



## Real-time Demo Workshop

## Table of Contents

Table of Contents .....	2
Workshop User Instructions .....	3
Creating AWS Workshop Pre-Requisites .....	3
Creating a CrateDB Cluster.....	9
Creating AWS Workshop Environment .....	11
Installing the demo source code .....	14
Running the data producer .....	14
AWS Lambda function for Data Consumer .....	16
Create a trigger for the AWS Lambda function .....	21
Installing Grafana onto AWS EC2 instance .....	21
Update packages.....	21
Add the Grafana repository .....	21
Install Grafana .....	22
Enable and start the service .....	22
Open Grafana port .....	22
Import dashboards .....	24

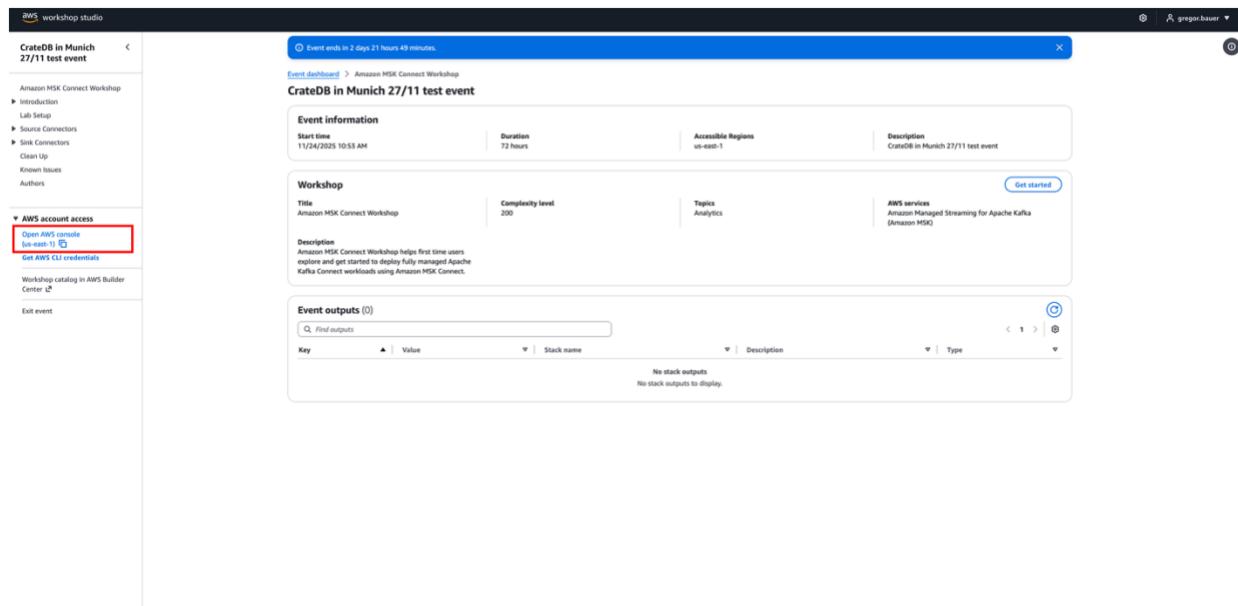
# Workshop User Instructions

## 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.

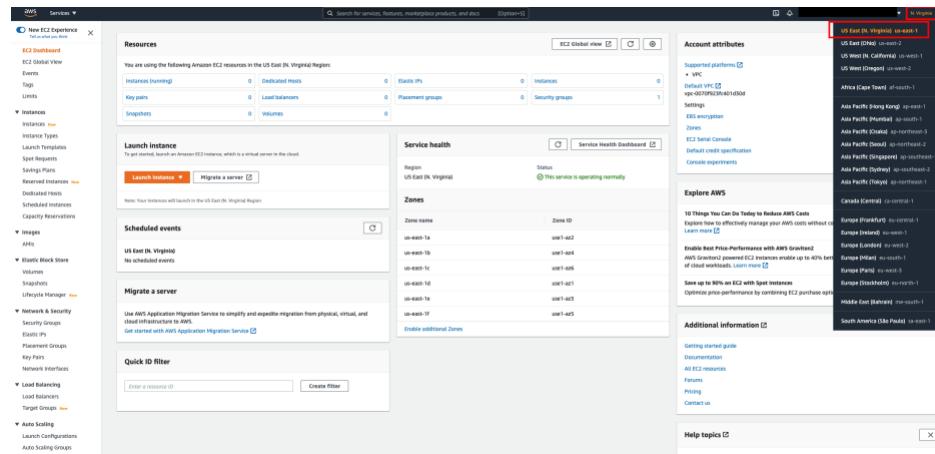
Note: If you already have AWS console it's best to use another browser

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



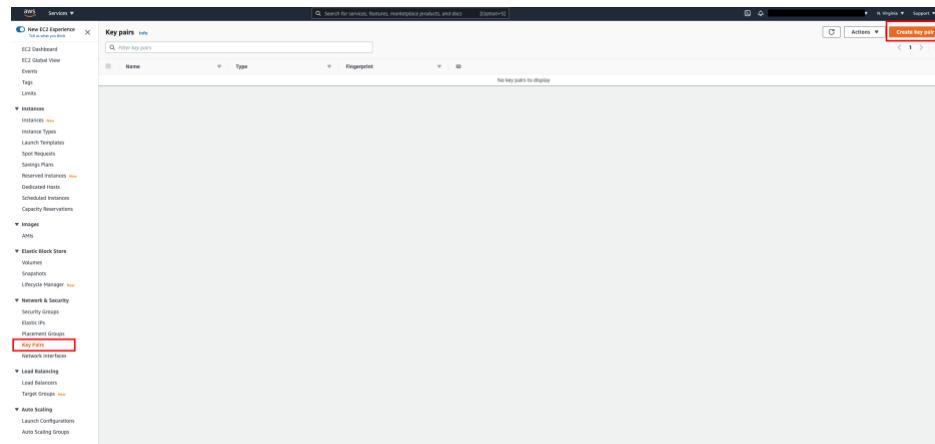
The screenshot shows the AWS workshop studio interface. On the left, there's a sidebar with navigation links like 'Introduction', 'Lab Setup', 'Source Connectors', 'Sink Connectors', 'Clean Up', 'Known Issues', 'Authors', and 'AWS account access'. Under 'AWS account access', there are two buttons: 'Open AWS console (us-east-1)' and 'Get AWS CLI credentials'. The main area is titled 'Event dashboard > Amazon MSK Connect Workshop CrateDB in Munich 27/11 test event'. It displays 'Event information' with start time (11/24/2025 10:51 AM), duration (72 hours), accessible regions (us-east-1), and a description (CrateDB in Munich 27/11 test event). Below that is a 'Workshop' section with a title (Amazon MSK Connect Workshop), complexity level (200), topics (Analytics), and AWS services (Amazon Managed Streaming for Apache Kafka (Amazon MSK)). At the bottom, there's a 'Event outputs (0)' section with a table for stack outputs, which currently shows 'No stack outputs' and '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.



