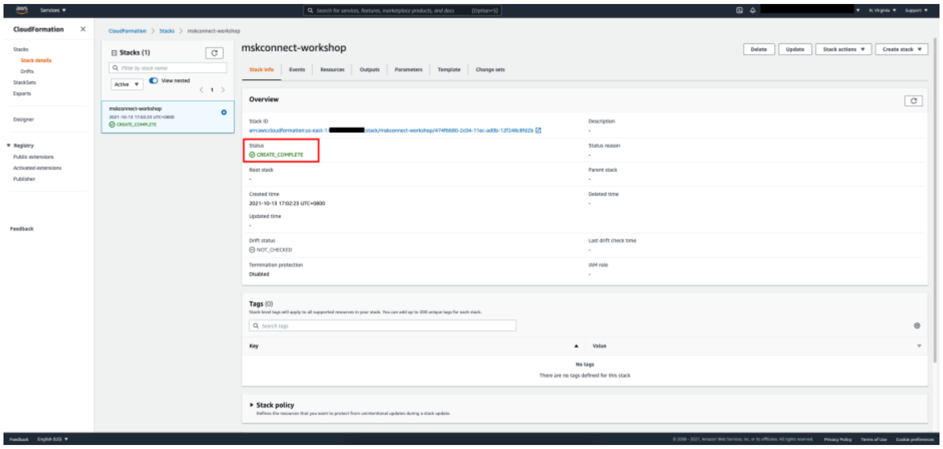
ⓘ **The following resource(s) require capabilities: [AWS::IAM::Role]**

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.**

aws
Services ▾
Search for services, features, marketplace products, and docs [Option+S]
Cancel
Previous
Create change set
Create stack

12. Wait until the stack creation is completed before proceeding to the next steps. It may take a while for the MSK cluster creation (~30 to 45 mins).



mskconnect-workshop

Overview

Stack ID
arn:aws:cloudformation:us-east-1:[redacted]:stack/mskconnect-workshop/474f6680-2c04-11ec-ad0b-12f248c8fd2b

Description
-

Status
CREATE_COMPLETE

Status reason
-

Root stack
-

Parent stack
-

Created time
2021-10-13 17:02:23 UTC+0800

Deleted time
-

Updated time
-

Drift status
NOT_CHECKED

Last drift check time
-

Termination protection
Disabled

IAM role
-

Tags (0)
Stack-level tags will apply to all supported resources in your stack. You can add up to 200 unique tags for each stack.

No tags
There are no tags defined for this stack

▸ Stack policy
Defines the resources that you want to protect from unintentional updates during a stack update.

What does the above CloudFormation template do?

The CloudFormation stack creates the following resources:

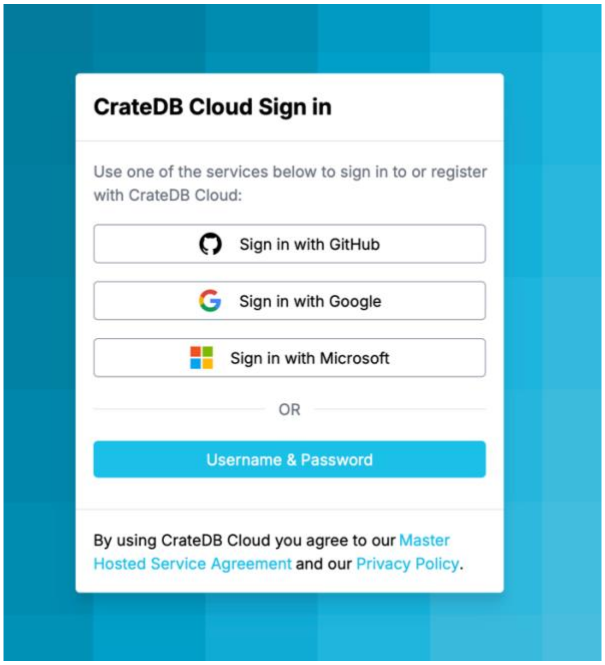
- 2 CloudWatch Log Groups for MSK Cluster and MSK Connect connectors respectively, each with 7 days log retention period
- 1 S3 bucket for storing required Apache Kafka Connect plugin files, and other testing results in some workshop modules
- 1 VPC with 1 Public subnet and 3 Private subnets, and the required networking components such as Route Tables, Internet Gateway, NAT Gateway & Elastic IP, VPC Gateway Endpoint for S3
- 1 m5.large EC2 instance with Java 1.8.0 and Apache Kafka version 3.6.0 installed, and the required Security Group, IAM Role, EC2 Instance Profile
  Note: we will not be using this and instead creating a new instance.
- 1 MSK Cluster with 3 kafka.m5.large broker nodes configured with Apache Kafka version 3.6.0, and the required Security Group

Note that the encryption options are disabled in order to reduce the chance of errors in the workshop, this is not a best practice, and you should not use the provided CloudFormation template in any production environments.

Note: While this is running, this would be a good time to create the CrateDB cluster in the next section.

Creating a CrateDB Cluster

1. Go to https://cratedb.com
2. Click on the **Start Free** button
3. You have several options to create an account, use the one that is appropriate for you. You won't need to enter any payment details.

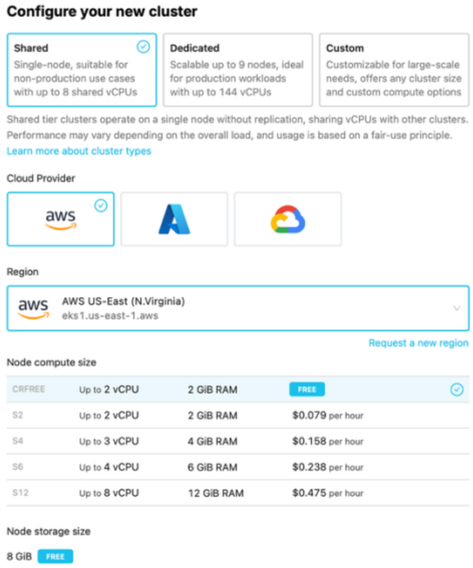


CrateDB Cloud Sign in

Use one of the services below to sign in to or register with CrateDB Cloud:

By using CrateDB Cloud you agree to our Master Hosted Service Agreement and our Privacy Policy.

4. Once you have an account, log in and deploy a **CRFREE** cluster



Configure your new cluster

☑ **Shared** Single-node, suitable for non-production use cases with up to 8 shared vCPUs

Dedicated Scalable up to 9 nodes, ideal for production workloads with up to 144 vCPUs

Custom Customizable for large-scale needs, offers any cluster size and custom compute options

Shared tier clusters operate on a single node without replication, sharing vCPUs with other clusters. Performance may vary depending on the overall load, and usage is based on a fair-use principle.

Cloud Provider: ☑ aws

Region: AWS US-East (N.Virginia) eks1.us-east-1.aws

Node compute size

CRFREE	Up to 2 vCPU	2 GiB RAM	FREE	☑
S2	Up to 2 vCPU	2 GiB RAM	$0.079 per hour	
S4	Up to 3 vCPU	4 GiB RAM	$0.158 per hour	
S6	Up to 4 vCPU	6 GiB RAM	$0.238 per hour	
S12	Up to 8 vCPU	12 GiB RAM	$0.475 per hour	

Node storage size

8 GiB FREE

- Select a **Shared** cluster (this is the only way to get a free cluster)
- Select **AWS** as the cloud provider

    - Select **AWS US-East** as the region (the same as the workshop runs within)
    - Ensure **CRFREE** is selected as compute size
    - Click **Deploy Cluster** to create it, which will take a short amount of time

