DEBUGGING AND TROUBLESHOOTING KAFKA CONNECT

Đơn vị: Công ty CP Giáo dục và Công nghệ QNET

Quality Network for Education and Technology

QNET JOINT STOCK COMPANY

Address: 14th Floor, VTC Online Tower
18 Tam Trinh Street. Hoang Mai District
Hanoi, Vietnam

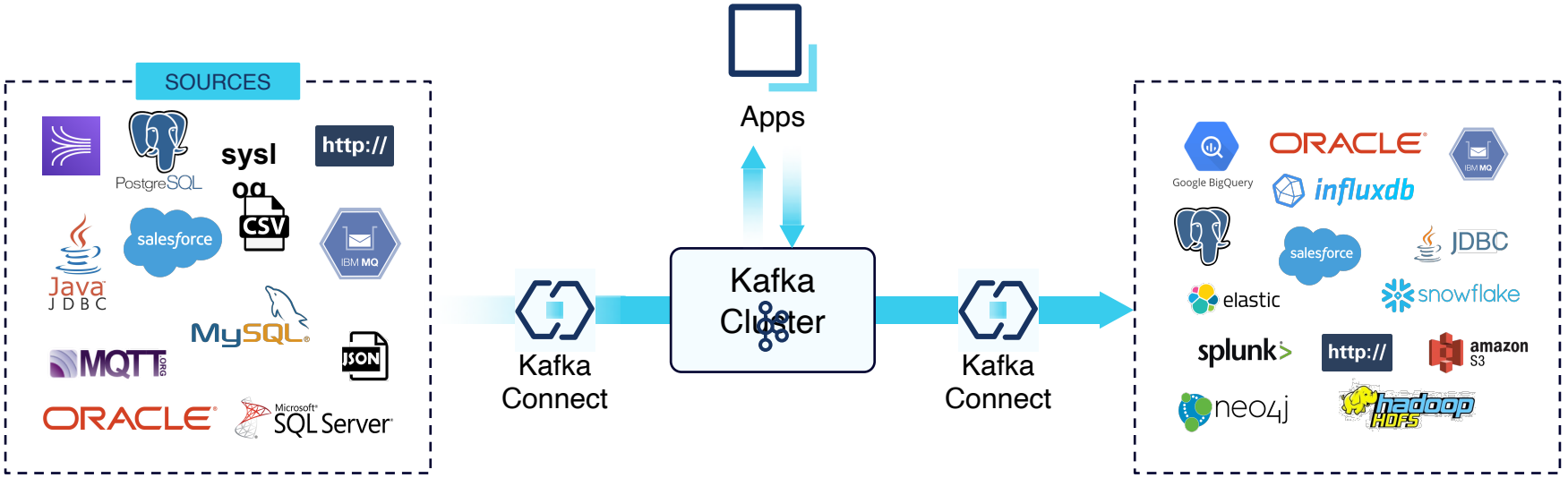
KAFKA CONNECT

Learning Objects

1. Introduction to Kafka Connect
2. Running Kafka Connect
3. Connectors, Configuration, Converters, and Transforms
4. Deploy Kafka Connect
5. Hands On: Run a self-managed Connector in Docker
6. Kafka Connect's REST API
7. Errors and Dead Letter Queues

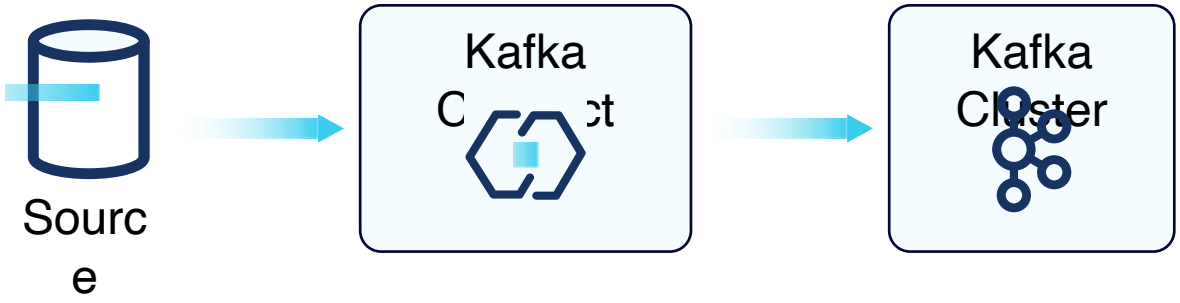
KAFKA CONNECT

Ingest Data from Upstream Systems



KAFKA CONNECT

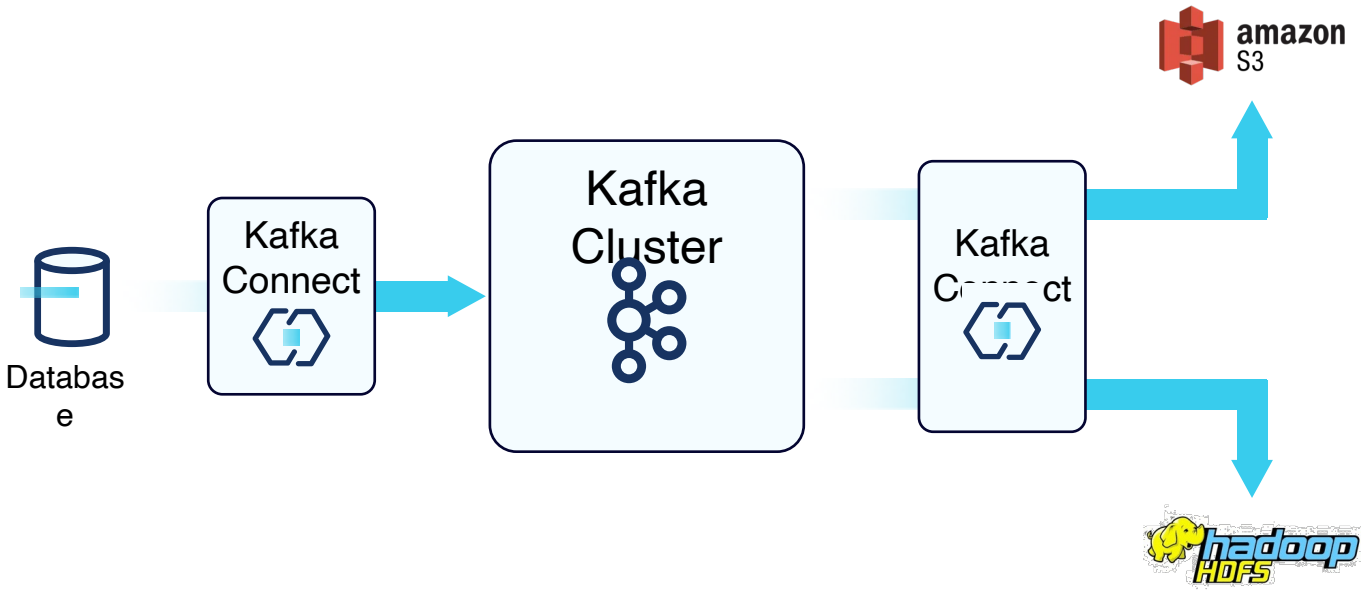
How Kafka Connect Works




```
{
        "connector.class":
                "io.confluent.connect.jdbc.JdbcSourceConnector
                ",
        "connection.url":
                "jdbc:mysql://asgard:3306/demo",
        "table.whitelist":
                "sales,orders,customers"
}
```