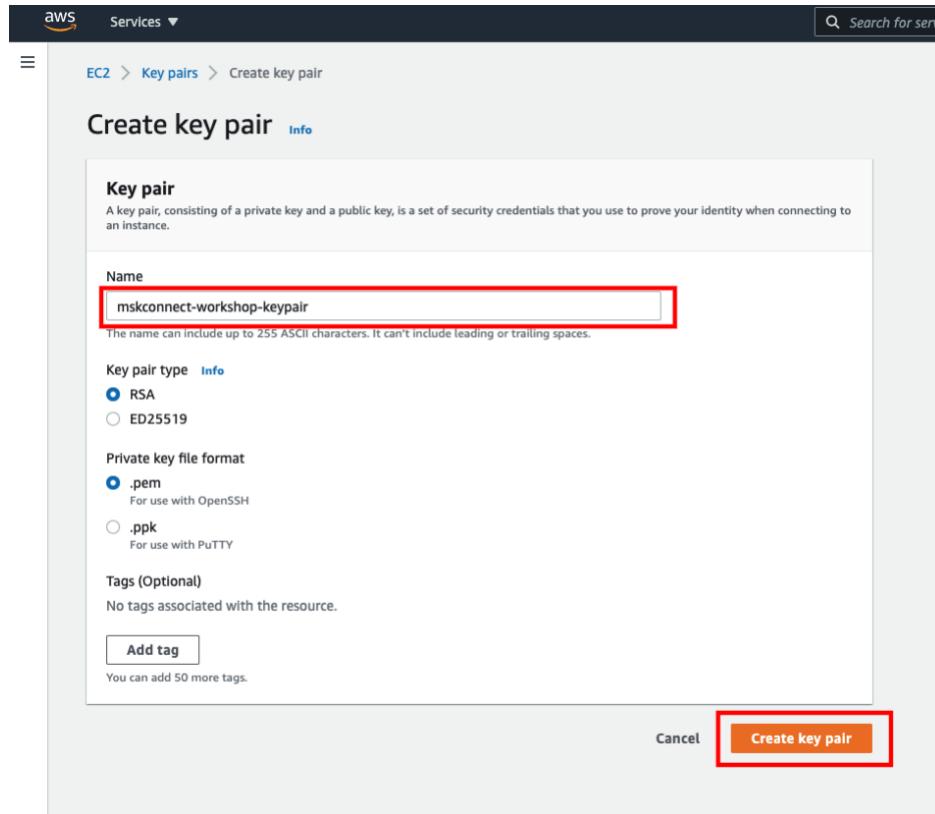
The screenshot shows the AWS EC2 Dashboard. On the left navigation bar, under the 'Key' section, 'Key Pairs' is highlighted. The main content area displays a table for 'Key pairs' with columns for Name, Type, and Region. A red box highlights the 'Create key pair' button at the top right of the table.

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



The screenshot shows the 'Key Pairs' page in the AWS EC2 service. The left navigation bar has 'Key Pairs' selected. The main area shows a table of existing key pairs. At the bottom right of the table, there is a prominent red button labeled 'Create key pair'.

- 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.

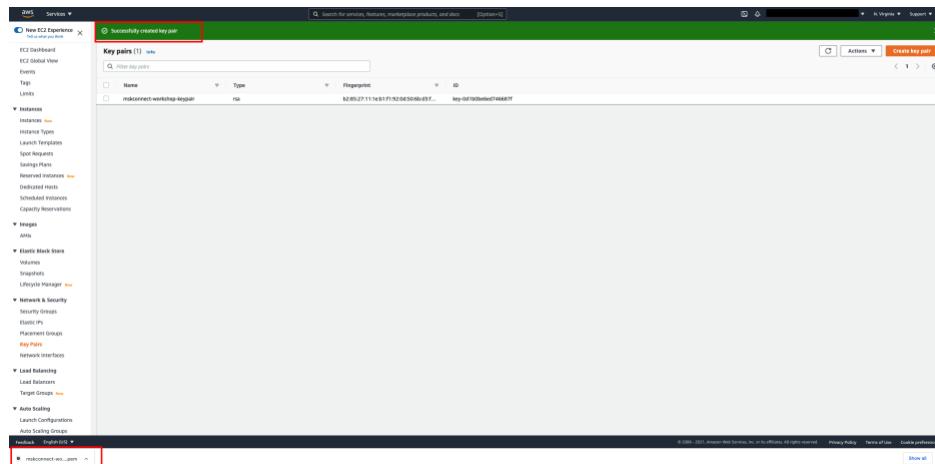
**Name**  
 The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type** Info  
 RSA  
 ED25519

**Private key file format**  
 .pem  
For use with OpenSSH  
 .ppk  
For use with PuTTY

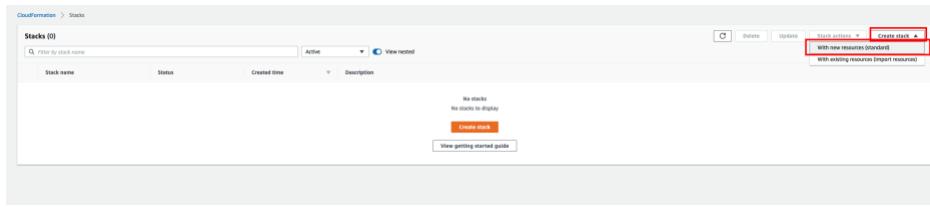
**Tags (Optional)**  
No tags associated with the resource.  
  
You can add 50 more tags.