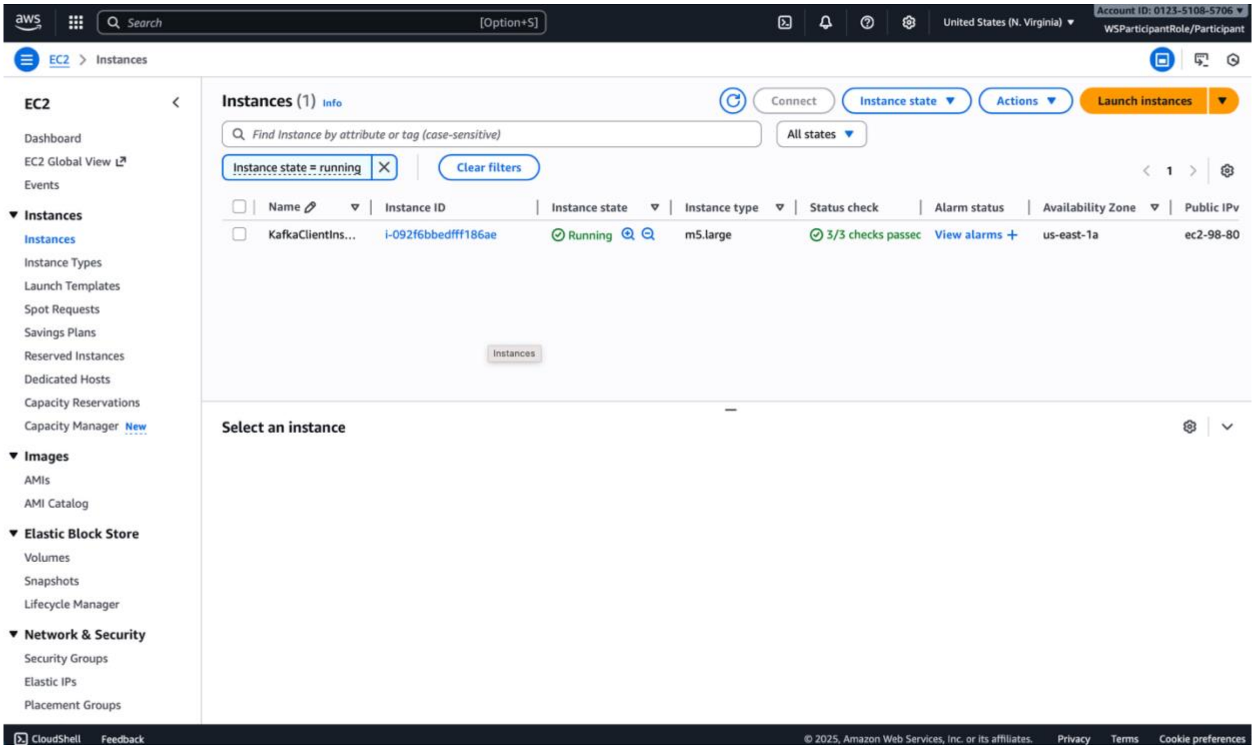
5. It's **very** important that you save the login credentials, so download or make a note now before moving on.
6. We need to create the destination table for the climate data, run the following in the Console:

```
CREATE TABLE IF NOT EXISTS "demo"."climate_data" (
   "timestamp" TIMESTAMP WITHOUT TIME ZONE,
   "geo_location" GEO_POINT,
   "data" OBJECT(DYNAMIC) AS (
      "temperature" DOUBLE PRECISION,
      "u10" DOUBLE PRECISION,
      "v10" DOUBLE PRECISION,
      "pressure" DOUBLE PRECISION,
      "latitude" DOUBLE PRECISION,
      "longitude" DOUBLE PRECISION
   )
);
```

7. The host information you will need later, so make a note or leave the webpage open for now.

Creating AWS Workshop Environment

Open the EC2 dashboard https://console.aws.amazon.com/ec2/ go to **Instances**



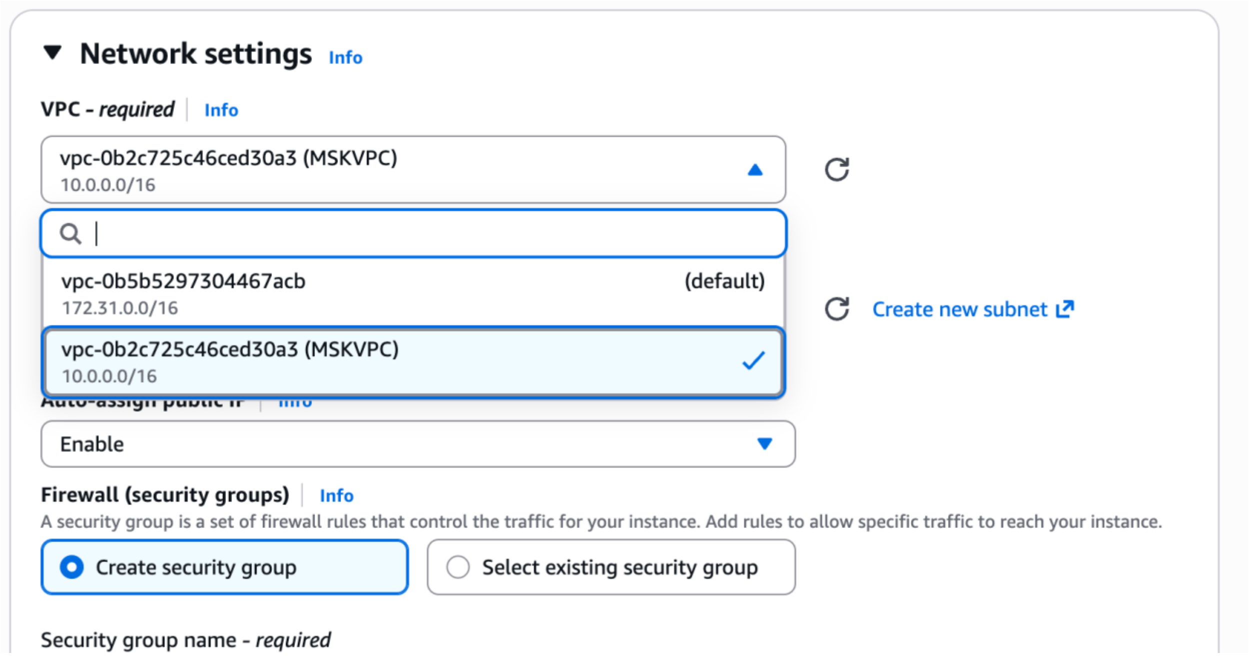
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv
KafkaClientIns...	i-092f6bbedfff186ae	Running	m5.large	3/3 checks passec	View alarms +	us-east-1a	ec2-98-80

"

Note: We need to create a new EC2 instance, do not use the one created as part of the CloudFormation process as it's ARM based, and you will need an x86 instance.

Click the **Launch instances** button and choose the following:

- Name: **cratedb-workshop**
- AMI: **Amazon Linux**
- Architecture: **x86**
- Instance type: **m5a.xlarge**
- Key-pair name: **Choose the one previously created**
- Network Settings: Click on **Edit**
  - VPC: Find and select the workshop **MSKVPC** (same as existing EC2)