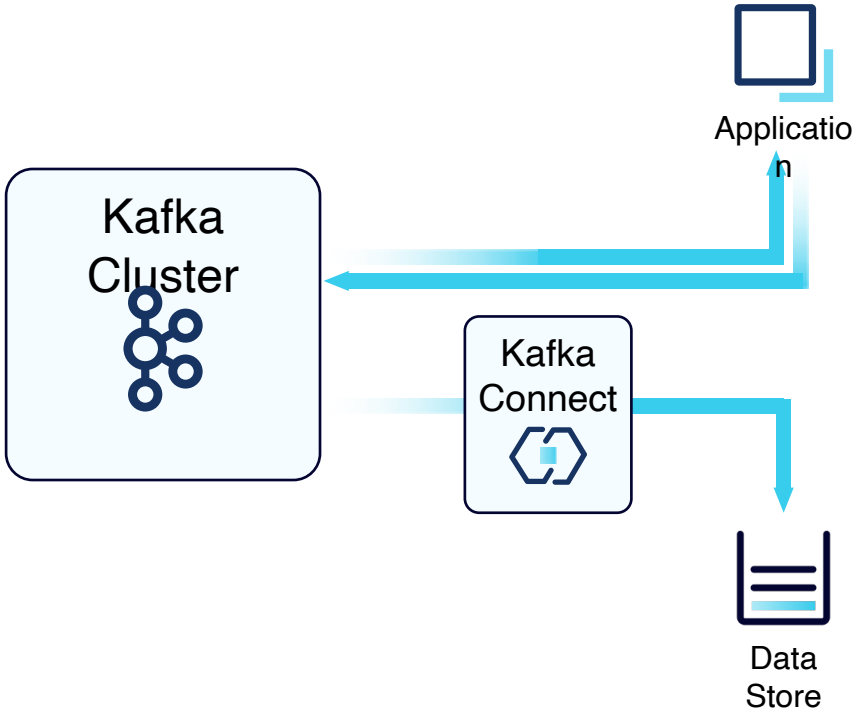
KAFKA CONNECT

Streaming Pipelines



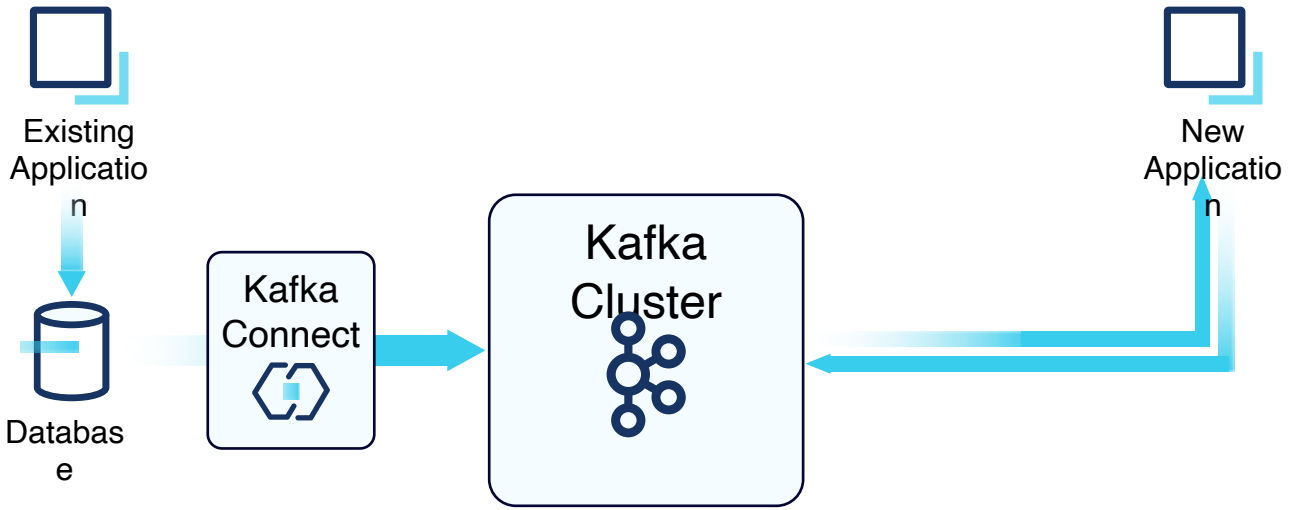
KAFKA CONNECT

Writing to Datastores from Kafka



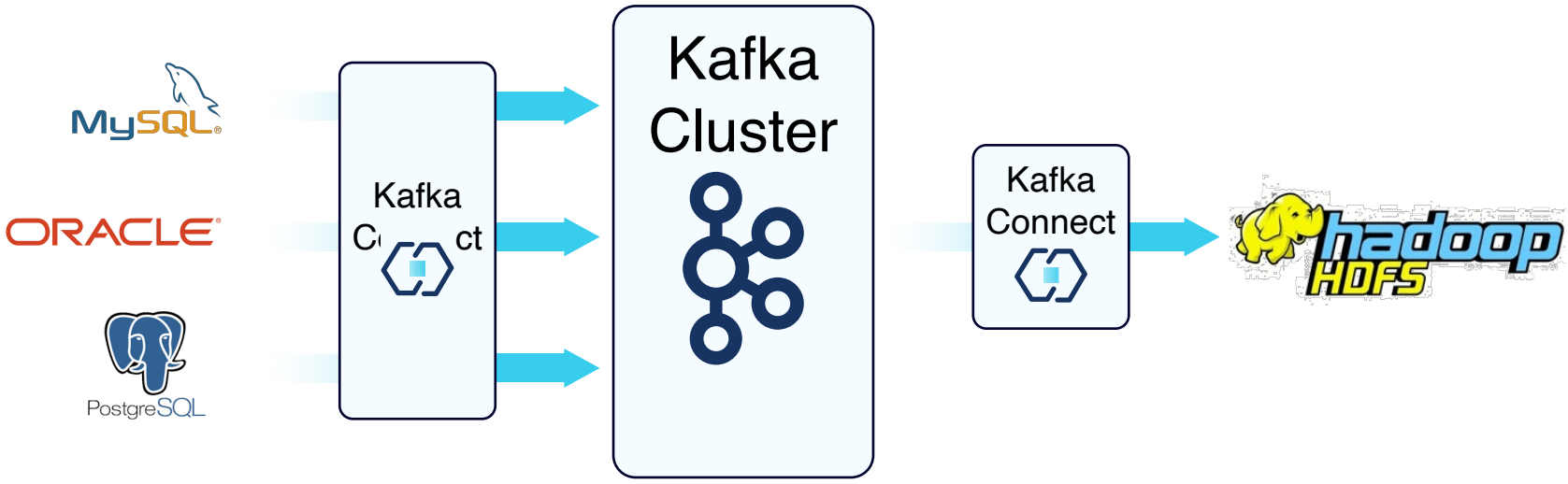
KAFKA CONNECT

Evolve Processing from Old Systems to New



KAFKA CONNECT

Make Systems Real Time



KAFKA CONNECT

Why Not Write Your Own Intergrations ?

Certainly possible using the Apache Kafka producer and consumer APIs

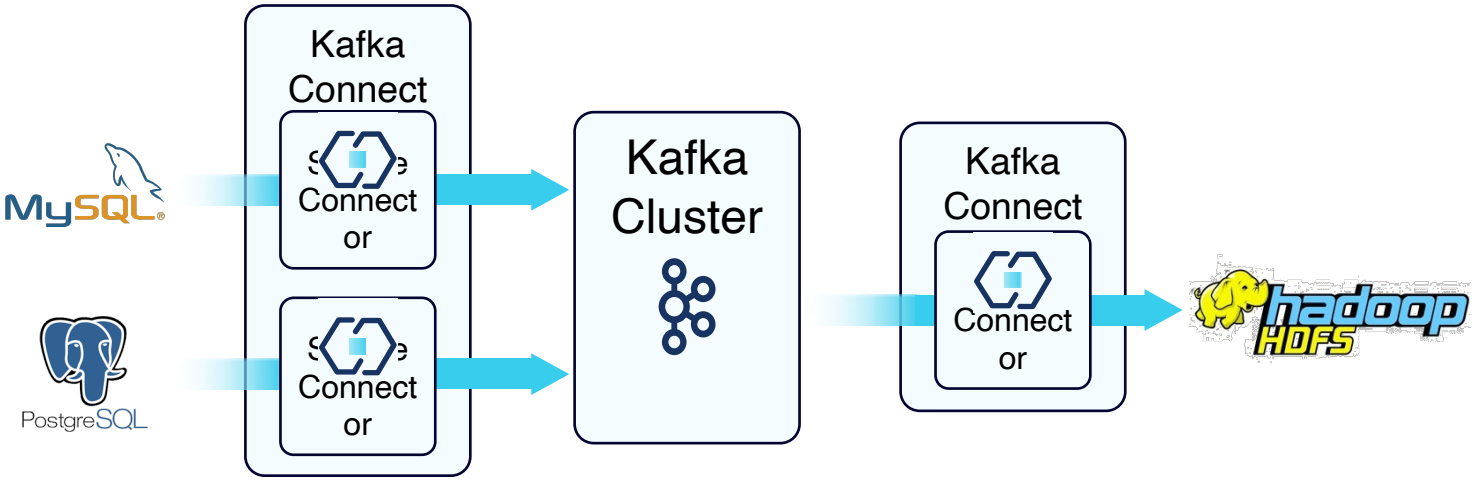
- Not so simple though when you consider:
  - Handling failures and restarts
  - Logging
  - Scaling up and down to meet varying data loads
  - Running across multiple nodes
  - Serialization and data formats
- Once written, this now complex application needs to be maintained and updated to changes in Kafka as well as the external data sources and targets