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

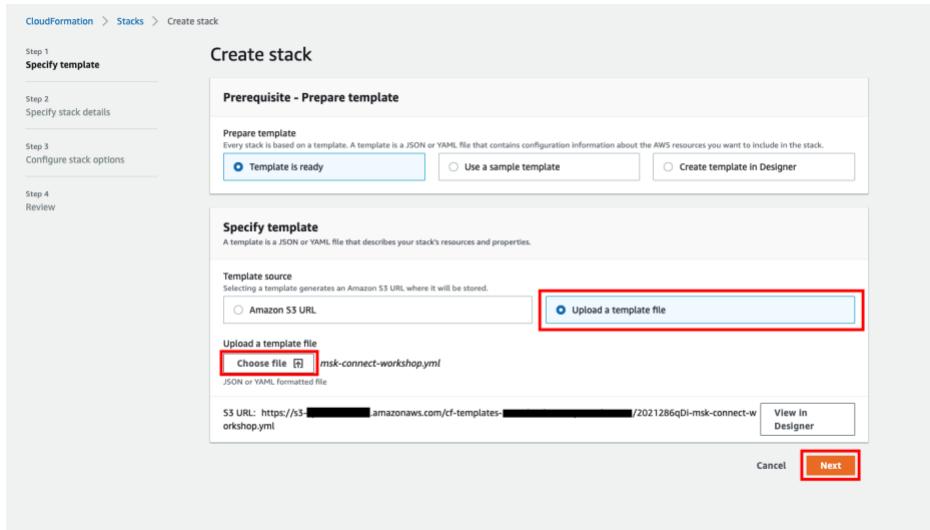


## Deploy required AWS resources via CloudFormation

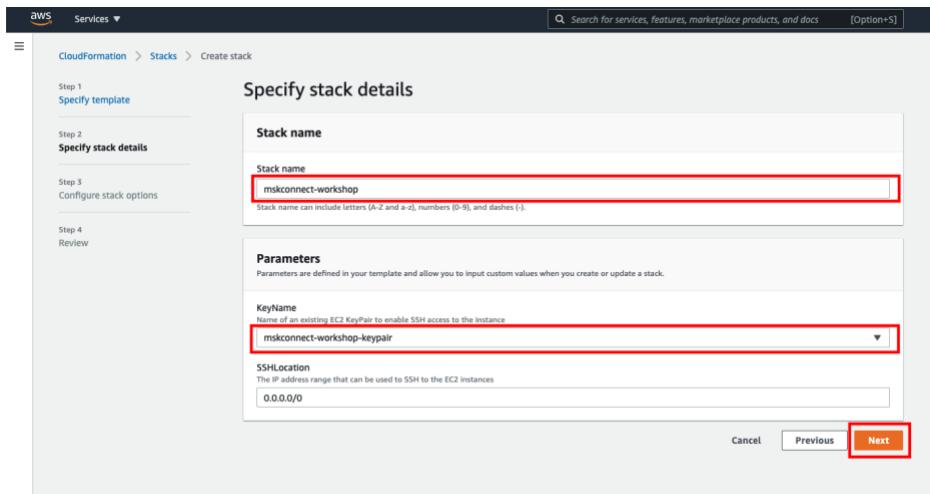
5. Download the CloudFormation template here: [msk-connect-workshop-3.6.0.yml](#)
6. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>
7. Click on **Create stack** and select **With new resources (standard)**.



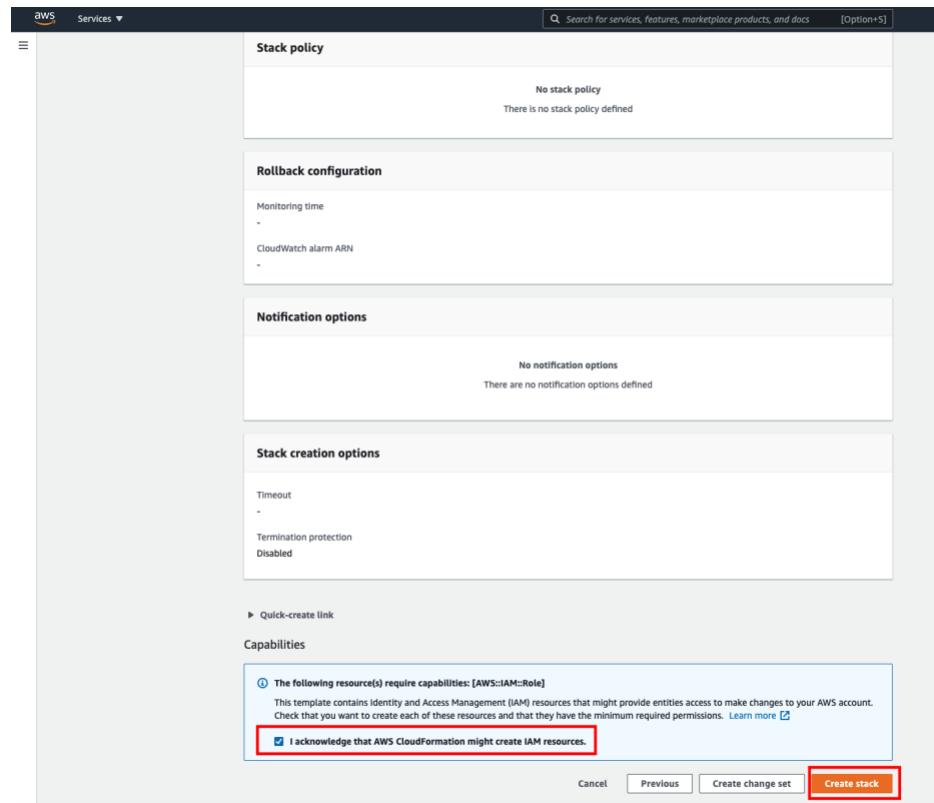
8. Choose **Upload a template file** option and upload the CloudFormation template from your local machine, then click **Next**.



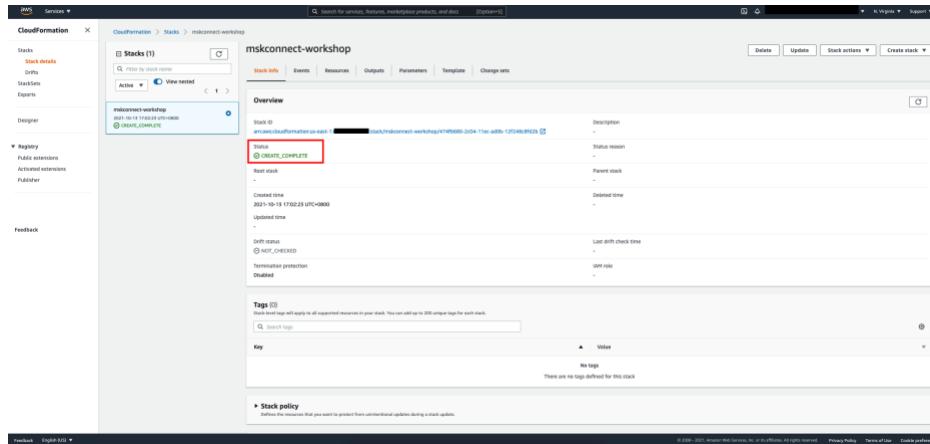
9. Enter **mskconnect-workshop** as the stack name, choose the **mskconnect-workshop-keypair** under **KeyName**, and then click **Next**.



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.



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 45mins).



**What does the above CloudFormation template do?**

The CloudFormation stack creates the following resources:

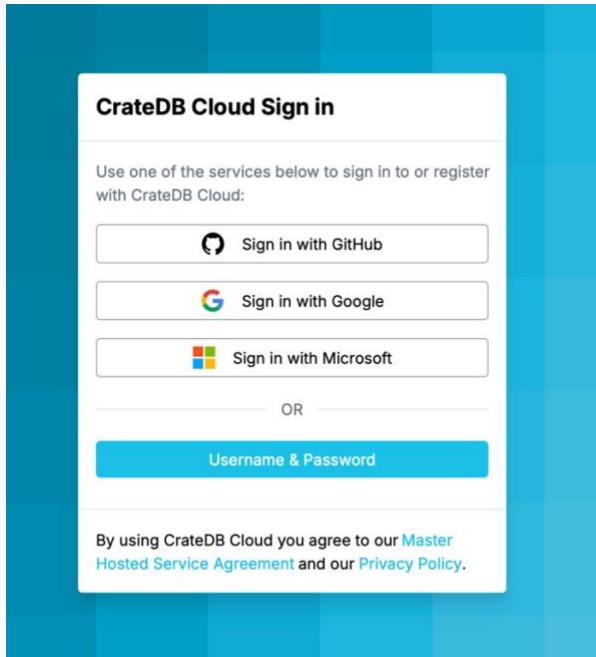
- 2 CloudWatch Log Groups for MSK Cluster and MSK Connect connectors respectively, each with 7 days log retention period
- 1 S3 bucket for storing required Apache Kafka Connect plugin files, and other testing results in some workshop modules
- 1 VPC with 1 Public subnet and 3 Private subnets, and the 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.