▼ Network settings Info

VPC - *required* | Info

vpc-0b2c725c46ced30a3 (MSKVPC)
10.0.0.0/16

vpc-0b5b5297304467acb (default)
172.31.0.0/16

vpc-0b2c725c46ced30a3 (MSKVPC) ✓
10.0.0.0/16

Create new subnet

Auto-assign public IP | Info

Enable

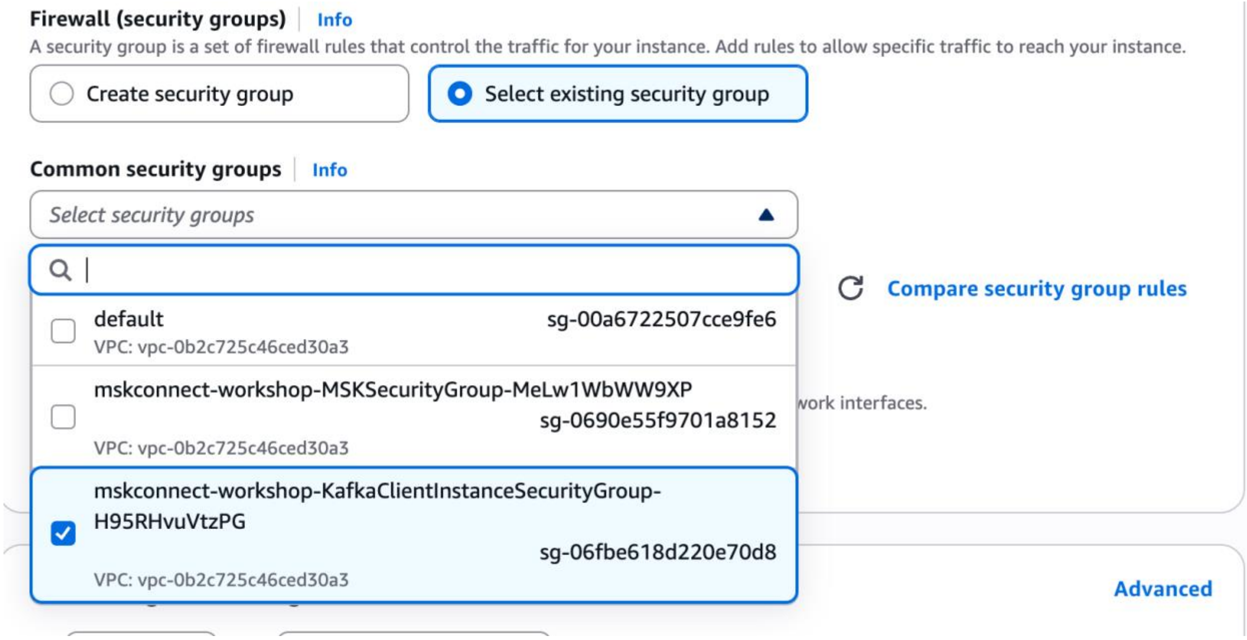
Firewall (security groups) | Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

● Create security group ○ Select existing security group

Security group name - *required*

- ○ Security group: Existing one starting mskconnect-workshop-KafkaClientInstanceSecurityGroup (same as existing EC2)



Firewall (security groups) | Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

○ Create security group ● Select existing security group

Common security groups | Info

Select security groups

Compare security group rules

□ default sg-00a6722507cce9fe6
VPC: vpc-0b2c725c46ced30a3

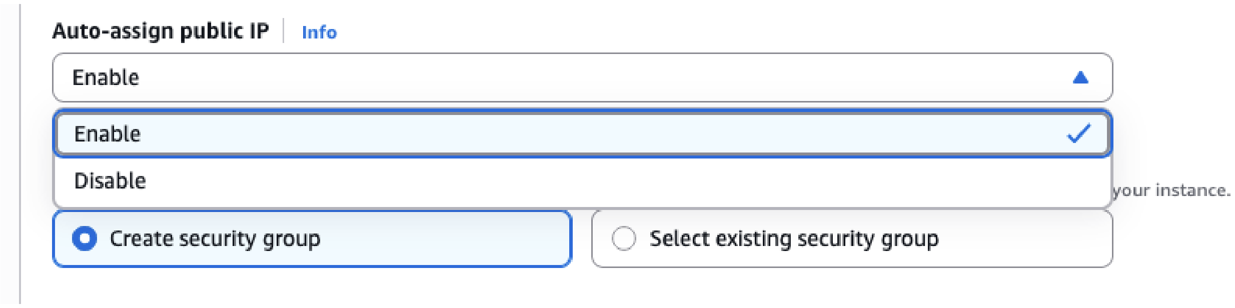
□ mskconnect-workshop-MSKSecurityGroup-MeLw1WbWW9XP sg-0690e55f9701a8152
VPC: vpc-0b2c725c46ced30a3

☑ mskconnect-workshop-KafkaClientInstanceSecurityGroup-H95RHvuVtzPG sg-06fbe618d220e70d8
VPC: vpc-0b2c725c46ced30a3

work interfaces.

Advanced

- Configure storage: **30GB**
- Ensure it's created in a **public** subnet, not private
- Ensure a public elastic IP is assigned by setting the **Auto-assign public IP** to **Enable**



Auto-assign public IP | Info

Enable

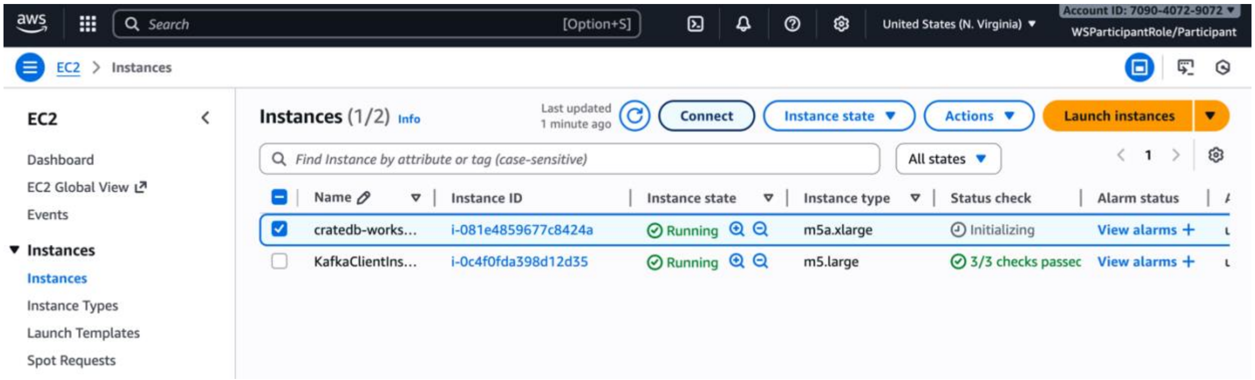
- Enable ✓
- Disable

your instance.

● Create security group ○ Select existing security group

Create the instance and wait for it to be fully ready before attempting to connect

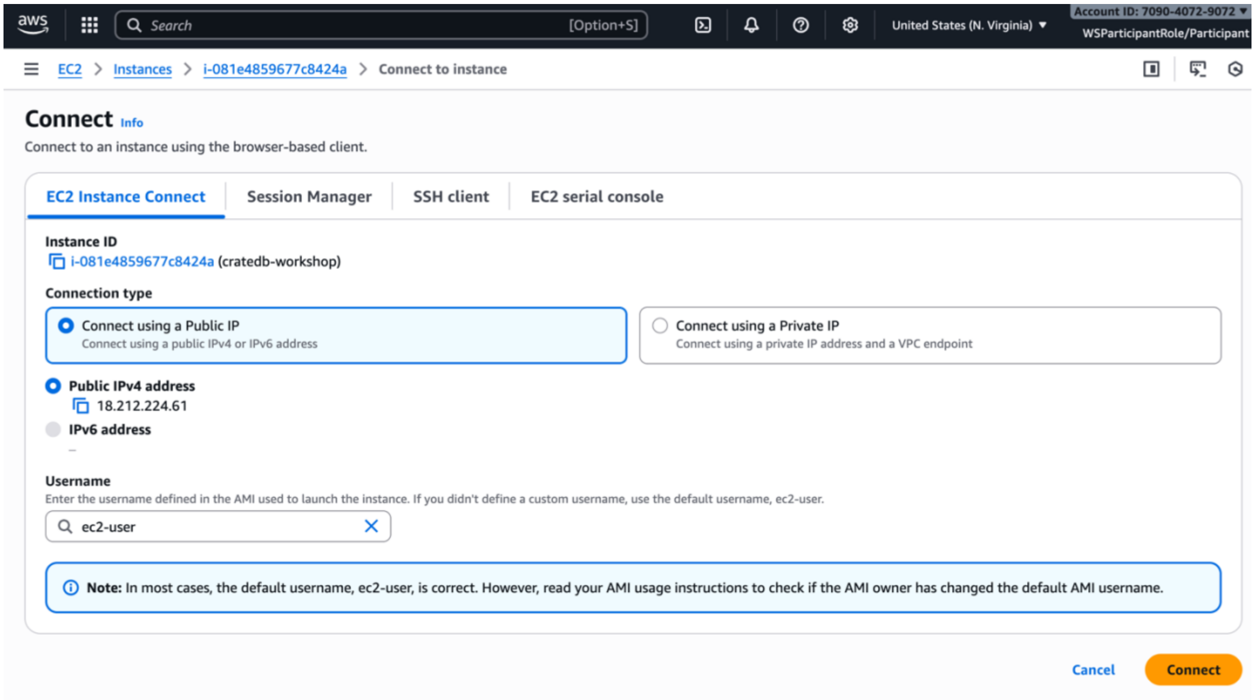
- Go to the EC2 instance and select the create one
- Click on the **Connect** button and then **Connect** again



Instances (1/2) Info

Find Instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
☑	cratedb-works...	i-081e4859677c8424a	Running	m5a.xlarge	Initializing	View alarms +
□	KafkaClientIns...	i-0c4f0fda398d12d35	Running	m5.large	3/3 checks passec	View alarms +



aws
Search
[Option+S]
United States (N. Virginia) ▾
Account ID: 7090-4072-9072 ▾
WSParticipantRole/Participant
EC2 > Instances > i-081e4859677c8424a > Connect to instance

Connect Info

Connect to an instance using the browser-based client.

