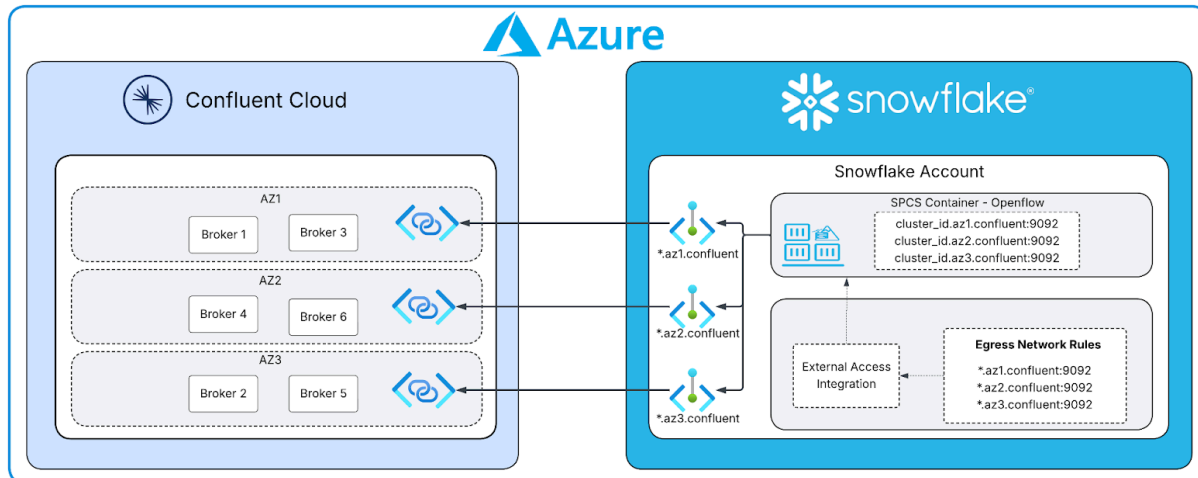


Confluent Cloud Azure Kafka Dedicated Clusters



- [1. Confluent Networking details](#)
- [2. Provision Snowflake private endpoint\(s\)](#)
- [3. Allowlist Snowflake subscription ID in Confluent](#)
- [4. Create a Network Rule](#)
- [5. Create an External Access Integration](#)
- [6. Download and install the Notebook to validate the configuration](#)
- [7. Enable the PostgreSQL EAI in the Notebook](#)
- [8. Configure the variables and execute the Notebook](#)

Note: this guide applies to outbound private connectivity from an Azure Snowflake account to a **dedicated** Kafka cluster in Confluent Cloud Azure.

The example used below refers to a multi-az cluster, but also applies to single-az clusters.

Important: all the values used in the examples below are for demonstration purposes. Replace all values with your Kafka cluster information.

1. Confluent Networking details

1. Access your Confluent Cloud portal
2. Navigate to your dedicated Cluster Overview > Networking > Details of our attached private network configuration
3. Gather the Availability Zone (AZ) specific **DNS domain** and **Resource ID** for each AZ

uksouth-network-private

Network overview	
ID	n-gz1vq7
Provider	Azure
Region	uksouth
Connection types	Private Link
Confluent Cloud VNet	s-9vezm
Confluent Cloud Subscription ID	54723f8a-2676-4ace-b32f-7e6003485991
DNS resolution	Private
DNS domain ⓘ	dom6pke3rmp.uksouth.azure.confluent.cloud
Endpoint suffix ⓘ	<service-identifier>.dom6pke3rmp.uksouth.azure.confluent.cloud
Availability zone 3	
• DNS subdomain	az3.dom6pke3rmp.uksouth.azure.confluent.cloud
• Service Alias	s-9vezm-privatelink-3.33a9bb3d-4068-497f-8929-61cb99dc275f.uksouth.azure.privatelinkservice
• Resource ID	/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-3
Availability zone 2	
• DNS subdomain	az2.dom6pke3rmp.uksouth.azure.confluent.cloud
• Service Alias	s-9vezm-privatelink-2.8aaf00ba-1fdf-4aad-b3db-c624a8aa0a0c.uksouth.azure.privatelinkservice
• Resource ID	/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-2
Availability zone 1	
• DNS subdomain	az1.dom6pke3rmp.uksouth.azure.confluent.cloud
• Service Alias	s-9vezm-privatelink-1.292a073b-496e-4789-aa8f-7fd221ab6351.uksouth.azure.privatelinkservice
• Resource ID	/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-1