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

**Configure your new cluster**

<b>Shared</b> Single-node, suitable for non-production use cases with up to 8 shared vCPUs	<b>Dedicated</b> Scalable up to 9 nodes, ideal for production workloads with up to 144 vCPUs	<b>Custom</b> 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.  
[Learn more about cluster types](#)

**Cloud Provider**

AWS  Azure  Google Cloud

**Region**

AWS AWS US-East (N.Virginia)  AWS US-West (Oregon)  AWS EU (Ireland)  AWS APAC (Singapore)  AWS APAC (Sydney)  AWS APAC (Tokyo)  AWS MEA (Bahrain)  AWS GovCloud (US)

[Request a new region](#)

**Node compute size**

	Up to 2 vCPU	2 GiB RAM	FREE	
CRFREE	Up to 2 vCPU	2 GiB RAM	FREE	<input checked="" type="radio"/>
S2	Up to 2 vCPU	2 GiB RAM	\$0.079 per hour	<input type="radio"/>
S4	Up to 3 vCPU	4 GiB RAM	\$0.158 per hour	<input type="radio"/>
S6	Up to 4 vCPU	6 GiB RAM	\$0.238 per hour	<input type="radio"/>
S12	Up to 8 vCPU	12 GiB RAM	\$0.475 per hour	<input type="radio"/>