  Kafka Connect solves all of these problems
- In most cases, it should be used when data needs to be integrated with Kafka

RUNNING KAFKA CONNECT

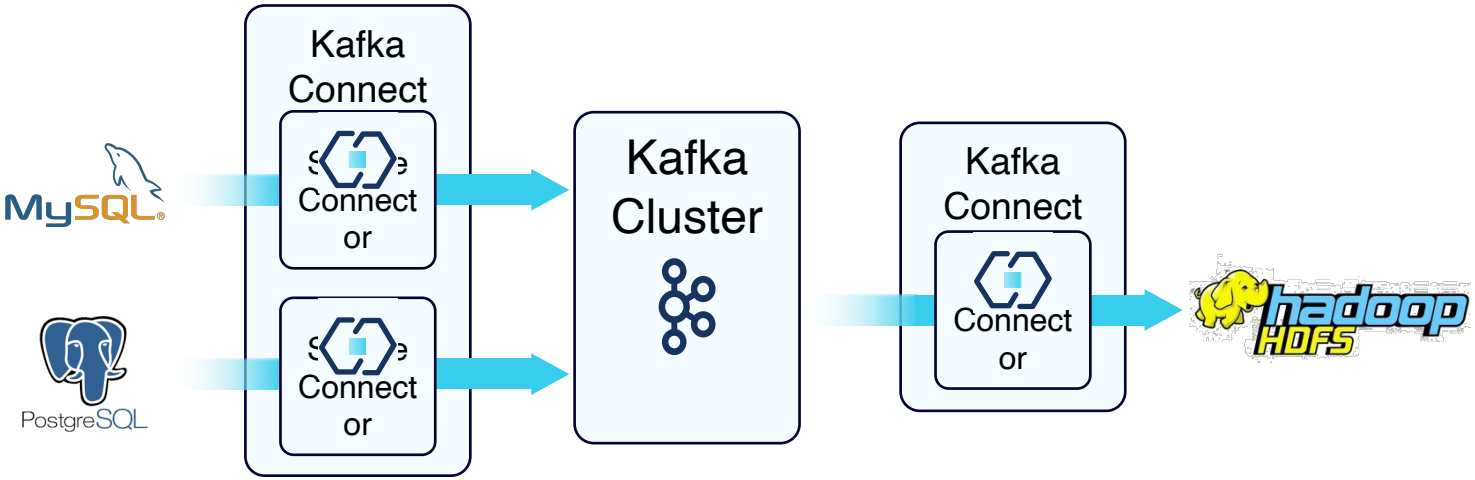
KAFKA CONNECT

Connectors



KAFKA CONNECT

Connectors

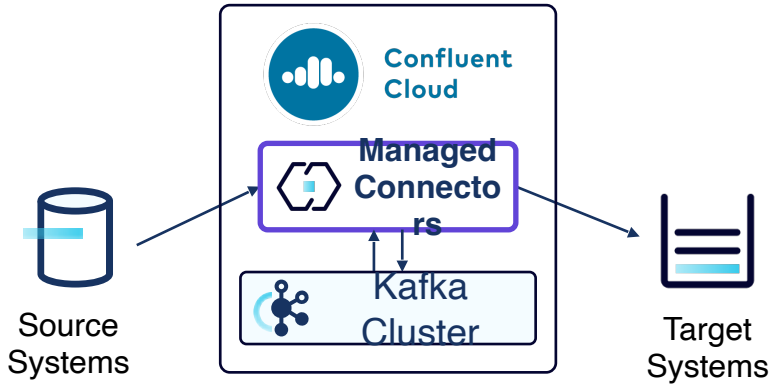


KAFKA CONNECT

Confluent Cloud Managed Connectors

Confluent Cloud provides fully managed connectors

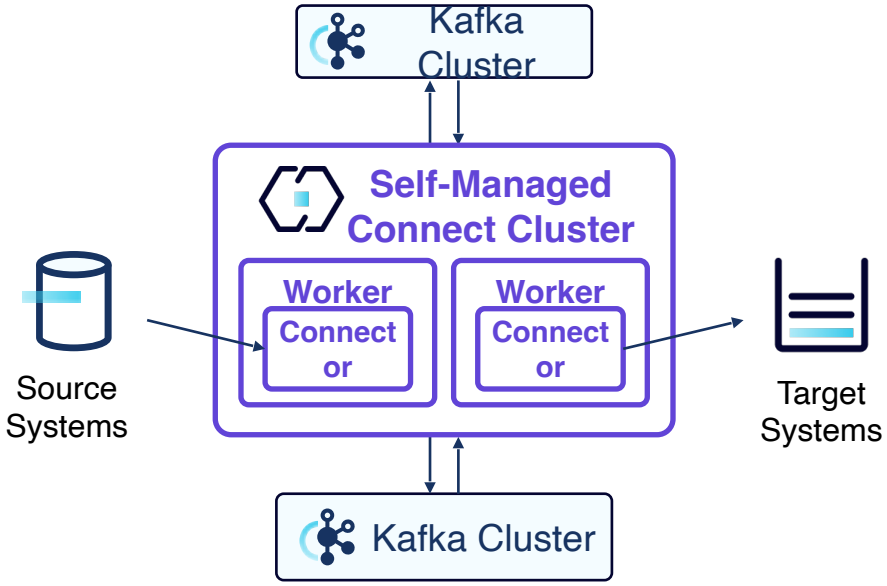
- You just select the connector and identify details regarding the source/target system
  - GUI/CLI/API to create and manage connector instances
- Confluent takes care of the rest on your behalf
  - Provisioning, execution, failures, and so on
- Managed connector limitations
  - Some connectors are not yet available
  - Some connector transformations are not yet available
  - Some config settings may not be available
  - Network connectivity requirements



KAFKA CONNECT

Self-Managed Kafka Connect

- Self-managed Kafka Connect consists of one or more Connect clusters depending upon the requirement
- Each Connect cluster consists of one or more Connect worker(s)
    - Connector instances run on Connect workers

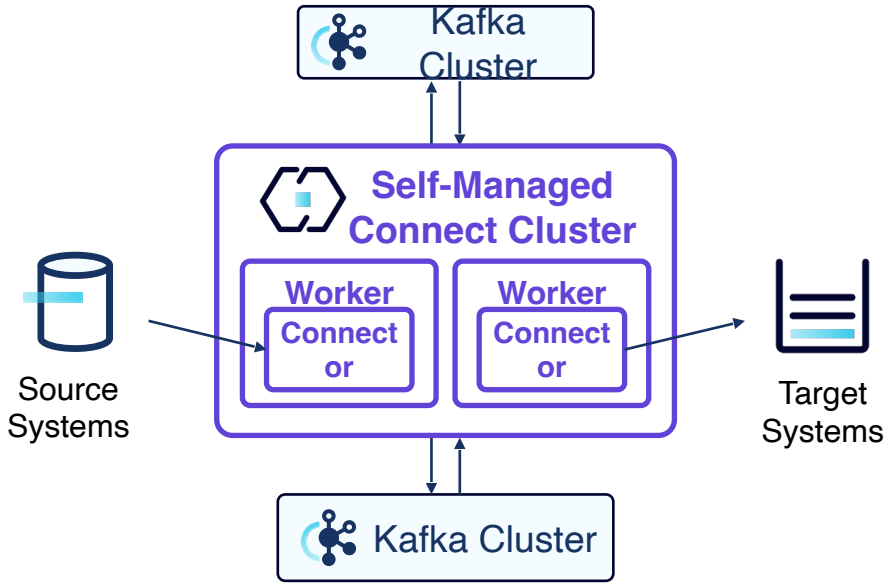


KAFKA CONNECT

Kafka Connect Workers

Kafka Connect workers are JVM processes

- Can be deployed on bare metal or containers, e.g.
  - Bare-metal on-premises
  - IaaS Compute (AWS EC2, Google Compute Engine, etc)
  - Docker
    - On-premises
    - Cloud-based