EC2 Instance Connect
Session Manager
SSH client
EC2 serial console

Instance ID

i-081e4859677c8424a (cratedb-workshop)

Connection type

● Connect using a Public IP
Connect using a public IPv4 or IPv6 address

○ Connect using a Private IP
Connect using a private IP address and a VPC endpoint

● **Public IPv4 address**
18.212.224.61

○ **IPv6 address**
–

Username

Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ec2-user.

ec2-user

Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Installing the demo source code

1. Ensure the git command-line tool is installed:

   ```
   sudo dnf install git
   ```

2. Clone the repository in the EC2 console (using HTTPS, not SSH). We use a particular "unauthenticated" branch (this removes the SASL authentication code, which is not needed).

   ```
   git clone https://github.com/crate/realtime-demo.git -b unauthenticated
   ```

3. Change into the directory:

   ```
   cd realtime-demo
   ```

Running the data producer

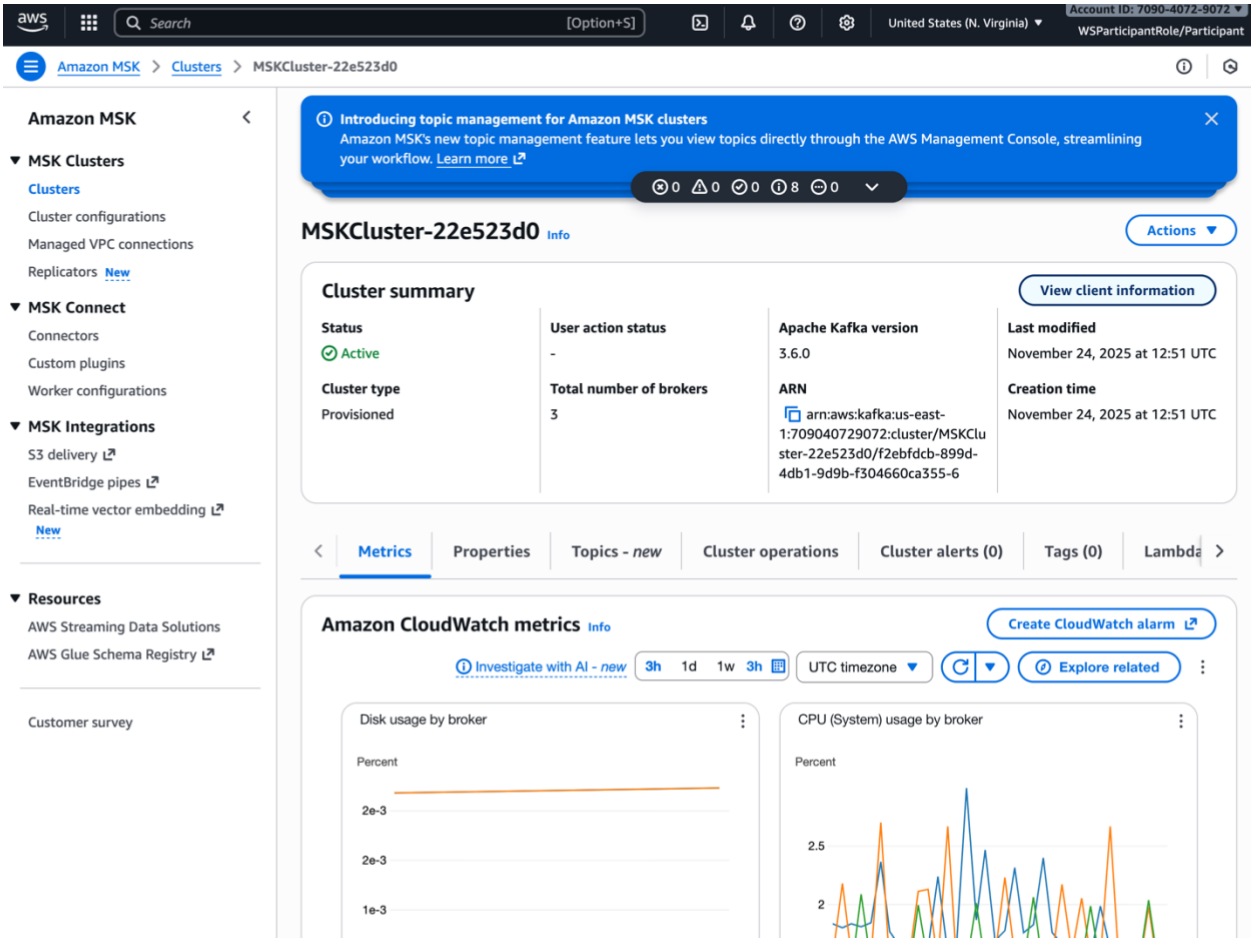
To run the producer, we need to set up a virtual Python environment and install dependencies:

```
cd data
python3 -m venv .venv
source .venv/bin/activate
pip3 install -U -r requirements.txt
```

Next, copy the example .env file

```
cp .env.example .env
```

Afterwards change the bootstrap server. Go to **MSK**, click on the MSK Cluster, then click on "View client information". Then copy one of the Bootstrap servers. Note that they are comma separated. Only select one and make sure it's fully copied.



Amazon MSK > Clusters > MSKCluster-22e523d0

Introducing topic management for Amazon MSK clusters
Amazon MSK's new topic management feature lets you view topics directly through the AWS Management Console, streamlining your workflow. Learn more