**Node storage size**

8 GiB  FREE  16 GiB  \$0.079 per hour  32 GiB  \$0.158 per hour  64 GiB  \$0.238 per hour  128 GiB  \$0.475 per hour

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

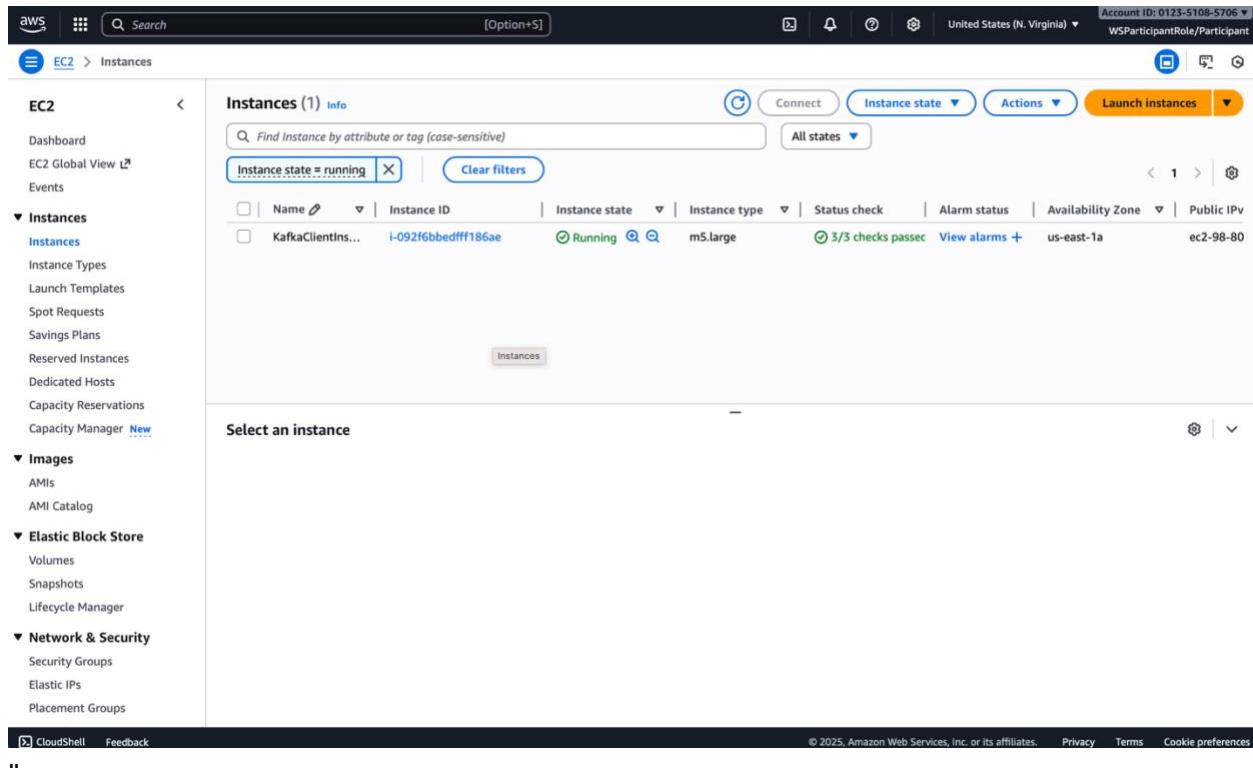
- Select **AWS US-East** as the region (the same as the workshop runs within)
  - Ensure **CRFREE** is selected as compute size
  - Click **Deploy Cluster** to create it, which will take a short amount of time
5. It's **very** important that you save the login credentials, so download or make a note 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,
        "foo" BIGINT
    )
)
```

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**



The screenshot shows the AWS EC2 Instances dashboard. On the left, there's a navigation sidebar with sections like EC2, Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, CloudShell, and Feedback. At the top right, it shows Account ID: 0123-5108-5706, United States (N. Virginia), and WSParticipantRole/Participant. The main area displays 'Instances (1) Info' with a search bar and filters for Instance state (set to 'running') and Instance type (m5.large). A single instance is listed: KafkaClientIns... (Instance ID: i-092f6bbbedffff186ae), which is Running and has 3/3 checks passed. It is located in us-east-1a with a Public IPv4 of ec2-98-80. Below the table is a section titled 'Select an instance'.

**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

| Create new subnet ↗

vpc-0b5b5297304467acb (default)  
172.31.0.0/16

vpc-0b2c725c46ced30a3 (MSKVPC)  
10.0.0.0/16 ✓

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

<input type="checkbox"/> default VPC: vpc-0b2c725c46ced30a3	sg-00a6722507cce9fe6
<input type="checkbox"/> mskconnect-workshop-MSKSecurityGroup-MeLw1WbWW9XP VPC: vpc-0b2c725c46ced30a3	sg-0690e55f9701a8152
<input checked="" type="checkbox"/> mskconnect-workshop-KafkaClientInstanceSecurityGroup-H95RHvuvTzPG VPC: vpc-0b2c725c46ced30a3	sg-06fbe618d220e70d8

work interfaces.

Advanced

- Configure storage: **30GB**
- Ensure it's created in a **public** subnet, not private
- Ensure a public elastic IP is assigned

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

Last updated 1 minute ago

Instances (1/2) [Info](#)

Find Instance by attribute or tag (case-sensitive)

All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input checked="" type="checkbox"/> cratedb-works...	i-081e4859677c8424a	 Running	m5.xlarge	 Initializing	<a href="#">View alarms +</a>
<input type="checkbox"/> KafkaClientIns...	i-0c4f0fda398d12d35	 Running	m5.large	 3/3 checks passed	<a href="#">View alarms +</a>

  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.

 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.

[Cancel](#) [Connect](#)

## 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.

aws | Search [Option+S] | United States (N. Virginia) | Account ID: 7090-4072-9072 | WSParticipantRole/Participant

Amazon MSK > Clusters > MSKCluster-22e523d0

**Amazon MSK**

- ▼ **MSK Clusters**
  - Clusters
  - Cluster configurations
  - Managed VPC connections
  - Replicators [New](#)
- ▼ **MSK Connect**
  - Connectors
  - Custom plugins
  - Worker configurations
- ▼ **MSK Integrations**
  - S3 delivery [New](#)
  - EventBridge pipes [New](#)
  - Real-time vector embedding [New](#)
- ▼ **Resources**
  - AWS Streaming Data Solutions
  - AWS Glue Schema Registry [New](#)
- Customer survey