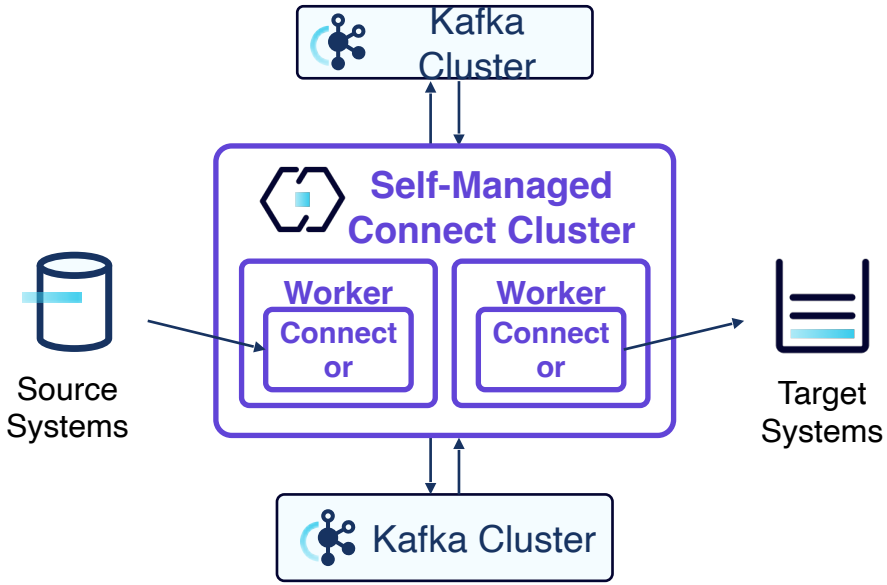


KAFKA CONNECT

Managing a Kafka Connect Cluster

Management responsibilities include:

- Worker configuration
- Scaling the Connect cluster up/ down to suit demand changes
- Monitoring for problems
  - Troubleshooting
  - Corrective actions