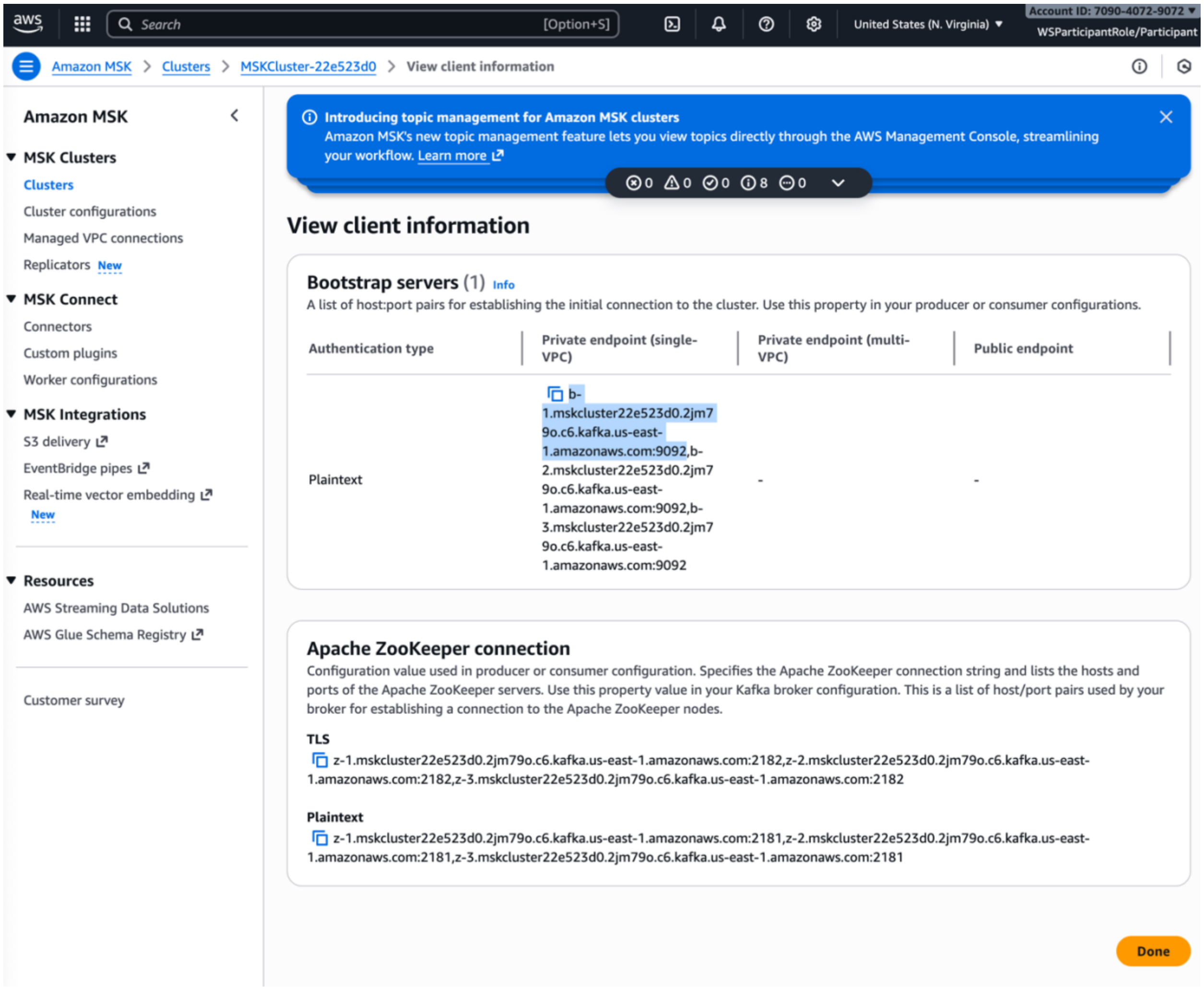
MSKCluster-22e523d0 Info

Cluster summary

Status	User action status	Apache Kafka version	Last modified
Active	-	3.6.0	November 24, 2025 at 12:51 UTC
Cluster type	**Total number of brokers**	**ARN**	**Creation time**
Provisioned	3	arn:aws:kafka:us-east-1:709040729072:cluster/MSKCluster-22e523d0/f2ebfdcb-899d-4db1-9d9b-f304660ca355-6	November 24, 2025 at 12:51 UTC

Amazon CloudWatch metrics Info

UTC timezone



Introducing topic management for Amazon MSK clusters

Amazon MSK's new topic management feature lets you view topics directly through the AWS Management Console, streamlining your workflow. Learn more

View client information

Bootstrap servers (1) Info

A list of host:port pairs for establishing the initial connection to the cluster. Use this property in your producer or consumer configurations.

Authentication type	Private endpoint (single-VPC)	Private endpoint (multi-VPC)	Public endpoint
Plaintext	b-1.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:9092,b-2.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:9092,b-3.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:9092	-	-

Apache ZooKeeper connection

Configuration value used in producer or consumer configuration. Specifies the Apache ZooKeeper connection string and lists the hosts and ports of the Apache ZooKeeper servers. Use this property value in your Kafka broker configuration. This is a list of host/port pairs used by your broker for establishing a connection to the Apache ZooKeeper nodes.

TLS

z-1.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2182,z-2.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2182,z-3.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2182

Plaintext

z-1.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2181,z-2.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2181,z-3.mskcluster22e523d0.2jm79o.c6.kafka.us-east-1.amazonaws.com:2181

Next modify the env file in the console with the bootstrap server you copied

```
vi .env
```

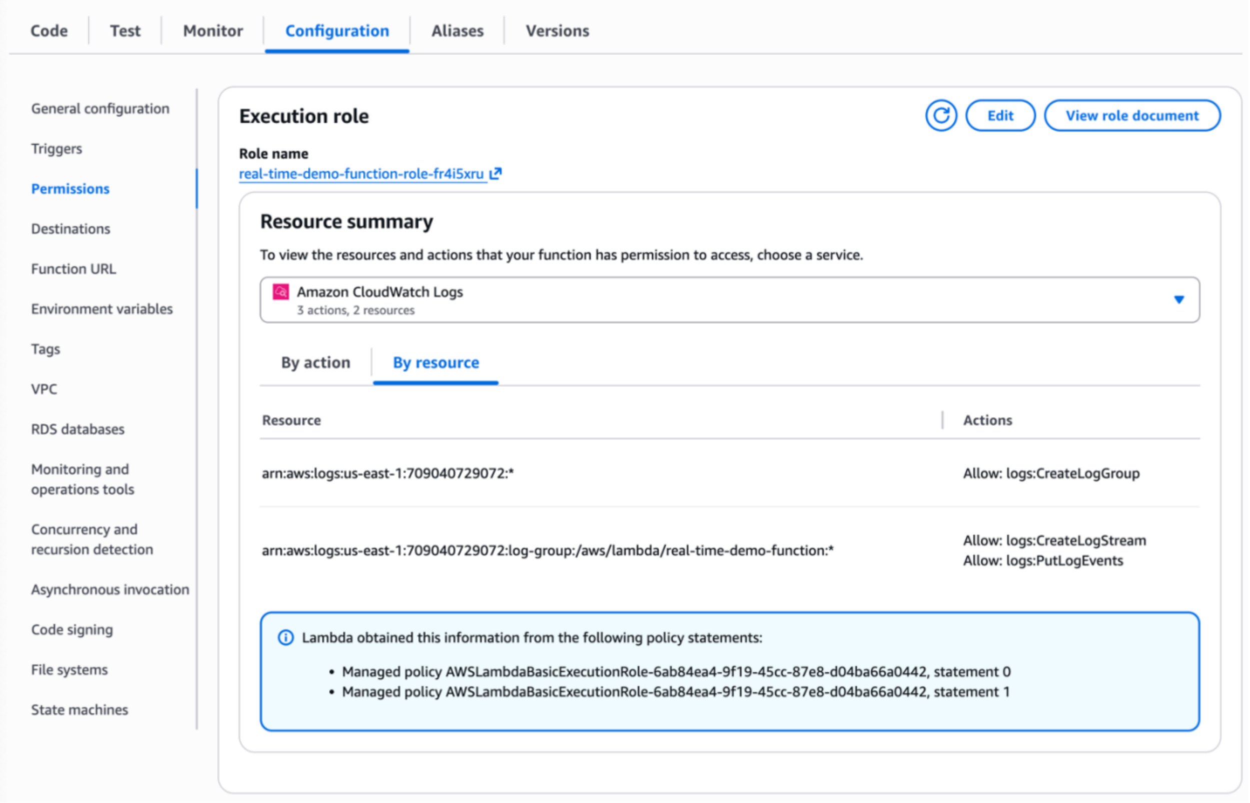
To run the producer, simply execute it.

```
python3 producer.py
```

AWS Lambda function for Data Consumer

- Create a Lambda function using the **Author from scratch** option
- Give it the name `real-time-demo-function`
- Choose the Python 3.13 runtime (**not** 3.14)
- Either architecture is fine
- Under **Additional Configurations**:

    - Choose the VPC created earlier (MSKVPC)
    - Select the **private** subnets (<u>not</u> public)
    - Select the previously created security group starting with **mskconnect-workshop-MSKSecurityGroup**
- Create the Lambda!
- We need to add some variables
- Go to **Configuration**
- Under **Environment Variables** enter following:
    - CRATEDB_HOST (URL excluding prefix and suffixes, just the hostname)
    - CRATEDB_DB (`crate`)
    - CRATEDB_PASS (password you got when your cratedb cluster was created)
    - CRATEDB_USER (`admin`)
    - CRATEDB_PORT (`4200`)
    - SOURCE_TOPIC (**dev-1-0**) (Note the -0 suffix)
- Some additional security permissions are needed to allow our Lambda to access other resources like the MSK cluster.
    - Go to: **Configuration**, then P**ermissions**, then click on the **Role name** at the top to open the Lambda execution role.



Execution role

Role name
real-time-demo-function-role-fr4i5xru

Resource summary

To view the resources and actions that your function has permission to access, choose a service.