**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](#) [Actions](#)

**Cluster summary**

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

[View client information](#)

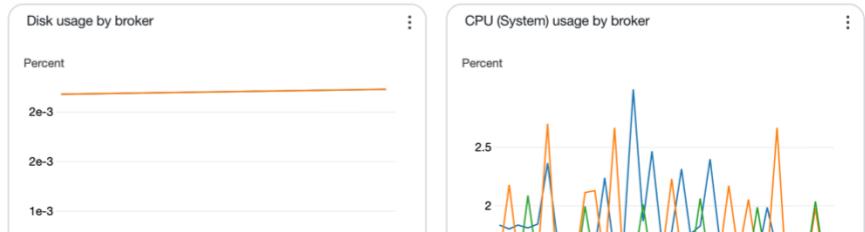
[Metrics](#) Properties Topics - new Cluster operations Cluster alerts (0) Tags (0) Lambda >

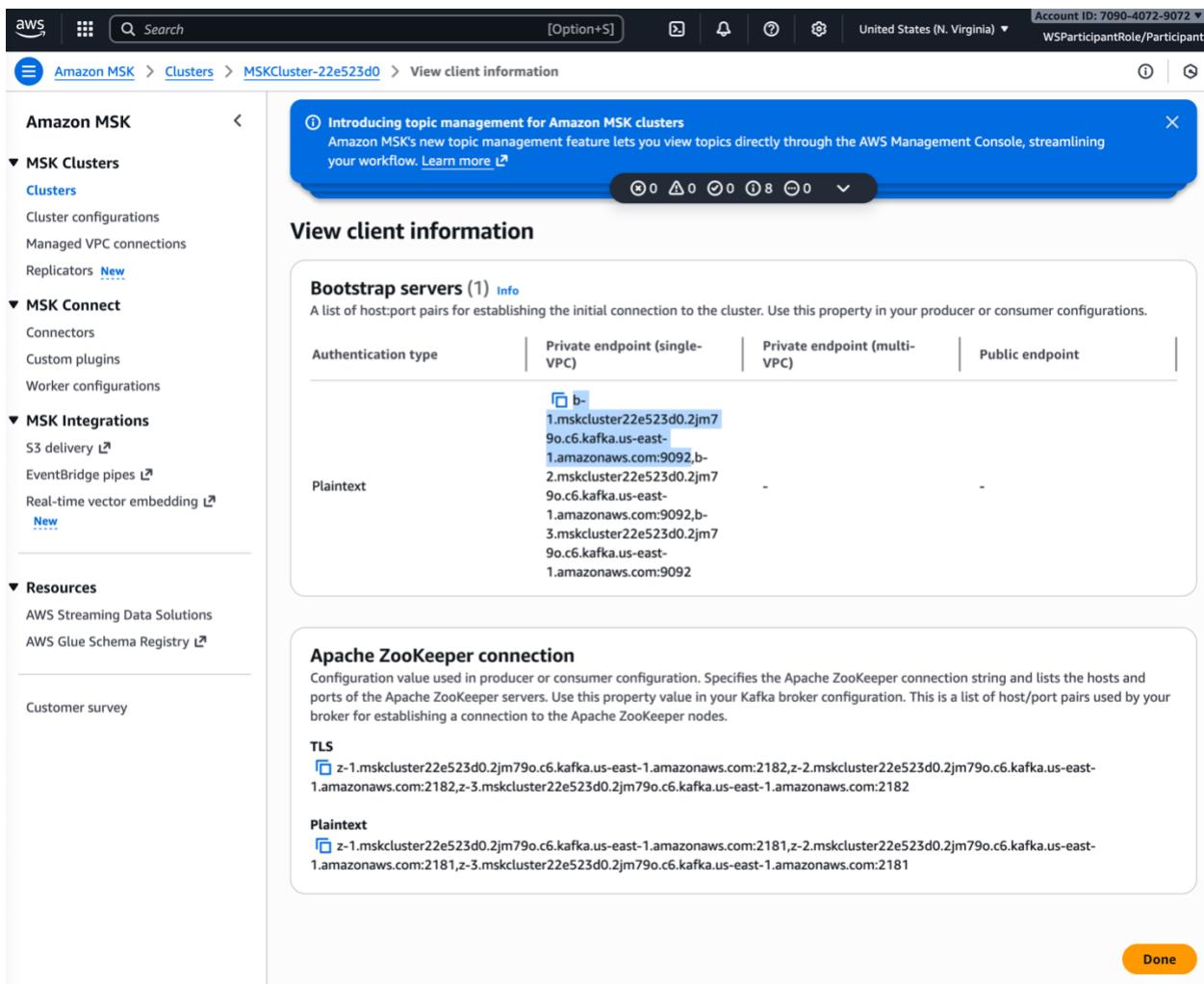
**Amazon CloudWatch metrics** [Info](#) [Create CloudWatch alarm](#)

[Investigate with AI - new](#) 3h 1d 1w 3h UTC timezone [Explore related](#) :

Disk usage by broker

CPU (System) usage by broker





The screenshot shows the AWS Management Console interface for an MSK cluster named 'MSKCluster-22e523d0'. The left sidebar navigation includes 'Amazon MSK', 'MSK Clusters', 'MSK Connect', 'MSK Integrations', and 'Resources'. The main content area is titled 'View client information' and contains two sections: 'Bootstrap servers (1) Info' and 'Apache ZooKeeper connection'. The 'Bootstrap servers' section lists host:port pairs for plaintext and TLS connections. The 'Apache ZooKeeper connection' section lists host:port pairs for both TLS and plaintext connections. A blue banner at the top right provides information about topic management for Amazon MSK clusters.

Authentication type	Private endpoint (single-VPC)	Private endpoint (multi-VPC)	Public endpoint
Plaintext	<input checked="" type="checkbox"/> 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	-	-
TLS	<input checked="" type="checkbox"/> 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	<input checked="" type="checkbox"/> 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 (not 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 **Permissions**, then click on the **Role name** at the top to open the Lambda execution role.

Code    Test    Monitor    **Configuration**    Aliases    Versions

General configuration  
Triggers  
**Permissions**  
Destinations  
Function URL  
Environment variables  
Tags  
VPC  
RDS databases  
Monitoring and operations tools  
Concurrency and recursion detection  
Asynchronous invocation  
Code signing  
File systems  
State machines

**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

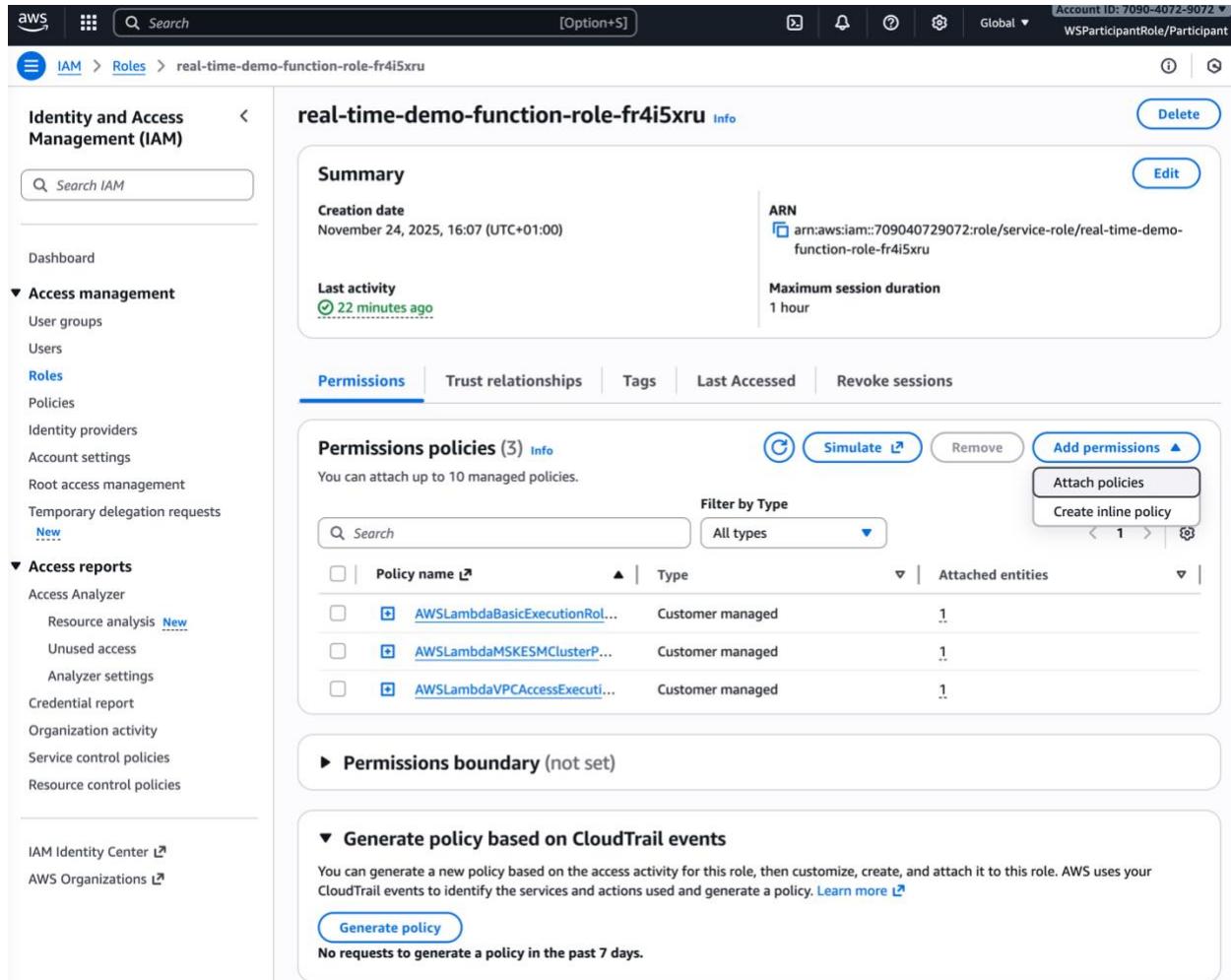
By action 

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

- Add the following permissions to the Lambda execution role:
  - **AdministratorAccess** (never do this in a production environment!)



The screenshot shows the AWS IAM Roles page. The role name is "real-time-demo-function-role-fr4i5xru". The ARN is listed as "arn:aws:iam::709040729072:role/service-role/real-time-demo-function-role-fr4i5xru". The maximum session duration is set to 1 hour. The "Permissions" tab is selected, showing three attached policies: "AWSLambdaBasicExecutionRole", "AWSLambdaMSKESMClusterPolicy", and "AWSLambdaVPCAccessExecutionRole". There is also a "Permissions boundary" section which is currently not set. A "Generate policy based on CloudTrail events" button is available, but it says "No requests to generate a policy in the past 7 days".

- After that add a new inline policy to allow the Lambda execution role to access MSK (this rule allows access to all MSK clusters)

aws  Search [Option+S] Global Account ID: 7090-4072-9072 WSParticipantRole/Participant

☰ IAM > Roles > real-time-demo-function-role-fr4i5xru > Create policy

Step 1  
 Specify permissions  
 Review and create

### Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

**Policy editor**