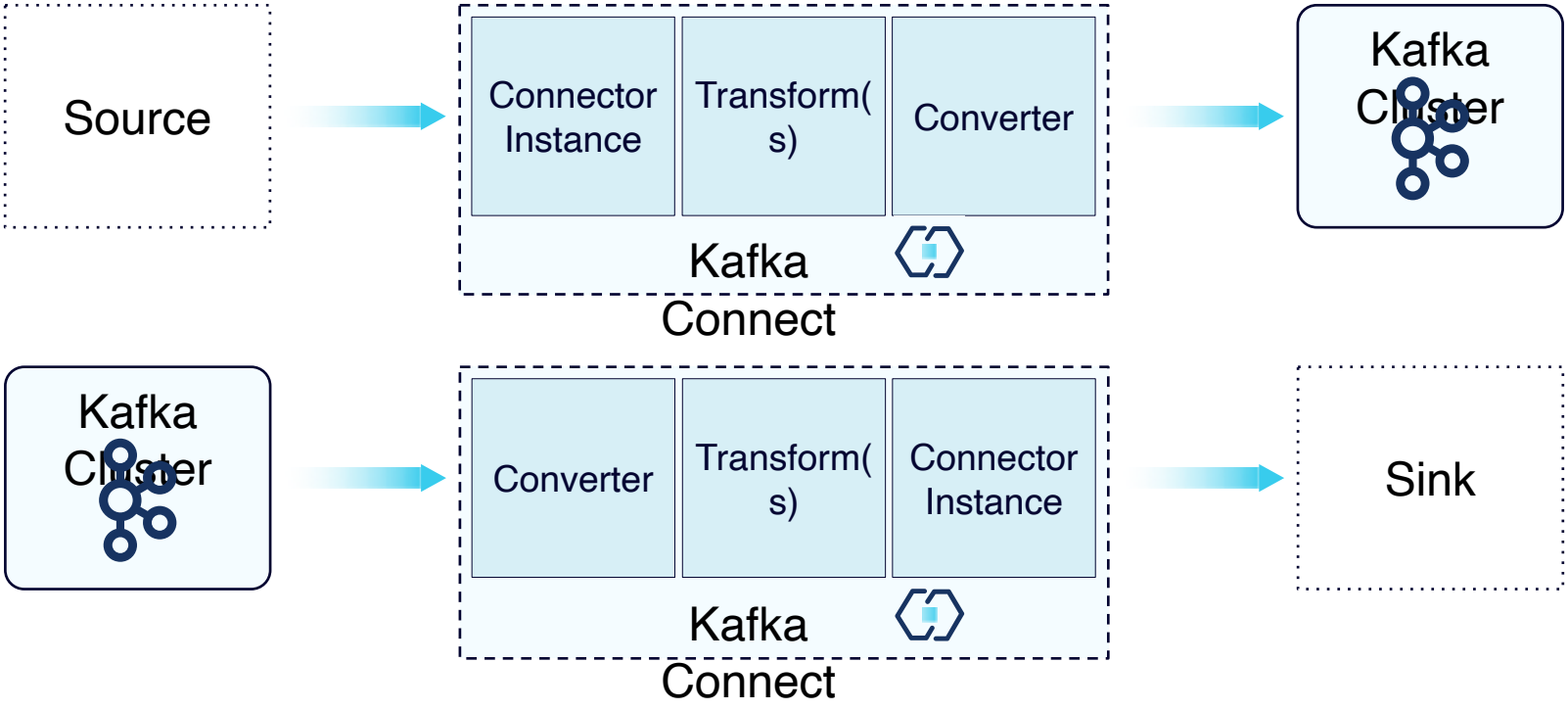


KAFKA CONNECT

Connectors, Configurations, Converters and Transforms

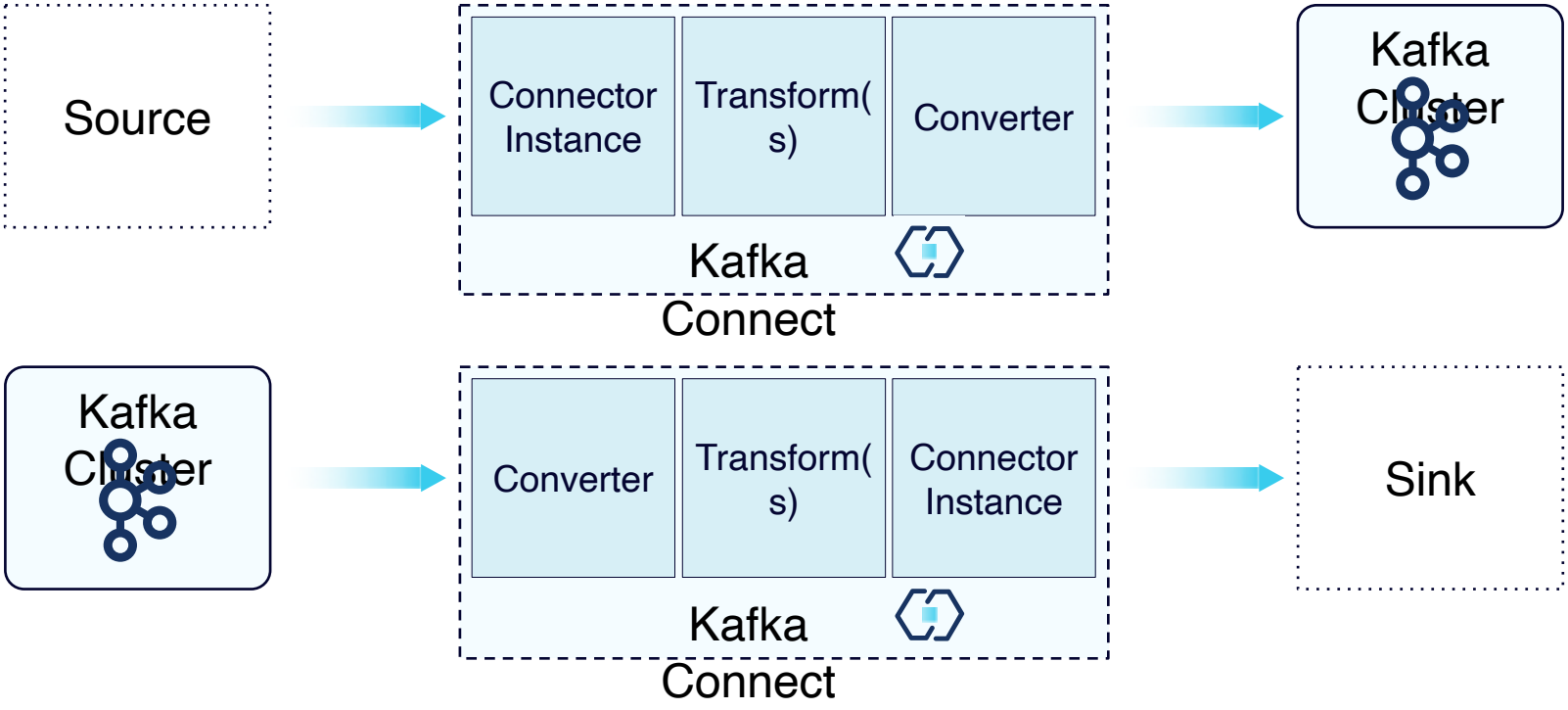
KAFKA CONNECT

Inside Kafka Connect



KAFKA CONNECT

Inside Kafka Connect



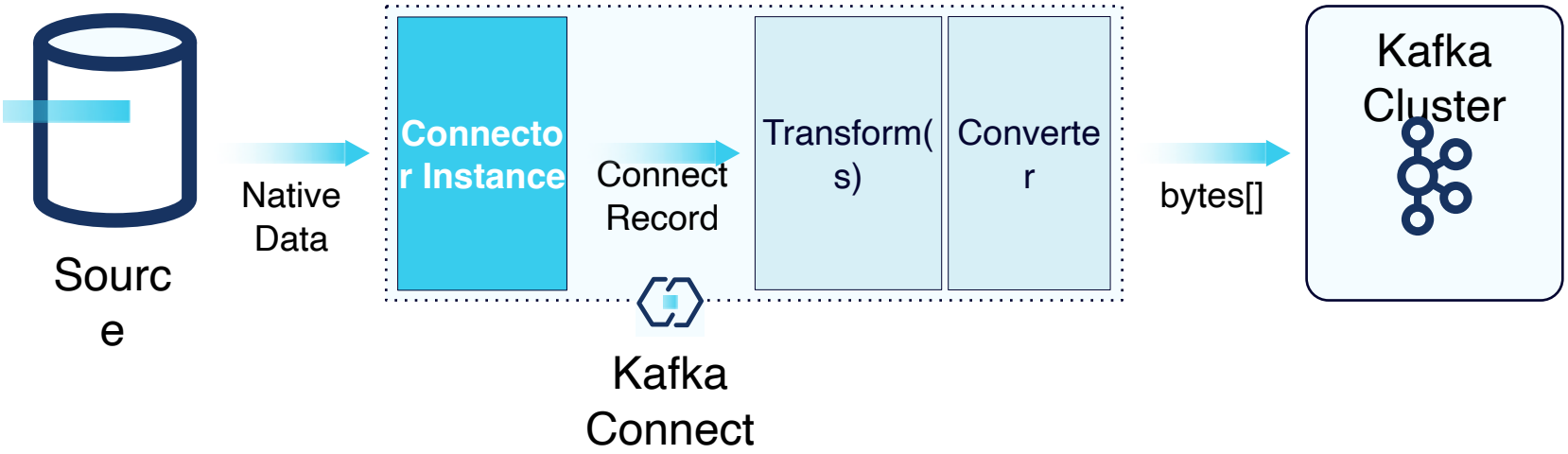
KAFKA CONNECT

Add a Connector Instance with the REST API

```
curl -X PUT -H  "Content-Type:application/json" http://localhost:8083/
connectors/sink-elastic-01/config \
    -d '{
"connector.class":
"io.confluent.connect.elasticsearch.ElasticsearchSinkConnector",
    "topics"          : "orders",
    "connection.url" : "http://elasticsearch:9200",
    "type.name"       : "_doc",
    "key.ignore"      : "false",
    "schema.ignore"   : "true"
}'
```