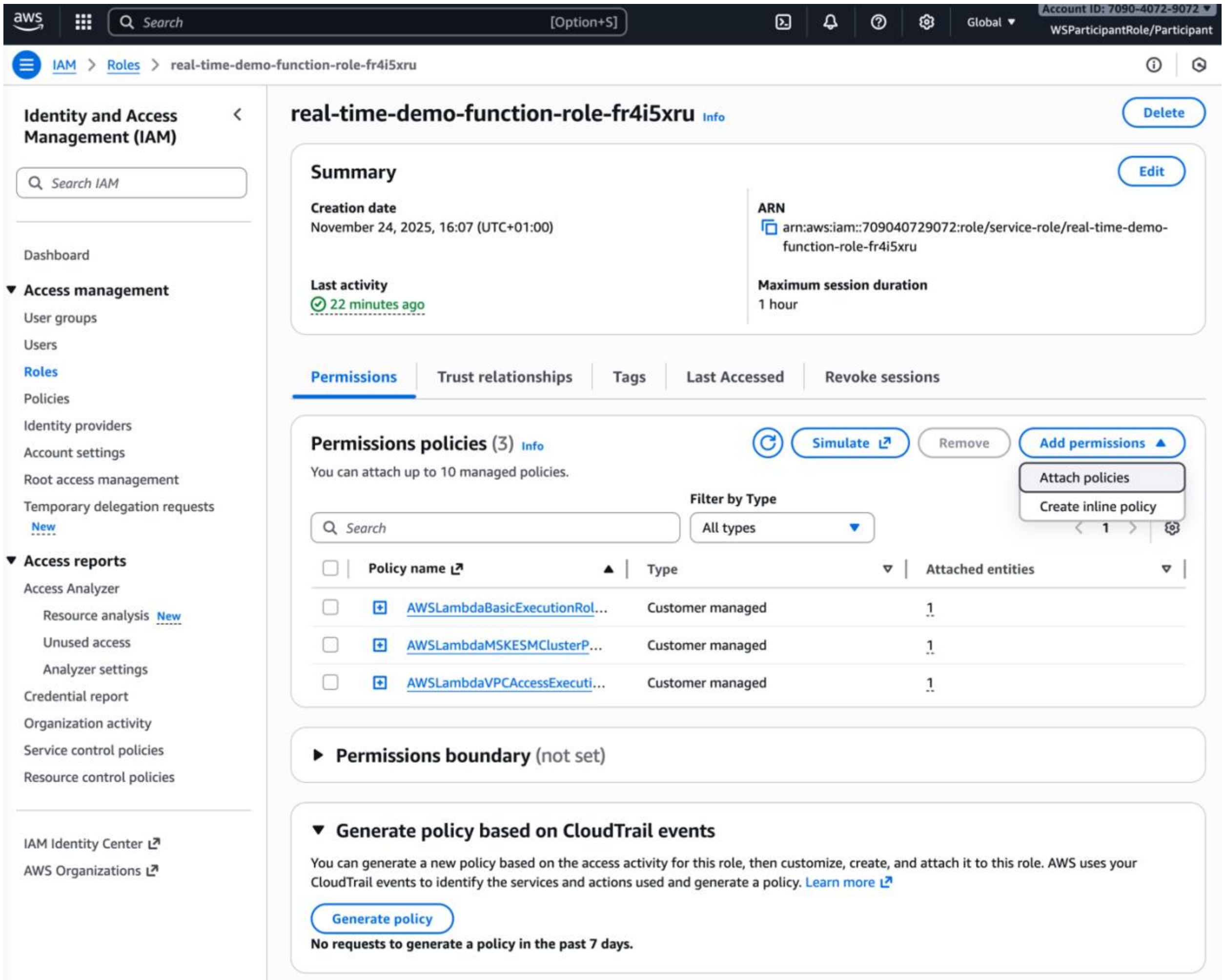
Amazon CloudWatch Logs
3 actions, 2 resources

Resource	Actions
arn:aws:logs:us-east-1:709040729072:*	Allow: logs:CreateLogGroup
arn:aws:logs:us-east-1:709040729072:log-group:/aws/lambda/real-time-demo-function:*	Allow: logs:CreateLogStream Allow: logs:PutLogEvents

Lambda obtained this information from the following policy statements:

- Managed policy AWSLambdaBasicExecutionRole-6ab84ea4-9f19-45cc-87e8-d04ba66a0442, statement 0
- Managed policy AWSLambdaBasicExecutionRole-6ab84ea4-9f19-45cc-87e8-d04ba66a0442, statement 1

- Attach the following policy to the Lambda execution role:
    - `AdministratorAccess` (never do this in a production environment!)



real-time-demo-function-role-fr4i5xru Info

Summary

Creation date: November 24, 2025, 16:07 (UTC+01:00)

ARN: arn:aws:iam::709040729072:role/service-role/real-time-demo-function-role-fr4i5xru

Last activity: 22 minutes ago

Maximum session duration: 1 hour

Permissions policies (3) Info

You can attach up to 10 managed policies.

	Policy name	Type	Attached entities
□	AWSLambdaBasicExecutionRol...	Customer managed	1
□	AWSLambdaMSKESMClusterP...	Customer managed	1
□	AWSLambdaVPCAccessExecuti...	Customer managed	1

▸ Permissions boundary (not set)

▾ Generate policy based on CloudTrail events

You can generate a new policy based on the access activity for this role, then customize, create, and attach it to this role. AWS uses your CloudTrail events to identify the services and actions used and generate a policy. Learn more

No requests to generate a policy in the past 7 days.

Now we upload the Lambda code, this has been provided in the form of a ZIP file release to make it simple to update.

- The link is: https://github.com/crate/realtime-demo/releases/tag/Resources
- Once downloaded go to the **Lambda** page and select the **Code** section
- Go to **Upload from** button which allows the ZIP file to be uploaded.

Create a trigger for the AWS Lambda function

Next step is to create a trigger from MSK when a new value in our demo topic is created. To do this:

- Go to created Lambda Function
- Go to **Configuration**
- Go to **Triggers**
- Select **Add Trigger**
- Select **MSK** as the source
- Find the cluster created earlier (likely named real-time-demo)
- The topic name is `dev-1`
- **Provisioned mode** should be **disabled**
- Use **IAM** authentication
- Set **Starting position** to **At timestamp**
- Starting position is timestamp `2025-10-14`
- Under additional settings, batch size determines how quickly data is ingested, set to `100`
- Click on **Add**

Installing Grafana onto AWS EC2 instance

Update packages

```
sudo dnf update -y
```

Add the Grafana repository

```
sudo tee /etc/yum.repos.d/grafana.repo <<EOF
[grafana]
name=Grafana OSS
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
EOF
```