2. Provision Snowflake private endpoint(s)

Provision one private endpoint per AZ

SQL

```
SELECT SYSTEM$PROVISION_PRIVATELINK_ENDPOINT(
  '/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-1'
  , '*.az1.dom6pke3rmp.uksouth.azure.confluent.cloud'); //Availability Zone #1
```

```
SELECT SYSTEM$PROVISION_PRIVATELINK_ENDPOINT(
  '/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-2'
  , '*.az2.dom6pke3rmp.uksouth.azure.confluent.cloud'); //Availability Zone #2
```

```
SELECT SYSTEM$PROVISION_PRIVATELINK_ENDPOINT(
  '/subscriptions/54723f8a-2676-4ace-b32f-7e6003485991/resourceGroups/s-9vezm/providers/Microsoft.Network/privateLinkServices/s-9vezm-privatelink-3'
  , '*.az3.dom6pke3rmp.uksouth.azure.confluent.cloud'); //Availability Zone #3
```

Gather the **Subscription ID** from the private endpoint resource ID in the **output** of the above function, for example:

```
/subscriptions/c4163da0-f07a-42ab-a254-9ca2bf882e98/resourceGroups/azw  
esteurope-snowplex-rg/providers/Microsoft.Network/privateEndpoints/0c0  
f8a7d-7721-4c25-9fa8-0925f7d72cbb
```

3. Allowlist Snowflake subscription ID in Confluent

In Confluent Cloud, under Network Management, create a new **Private Link Access**:

Add Private Link access

Azure Private Link allows for one-way secure connection access from your VNet to Confluent Cloud with an added protection against data exfiltration. [Learn more](#)

Name*

Snowflake inbound

Step 1

Register your Azure Subscription to allow Private Link connection requests from your Azure Subscription to this cluster

Azure subscription ID* ⓘ

c4163da0-f07a-42ab-a254-9ca2bf882e98

You can find your Azure subscription ID in the subscription section of your [Microsoft Azure Portal](#). Must be a **valid 32 character UUID string**.

Step 2

In your [Microsoft Azure Portal](#) create private endpoint(s) to the following private link service alias(es):

s-9vezm-privatelink-1.292a073b-496e-4789-aa8f-7fd221ab6351.uksouth.azure.privatelinkservice ⓘ

s-9vezm-privatelink-2.8aaf00ba-1fdf-4aad-b3db-c624a8aa0a0c.uksouth.azure.privatelinkservice ⓘ

s-9vezm-privatelink-3.33a9bb3d-4068-497f-8929-61cb99dc275f.uksouth.azure.privatelinkservice ⓘ

Cancel

Add

After these steps, the private endpoints connection should be in **Approved** status in Snowflake:

SQL

SELECT

```
parsed_value:provider_resource_id::STRING  
parsed_value:snowflake_resource_id::STRING
```

```
AS provider_resource_id,  
AS snowflake_resource_id,
```

```

        parsed_value:host::STRING                AS host,
        parsed_value:endpoint_state::STRING       AS endpoint_state,
        parsed_value:subresource::STRING         AS subresource,
        parsed_value:status::STRING              AS status
FROM TABLE(
    FLATTEN(
        INPUT => PARSE_JSON(SYSTEM$GET_PRIVATELINK_ENDPOINTS_INFO())
    )
),
LATERAL (
    SELECT PARSE_JSON(value) AS parsed_value
)
WHERE HOST ILIKE '%azure.confluent.cloud';