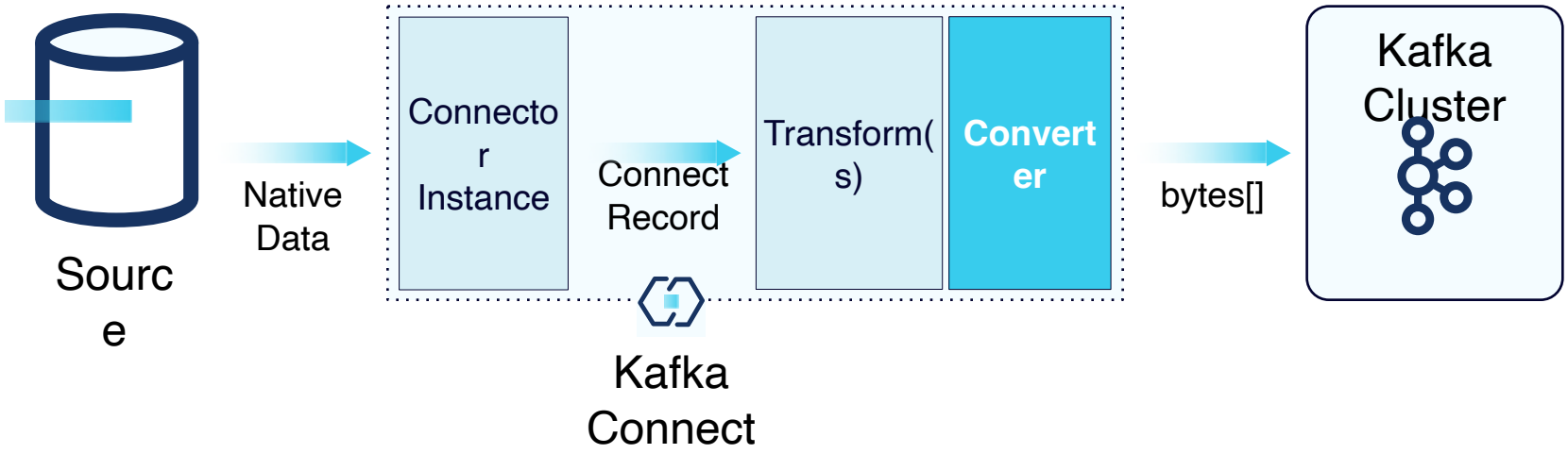
KAFKA CONNECT

Add a Connector Instance with the REST API



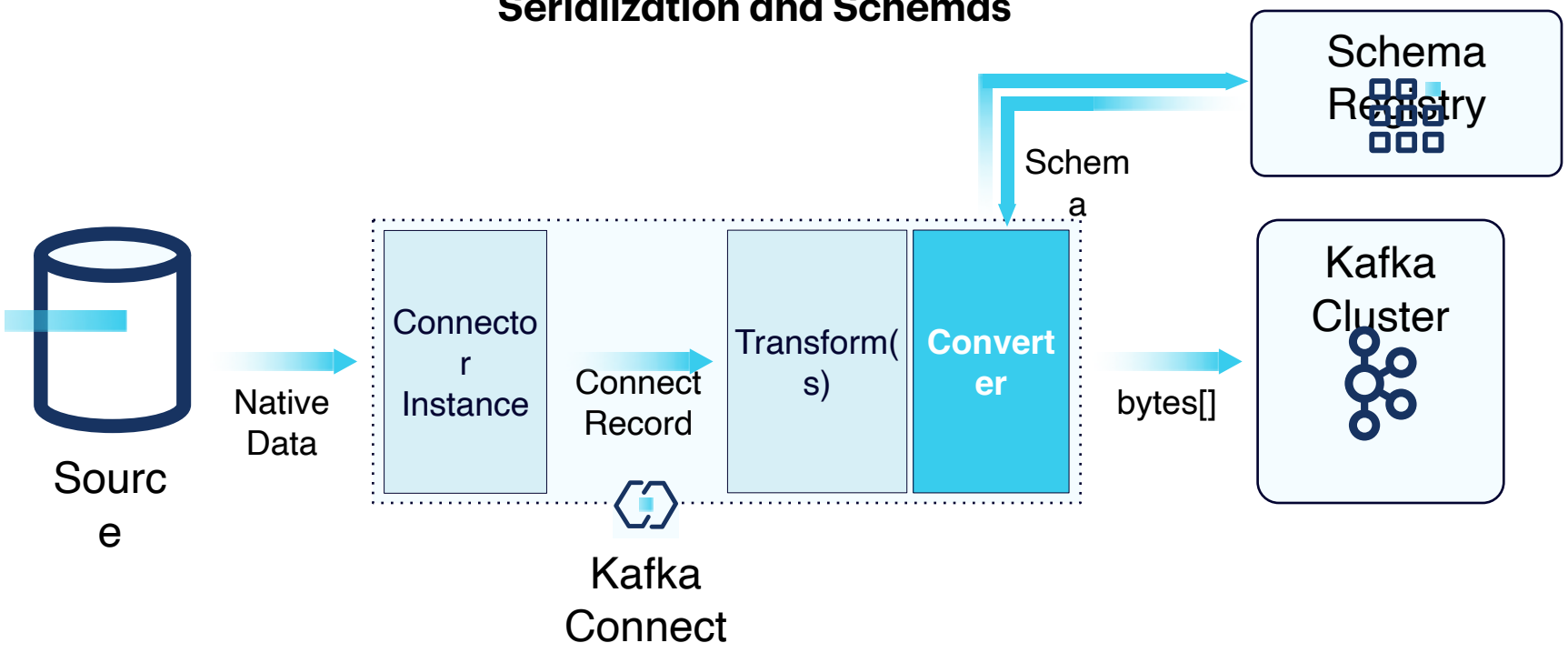
KAFKA CONNECT

Converters Serialize/Deserializer the Data



KAFKA CONNECT

Serialization and Schemas



KAFKA CONNECT

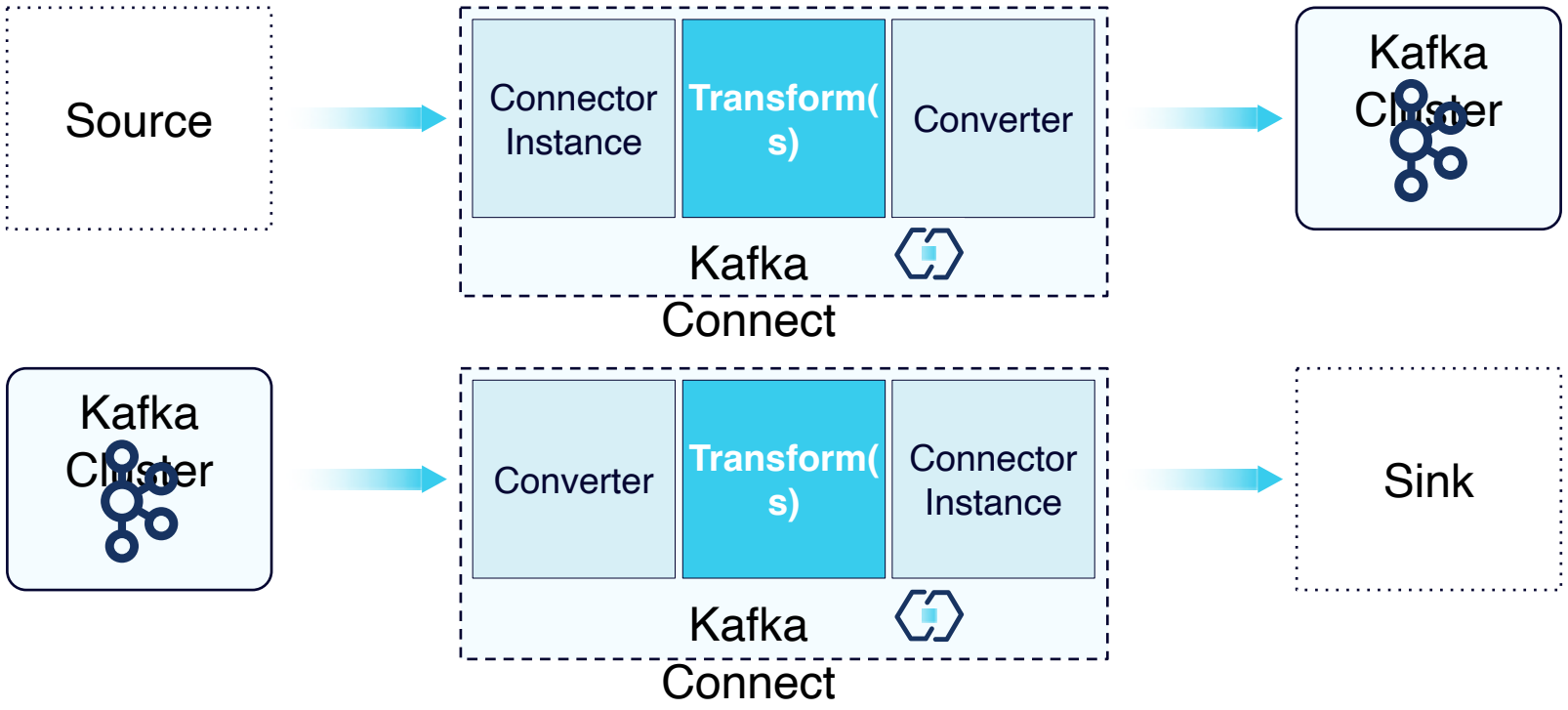
Converters Specified for Key and Value

```
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.storage.StringConverter
value.converter.schema.registry.url=http://localhost:8081
```

Set as a global default per worker; optionally can be overridden per connector instance