Install Grafana

```
sudo dnf install grafana -y
```

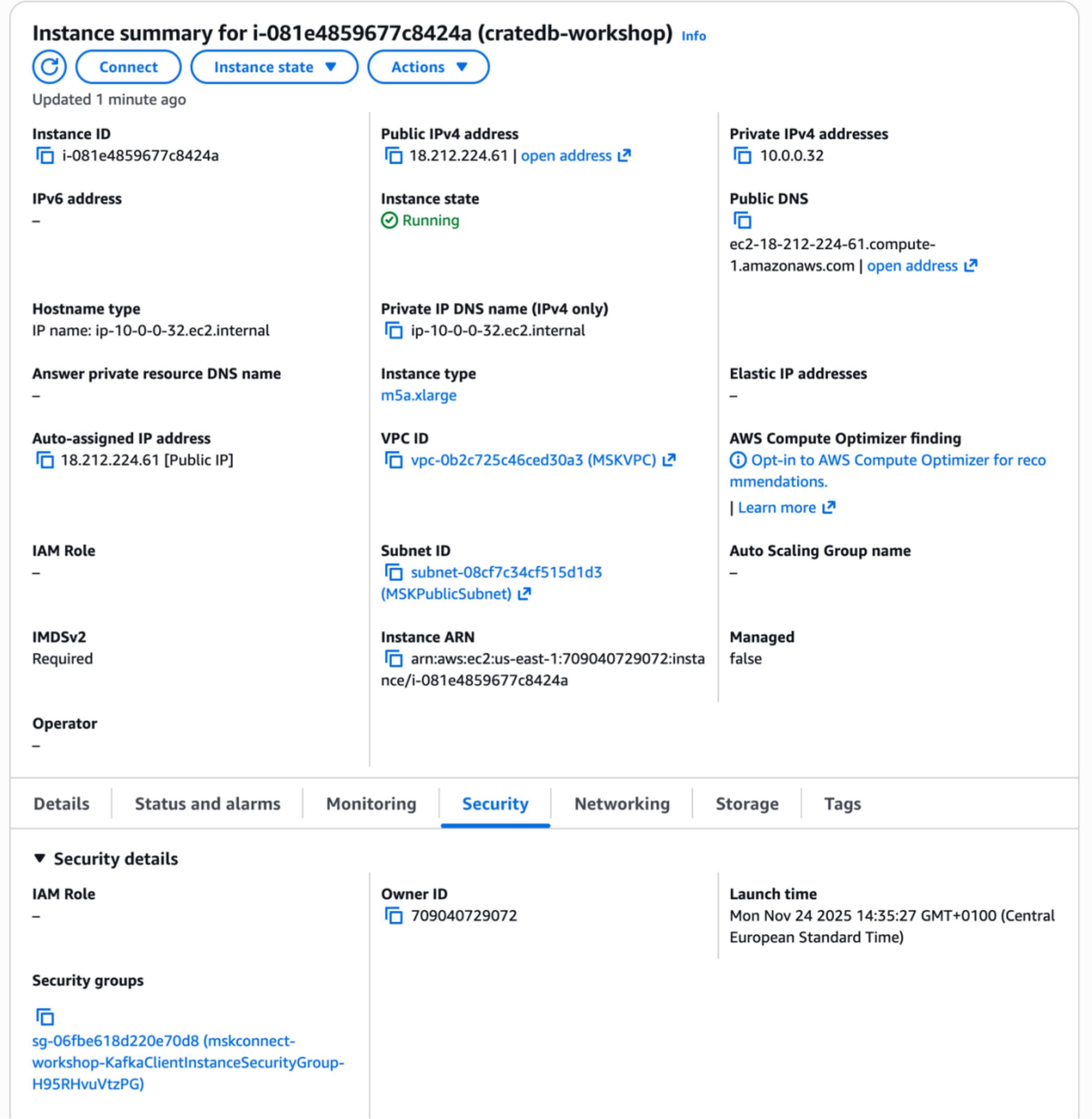
Enable and start the service

```
sudo systemctl enable grafana-server
sudo systemctl start grafana-server
```

Open Grafana port

Edit the security group for the EC2 instance and open port 3000 for the Grafana dashboards.

- Go to EC2
- Select the running workshop instance



Instance summary for i-081e4859677c8424a (cratedb-workshop) Info

Updated 1 minute ago

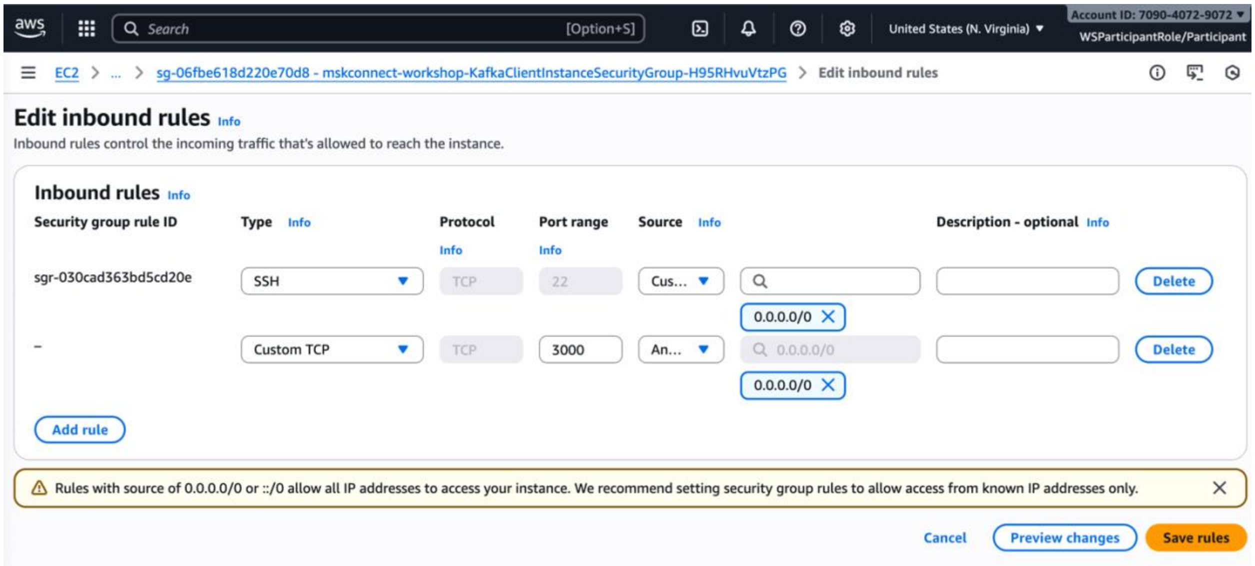
Instance ID i-081e4859677c8424a	**Public IPv4 address** 18.212.224.61 | open address	**Private IPv4 addresses** 10.0.0.32
IPv6 address –	**Instance state** Running	**Public DNS** ec2-18-212-224-61.compute-1.amazonaws.com | open address
Hostname type IP name: ip-10-0-0-32.ec2.internal	**Private IP DNS name (IPv4 only)** ip-10-0-0-32.ec2.internal	
Answer private resource DNS name –	**Instance type** m5a.xlarge	**Elastic IP addresses** –
Auto-assigned IP address 18.212.224.61 [Public IP]	**VPC ID** vpc-0b2c725c46ced30a3 (MSKVPC)	**AWS Compute Optimizer finding** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	**Subnet ID** subnet-08cf7c34cf515d1d3 (MSKPublicSubnet)	**Auto Scaling Group name** –
IMDSv2 Required	**Instance ARN** arn:aws:ec2:us-east-1:709040729072:instance/i-081e4859677c8424a	**Managed** false
Operator –		

▼ Security details

IAM Role –	**Owner ID** 709040729072	**Launch time** Mon Nov 24 2025 14:35:27 GMT+0100 (Central European Standard Time)
Security groups sg-06fbe618d220e70d8 (mskconnect-workshop-KafkaClientInstanceSecurityGroup-H95RHvuVtzPG)		

- Click on the security groups
- Click on **Edit inbound rules**

- Click on **Add rule**
- Select:
    - Type: Customer TCP
    - Port range: 3000
    - Source: Anywhere-IPv4



aws
Search
[Option+S]
United States (N. Virginia) ▼
Account ID: 7090-4072-9072 ▼
WSParticipantRole/Participant
EC2 > ... > sg-06fbe618d220e70d8 - mskconnect-workshop-KafkaClientInstanceSecurityGroup-H95RHvuVtzPG > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-030cad363bd5cd20e	SSH ▼	TCP	22	Cus... ▼	0.0.0.0/0 ✕		Delete
–	Custom TCP ▼	TCP	3000	An... ▼	0.0.0.0/0 0.0.0.0/0 ✕		Delete

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. ✕

- Click on **Save rules**
- Go back to EC2
- Select the running workshop instance
- Make note of the public IPv4 address of the EC2 instance

Import dashboards

Your dashboard should be accessible at http://ip:3000 (where IP is the assigned external IP address of the EC2 instance).

To login, the default Grafana username and passwords are admin/admin, then either skip or set a new password.