```

4. Create a Network Rule

In Snowflake, create the Network Rule and External Access Integration

```

SQL
CREATE NETWORK RULE IF NOT EXISTS CONFLUENT_KAFKA_NR
    MODE = EGRESS
    TYPE = PRIVATE_HOST_PORT
    VALUE_LIST = (
        '*.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092',
        '*.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:9092',
        '*.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:9092' );

```

Note: if using Schema Registry on the cluster, ensure the rule allows TCP port 443 as well:

```

SQL
CREATE NETWORK RULE IF NOT EXISTS CONFLUENT_KAFKA_NR
    MODE = EGRESS
    TYPE = PRIVATE_HOST_PORT
    VALUE_LIST = (
        '*.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092',
        '*.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:9092',
        '*.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:9092',
        '*.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:443',

```

```
'*.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:443',  
'*.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:443');
```

When configuring Openflow, use the following URLs for the schema registry configuration (replace with your own schema registry endpoint, but **do** include the az1, az2, az3 in a comma-separated list):

None

```
https://lsrc-xxxxxx.az1.dom6pke3rmp.uksouth.azure.confluent.cloud,  
https://lsrc-xxxxxx.az2.dom6pke3rmp.uksouth.azure.confluent.cloud,  
https://lsrc-xxxxxx.az3.dom6pke3rmp.uksouth.azure.confluent.cloud
```

5. Create an External Access Integration

Execute the following command to create the EAI referencing your Network Rule (step #4)

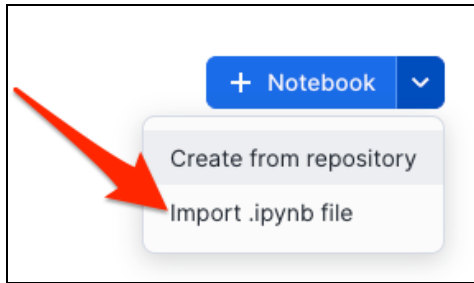
SQL

```
CREATE EXTERNAL ACCESS INTEGRATION IF NOT EXISTS CONFLUENT_KAFKA_EAI  
  ALLOWED_NETWORK_RULES = (CONFLUENT_KAFKA_NR)  
  ENABLED = true  
  COMMENT = 'External Access Integration for Confluent Kafka connectivity';
```

6. Download and install the Notebook to validate the configuration

Download the following Snowflake Notebook file from our Snowflake-Labs GitHub repository:

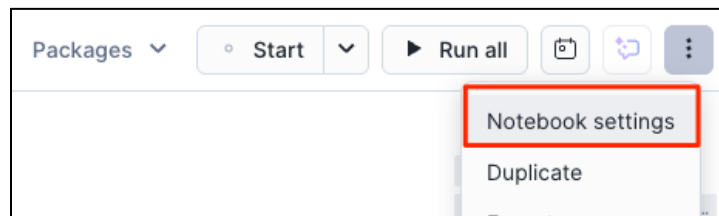
https://github.com/Snowflake-Labs/sfguide-getting-started-with-openflow-spcs/blob/main/notebooks/EAI_CONFLUENT_KAFKA_AZURE.ipynb



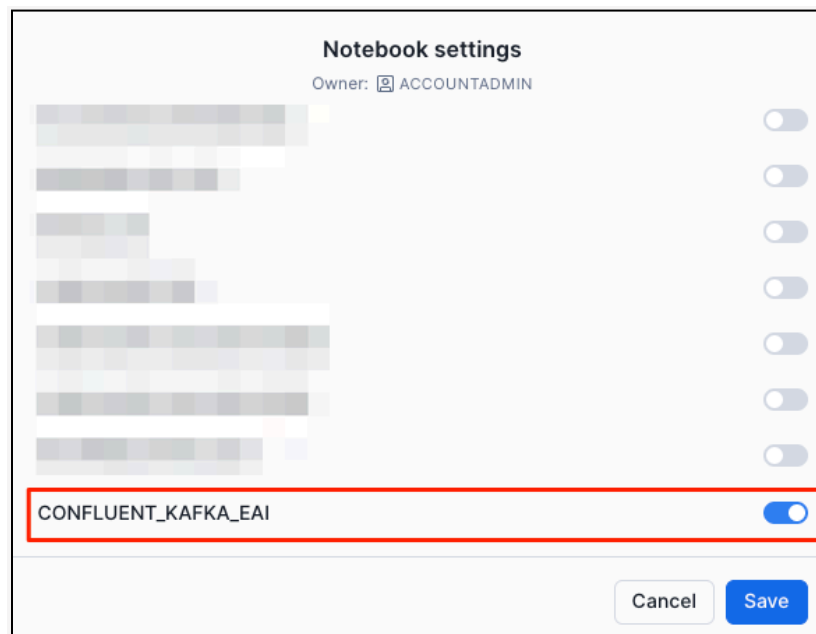
Import the file as a Notebook in Snowsight > Projects > Notebooks > **Import .ipynb file**