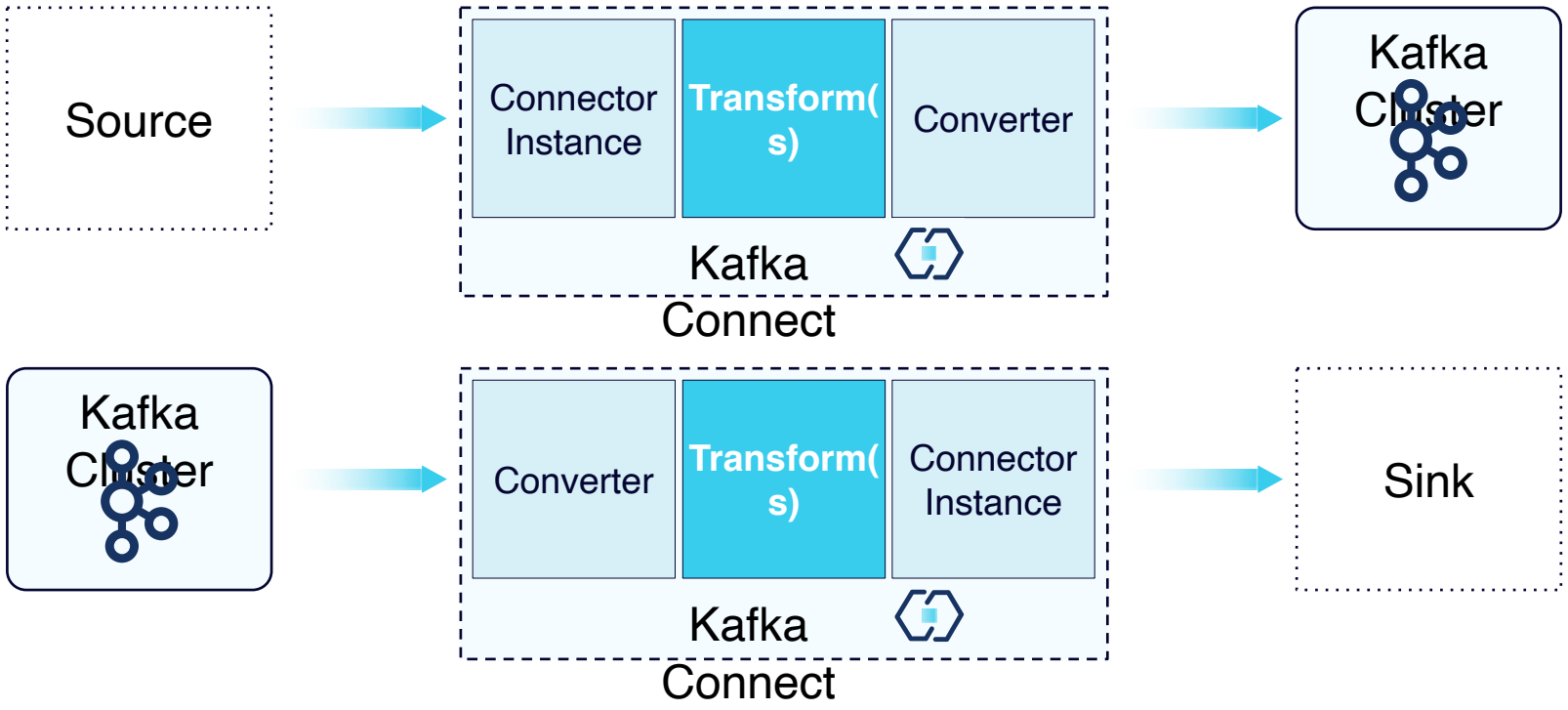
KAFKA CONNECT

Single Message Transforms



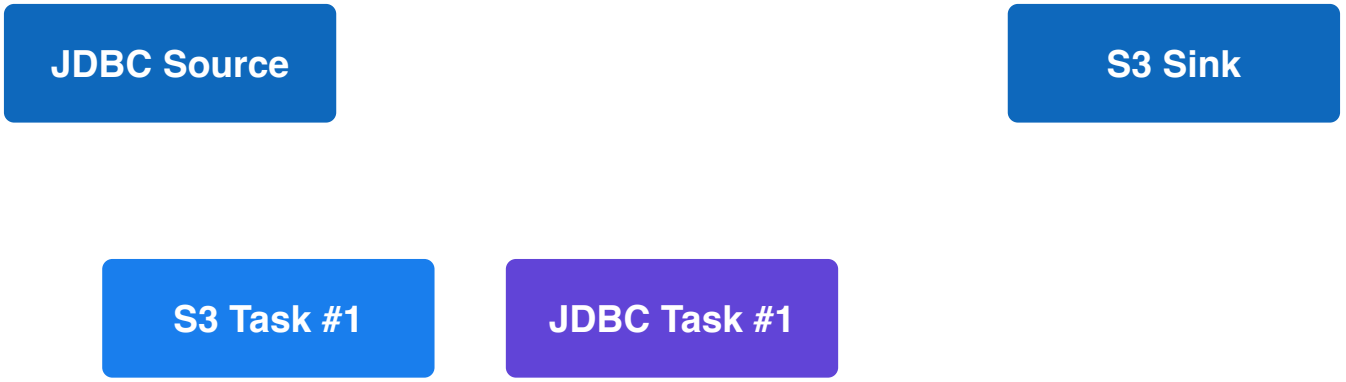
KAFKA CONNECT

Single Message Transforms



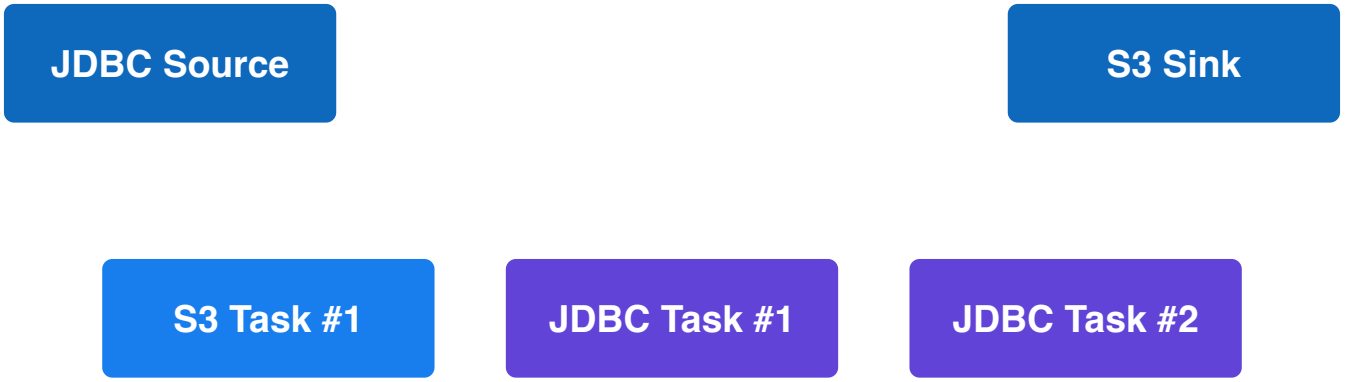
KAFKA CONNECT

Deploying Kafka Connect



KAFKA CONNECT

Tasks are the Unit of Parrallelism and Scale

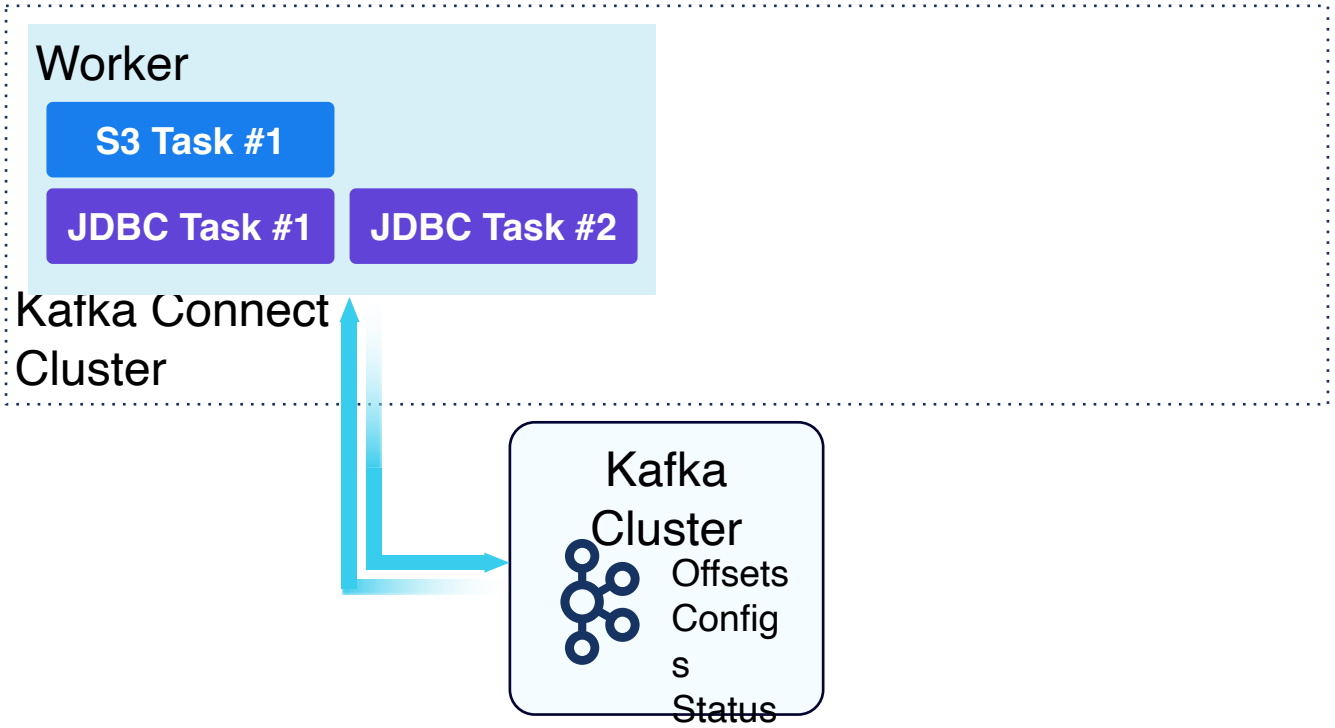


KAFKA CONNECT

Connect Worker

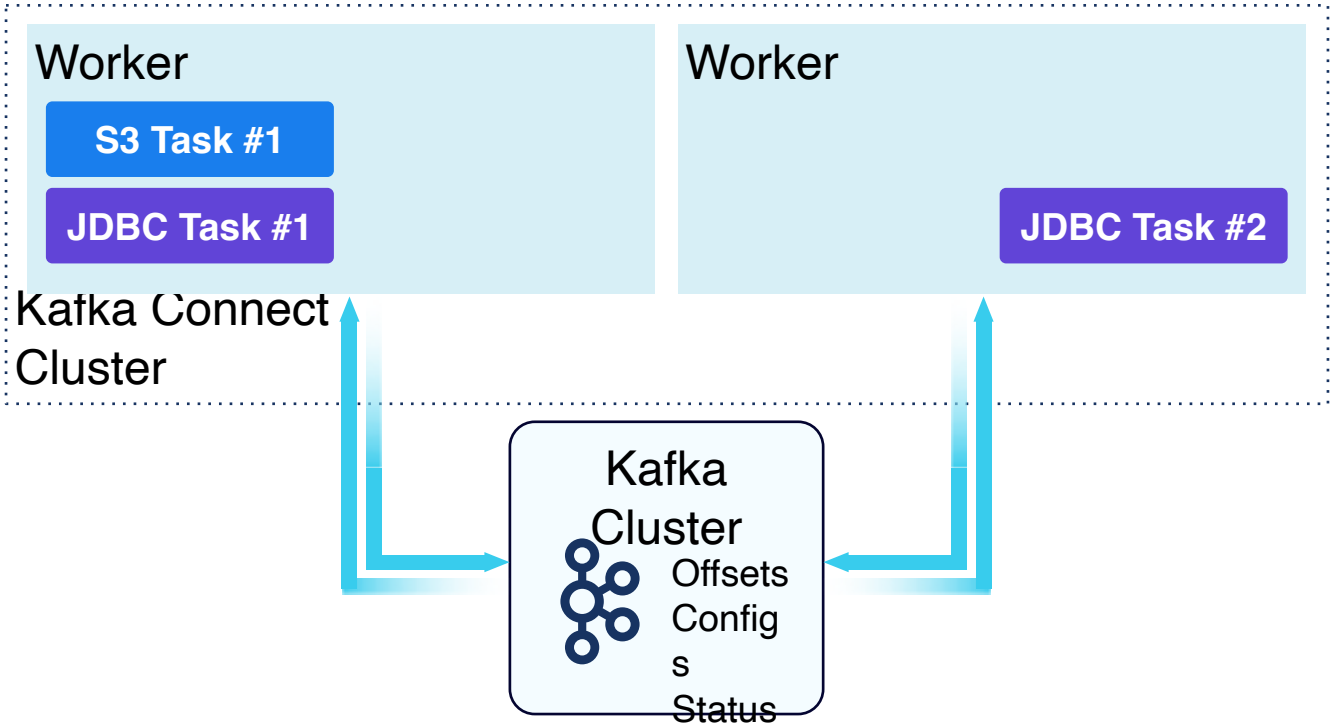
KAFKA CONNECT

Kafka Connect Distributed Mode



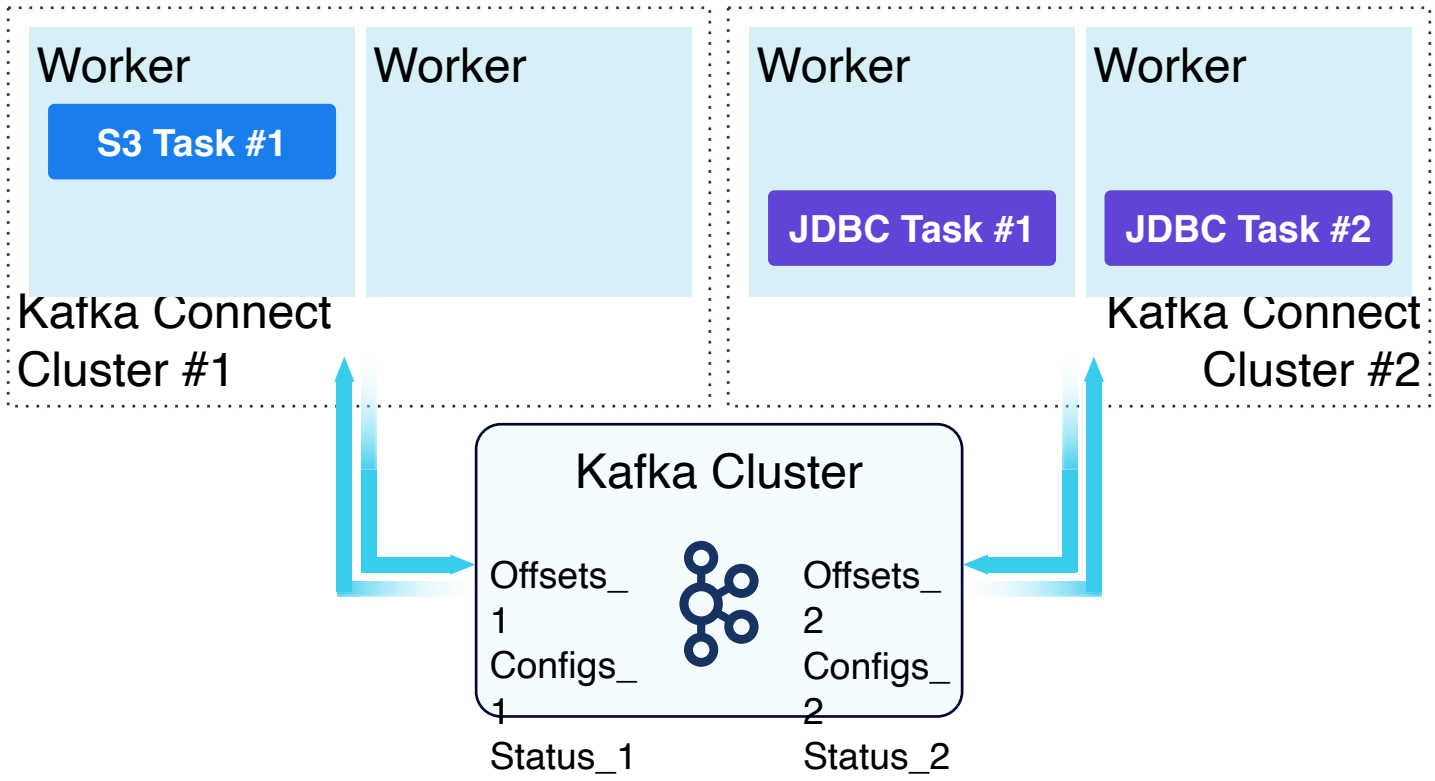
KAFKA CONNECT

Kafka Connect Scalability



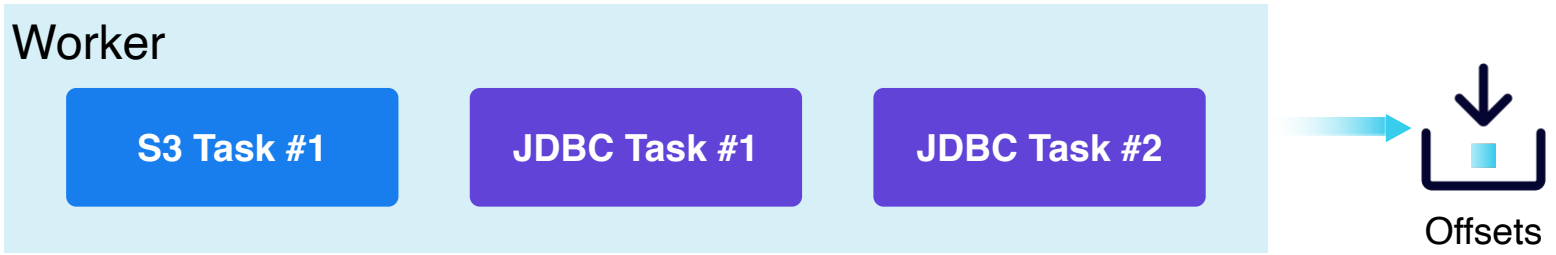
KAFKA CONNECT

Multiple Workers vs Multiple Clusters



KAFKA CONNECT

Multiple Workers vs Multiple Clusters



KAFKA CONNECT

Running Kafka Connect in Docker

KAFKA CONNECT

Kafka Connect Rest API

Get basic Connect Information:

```
curl http://localhost:8083/
```

Listing Installed Plugins:

```
curl -s localhost:8083/connector-plugins
```

KAFKA CONNECT

Kafka Connect Rest API

Formatting the Result of the Installed Plugin List

```
curl "http://localhost:8083/
```

KAFKA CONNECT

Cre

Rest API

```
curl -i -X PUT -H  "Content-Type:application/json" \
    http://localhost:8083/connectors/source-debezium-orders-00/config \
    -d '{
            "connector.class": "io.debezium.connector.mysql.MySqlConnector",
            "value.converter": "io.confluent.connect.json.JsonSchemaConverter",
            "value.converter.schemas.enable": "true",
            "value.converter.schema.registry.url": "'$SCHEMA_REGISTRY_URL'",
            "value.converter.basic.auth.credentials.source": "'$BASIC_AUTH_CREDENTIALS_SOURCE'",