```

1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Sid": "Statement1",
6              "Effect": "Allow",
7              "Action": "kafka:DescribeCluster",
8              "Resource": "arn:aws:kafka:*::cluster/*"
9          }
10     ]
11 }
```

Visual  JSON Actions ▾ 

Edit statement **Statement1** Remove

Add actions Choose a service

Available

- AI Operations
- AMP
- API Gateway
- API Gateway V2
- ARC Region switch
- ARC Zonal Shift
- ASC
- Access Analyzer
- Account

Add a resource

Add a condition (optional)

+ Add new statement

JSON Ln 7, Col 14 10138 of 10240 characters remaining

Security: 0 Errors: 0 Warnings: 0 Suggestions: 2

- Select JSON and paste the code below

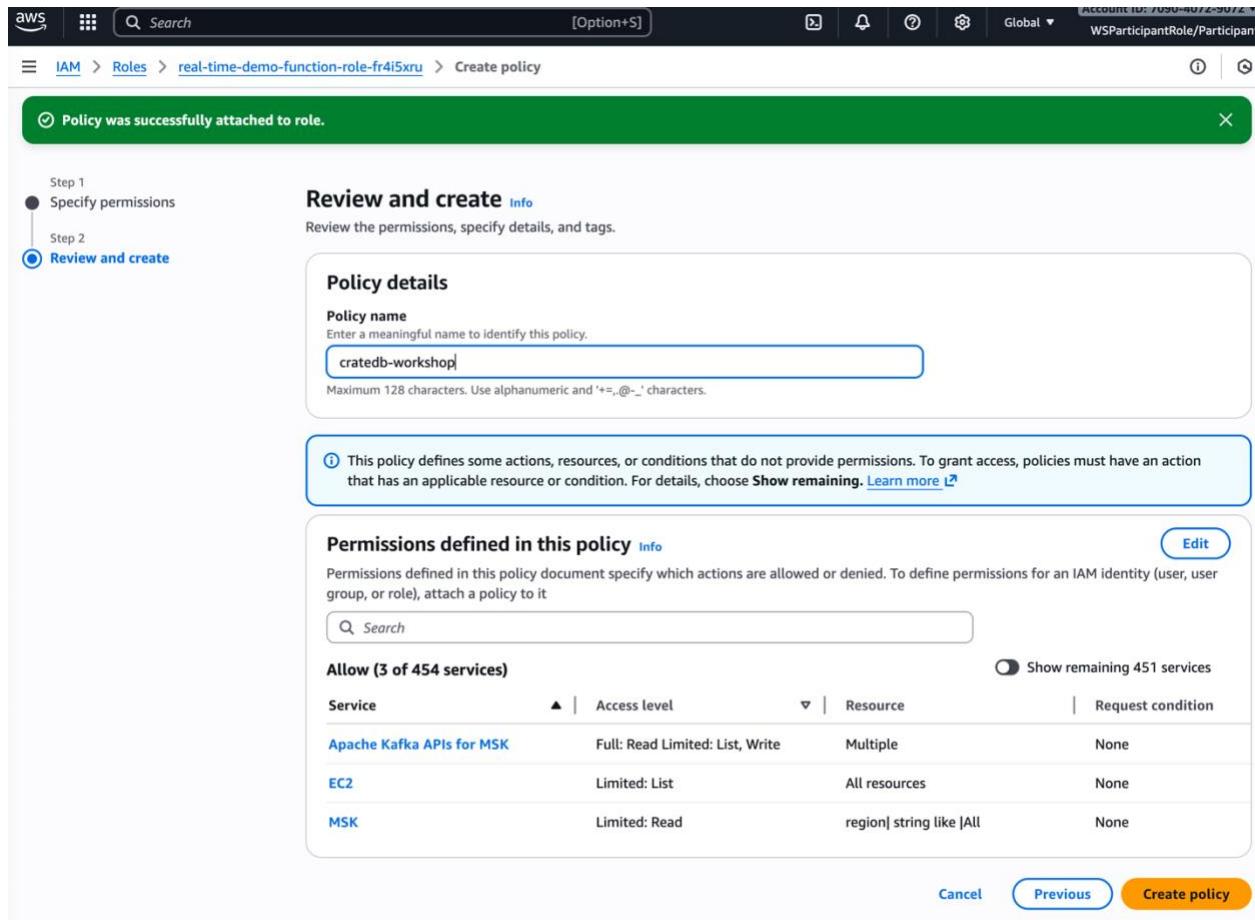
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kafka:DescribeCluster",
                "kafka:DescribeClusterV2",
                "kafka:GetBootstrapBrokers"
            ],
            "Resource": "arn:aws:kafka:*::cluster/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:Connect",
                "kafka-cluster:AlterCluster",
                "kafka-cluster:DescribeCluster",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:ReadData",
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ]
        }
    ]
}
```

```

        ],
        "Resource": [
            "arn:aws:kafka:*::cluster/*",
            "arn:aws:kafka:*::topic/*",
            "arn:aws:kafka:*::group/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs"
        ],
        "Resource": "*"
    }
]
}

```

- Click on Next and name this **cratedb-workshop**
- Create the policy



Policy was successfully attached to role.

**Review and create** Info

Review the permissions, specify details, and tags.

**Policy details**

**Policy name**  
Enter a meaningful name to identify this policy.  
**cratedb-workshop**

This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition. For details, choose [Show remaining](#). Learn more

**Permissions defined in this policy** Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Service	Access level	Resource	Request condition
Apache Kafka APIs for MSK	Full: Read Limited: List, Write	Multiple	None
EC2	Limited: List	All resources	None
MSK	Limited: Read	region  string like  All	None

[Edit](#) [Cancel](#) [Previous](#) [Create policy](#)

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](#)

[Connect](#) [Instance state](#) [Actions](#)

Updated 1 minute ago

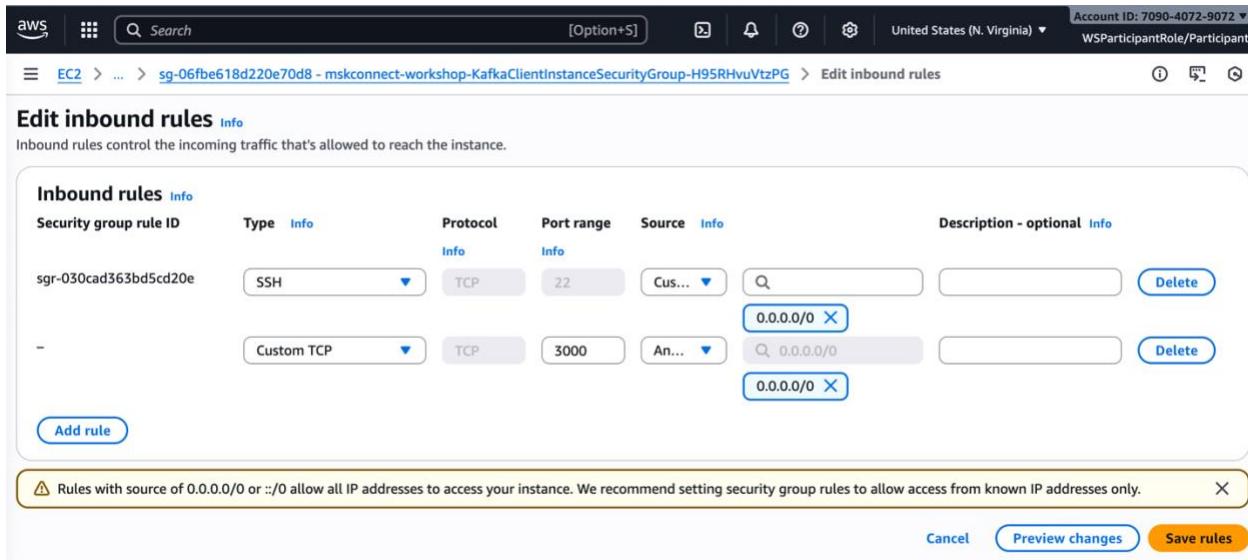
Instance ID <a href="#">i-081e4859677c8424a</a>	Public IPv4 address <a href="#">18.212.224.61</a>   <a href="#">open address ↗</a>	Private IPv4 addresses <a href="#">10.0.0.32</a>
IPv6 address -	Instance state <a href="#">Running</a>	Public DNS <a href="#">ec2-18-212-224-61.compute-1.amazonaws.com</a>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

[Details](#) [Status and alarms](#) [Monitoring](#) [Security](#) [Networking](#) [Storage](#) [Tags](#)

[▼ Security details](#)

IAM Role -	Owner ID <a href="#">709040729072</a>	Launch time Mon Nov 24 2025 14:35:27 GMT+0100 (Central European Standard Time)
Security groups  <a href="#">sg-06fbe618d220e70d8 (mskconnect-workshop-KafkaClientInstanceSecurityGroup-H95RHvuVtzPG)</a>		

- Click on the security groups
- Click on **Edit inbound rules**
- Click on **Add rule**
- Select:
  - Type: Customer TCP
  - Port range: 3000
  - Source: Anywhere-IPv4



**Edit inbound rules** Info

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

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-030cad363bd5cd20e	SSH	TCP	22	Cus... <input type="button" value="Edit"/>	<input type="text" value="0.0.0.0/0"/> <input type="button" value="Delete"/>
-	Custom TCP	TCP	3000	An... <input type="button" value="Edit"/>	<input type="text" value="0.0.0.0/0"/> <input type="button" value="Delete"/>

[Add rule](#)

⚠️ 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. [X](#)

[Cancel](#) [Preview changes](#) [Save rules](#)

- 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.