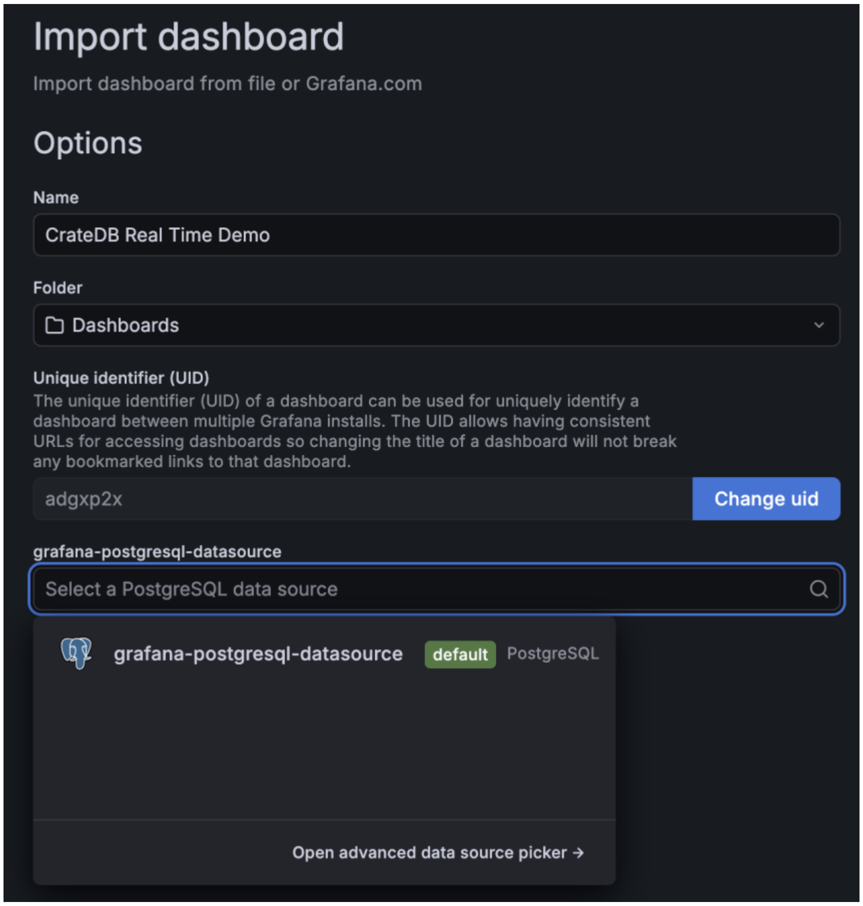
First, we will create a connection to the CrateDB cluster by configuring a corresponding connection.

- Go to **Connections** on the menu on the left
- Click on **Add new connection**
- Search for **PostgreSQL** and add those details from the beginning

  - Name: `grafana-postgresql-datasource`
  - Host URL: (the URL of you CrateDB cloud cluster without any protocol or port)
  - Database name: `crate`
  - Username: `admin`
  - Password: (password from the CrateDB instance)
- Save and test

To set up the dashboard, download both JSON files from https://github.com/crate/realtime-demo/releases/tag/Dashboards-v1.1 and use the import functionality in Grafana.

- Go to **Dashboards**
- Click on **New** and select **Import**
- Select one of the downloaded two JSON documents
- Click on **Import**
- Select the previously created `grafana-postgresql-datasource`



Import dashboard

Import dashboard from file or Grafana.com

Options

Name

CrateDB Real Time Demo

Folder

Dashboards

Unique identifier (UID)

The unique identifier (UID) of a dashboard can be used for uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

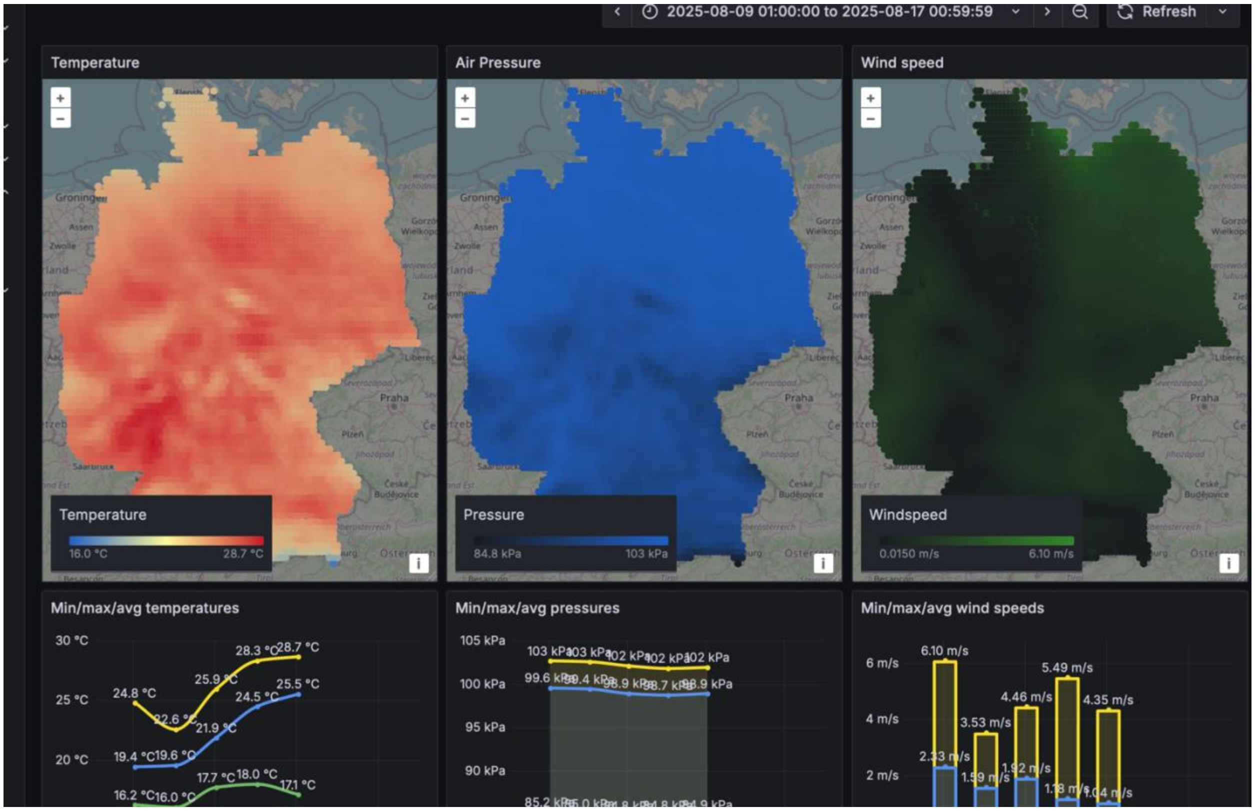
adgxp2x

grafana-postgresql-datasource

Select a PostgreSQL data source

grafana-postgresql-datasource default PostgreSQL

- Upload the second one as well



All complete!

Appendix A

If you want to run the demo again and want to see the visualization in real time building up, simply go back to your query console and truncate the table:

```
delete from demo.climate_data;
```

Then go back to your shell and start the producer again

```
python3 producer.py
```

Now quickly go back to the Grafana dashboard and click frequently on refresh.

Appendix B

Try some queries to see the flexibility in the data model

```
INSERT INTO demo.climate_data (timestamp, geo_location, data) VALUES
(123, [8.78831111111111, 54.903], {"longitude" = 8.78831111111111,
"latitude" = 54.903, "temperature" = 16.868310546875023, "u10" =
4.472952365875244, "v10" = -1.3958832025527954, "pressure" =
102426.1015625});
```

Add a new key in the JSON object without the need to modify the schema

```
INSERT INTO demo.climate_data (timestamp, geo_location, data) VALUES
(123, [8.78831111111111, 54.903], {"newkey" = 'new', "longitude" =
8.78831111111111, "latitude" = 54.903, "temperature" =
16.868310546875023, "u10" = 4.472952365875244, "v10" = -
1.3958832025527954, "pressure" = 102426.1015625});
```

Now try to query the new key

```
select data['newkey'] from demo.climate_data where timestamp = 123;
```

Appendix C

If you like, you can modify the source data to be local to the Netherlands. It will require to get an account for the weather data and the data to be downloaded by the producer code which can take some time. Here are the steps of how to do it.

Configure Climate Data Store API

You need your personal CDS API key. https://cds.climate.copernicus.eu/how-to-api

Then store the key in the data directory

```
echo "url: https://cds.climate.copernicus.eu/api

key: INSERT-YOUR-KEY" > ~/.cdsapirc
```

Change Country to Netherlands (NLD)

In your parser code initialization, change:

```
parser = ClimateParser("DEU")
```

to:

```
parser = ClimateParser("NLD")
```

Clear Old Data (delete table) BEFORE starting the producer

Delete the existing data so the dashboard shows only fresh Netherlands data:

```
delete from demo.climate_data;
```

Start the Producer Again

Now run the producer to download and ingest new data:

```
python3 producer.py
```

The pipeline retrieves ERA5-Land data for the Netherlands, parses it, and produces JSON records.

View Updated Grafana Dashboard

Open Grafana and refresh the dashboard.

It will now display only the newly ingested Netherlands data.