7. Enable the PostgreSQL EAI in the Notebook

Navigate to the **Notebook settings** at the top right corner of your screen



Under **External access**, enable the CONFLUENT_KAFKA_EAI Integration and Save



Wait for the Notebook to **restart**

8. Configure the variables and execute the Notebook

Python

```
KAFKA_BOOTSTRAP_SERVERS = [
    "lkc-kd6onp.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092",
    "lkc-kd6onp.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:9092",
    "lkc-kd6onp.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:9092"]
# 1x bootstrap per AZ

KAFKA_SASL_USERNAME = "FA2xxxxxTYR"
KAFKA_SASL_PASSWORD = "cf1xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx0GQ"
KAFKA_SASL_MECHANISM = "PLAIN" #DO NOT change
KAFKA_SECURITY_PROTOCOL = "SASL_SSL"
IMPLEMENTATION_ROLE = "ACCOUNTADMIN" #change accordingly
OPENFLOW_RUNTIME_ROLE = "OPENFLOW_ADMIN" #change accordingly
```

Connectivity test

```
=====
TEST 3a: SOCKET CONNECTIVITY
=====
✅ SUCCESS: Socket connection to all AZs established
=====
✅ lkc-kd6onp.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
✅ lkc-kd6onp.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
✅ lkc-kd6onp.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
=====
```

Authentication & metadata test

```
=====
TEST 3b: KAFKA PRODUCER & METADATA
=====
Connecting to Kafka cluster...
Using bootstrap server: lkc-kd6onp.az2.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
Fetching cluster metadata (timeout: 10s)...
✅ SUCCESS: Connected to Kafka cluster
Cluster ID: lkc-kd6onp
Number of brokers: 6
List of brokers: {0: BrokerMetadata(0, lkc-kd6onp-g000.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092)}
Number of topics: 7
=====
```

Creating new topics


```
Using bootstrap server: lkc-kd6onp.az1.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
🔧 Creating topic 'new_topic_test'...
✅ Topic 'new_topic_test' created successfully!
```

Reading messages in existing topics

```
Using bootstrap server: lkc-kd6onp.az3.dom6pke3rmp.uksouth.azure.confluent.cloud:9092
🔊 Listening for messages on topic 'topic1'...
🌿 Received message: {
  "TRADE_ID" : "f4a2c475-0a2b-484e-af4a-dc7015ac628a",
  "TICKER_SYMBOL" : "META",
  "TRADE_TIME" : "1747353899797",
  "TRADE_TYPE" : "BUY",
  "QUANTITY" : "487",
  "PRICE" : "81.00",
  "EXCHANGE" : "ARCA",
  "ACCOUNT_ID" : "ACC-73203",
  "ORDER_ID" : "1eb81aa6-4beb-4c7c-bfe2-e075a2359017",
  "COMMISSION" : "1.00",
  "METADATA_ACTION" : "INSERT",
  "METADATA_ISUPDATE" : false,
  "METADATA_ROW_ID" : "0d0b2b523e764405f616f4a7473596e0d2890026"
} (partition 5)
🌿 Received message: {
  "TRADE_ID" : "6ffa8763-3d65-4d81-aa60-d477dbfc95a8",
  "TICKER_SYMBOL" : "NVDA",
  "TRADE_TIME" : "1742783233797",
```