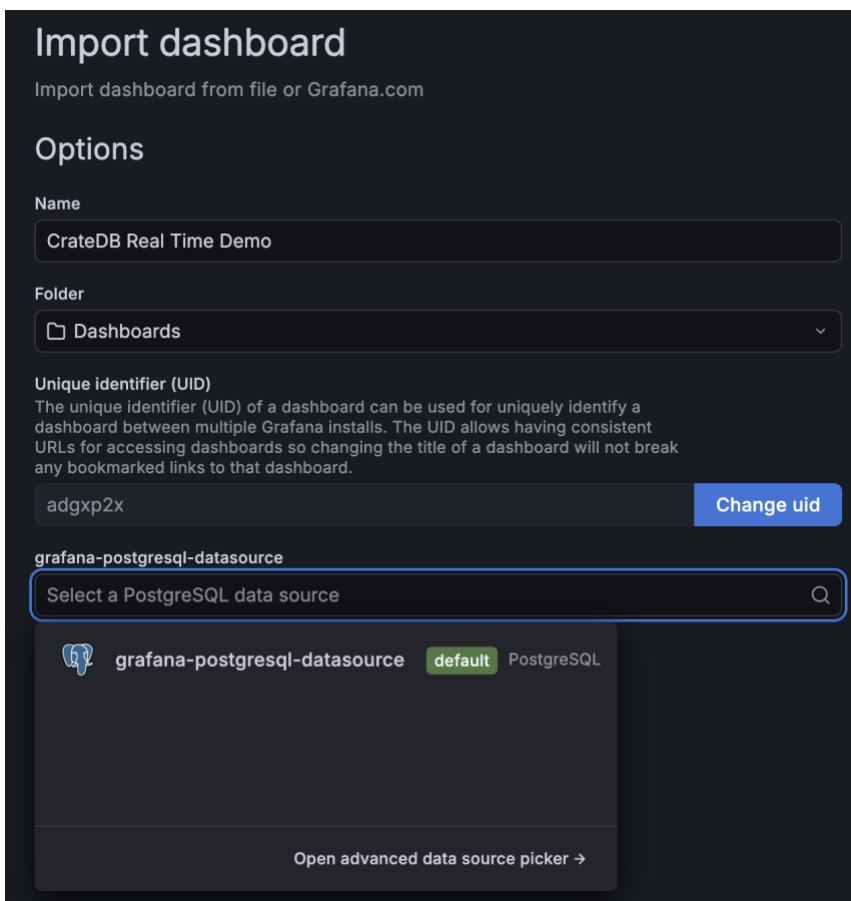
- Go to **Connections** on the menu on the left
- Click on **Add new connection**
- Search for **PostgreSQL** and add those details from the beginning
  - Name: **grafana-postgresql-datasource**
  - Host URL: (the URL of your CrateDB cloud cluster without any protocol or port)
  - Database name: **crate**

- Username: **admin**
- Password: (password from the CrateDB instance)
- Save and test

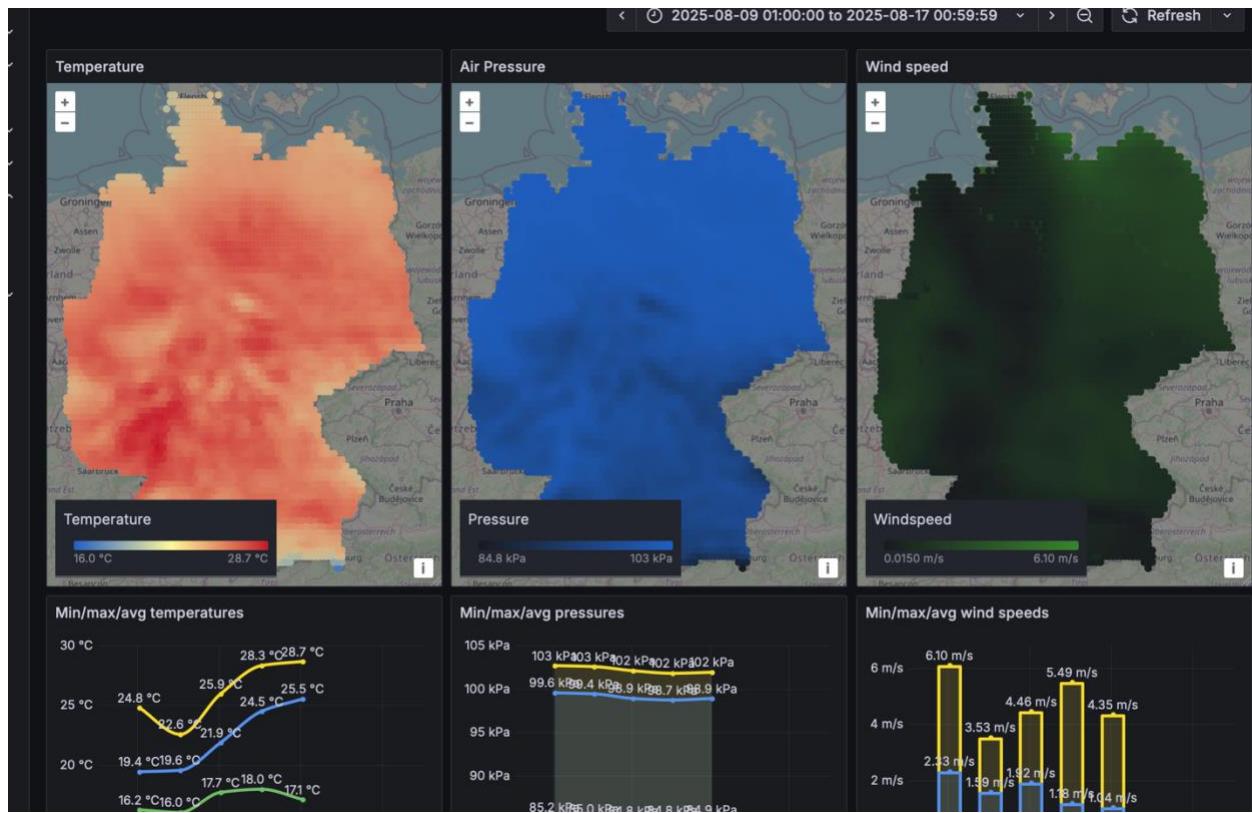
To set up the dashboard, download both JSON files from

<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**



- Upload the second one as well



All complete!