            "value.converter.basic.auth.user.info": "'$SCHEMA_REGISTRY_BASIC_AUTH_USER_INFO'",
            "database.hostname": "mysql",
            "database.port": "3306",
            "database.user": "debezium",
            "database.password": "dbz",
            "database.server.id": "42",
            "database.server.name": "asgard",
            "table.whitelist": "demo.orders",
            "database.history.kafka.bootstrap.servers": "'$BOOTSTRAP_SERVERS'",
            "database.history.consumer.security.protocol": "SASL_SSL",
            "database.history.consumer.sasl.mechanism": "PLAIN",
            "database.history.consumer.sasl.jaas.config": "org.apache.kafka.common.security.plain.PlainLoginModule required username=\"'$CLOUD_KEY'\"
password=\"'$CLOUD_SECRET'\";",
            "database.history.producer.security.protocol": "SASL_SSL",
            "database.history.producer.sasl.mechanism": "PLAIN",
            "database.history.producer.sasl.jaas.config": "org.apache.kafka.common.security.plain.PlainLoginModule required username=\"'$CLOUD_KEY'\"
password=\"'$CLOUD_SECRET'\";",
            "database.history.kafka.topic": "dbhistory.demo",
            "topic.creation.default.replication.factor": "3",
            "topic.creation.default.partitions": "3",
            "decimal.handling.mode": "double",
            "include.schema.changes": "true",
            "transforms": "unwrap,addTopicPrefix",
            "transforms.unwrap.type": "io.debezium.transforms.ExtractNewRecordState",
            "transforms.addTopicPrefix.type":"org.apache.kafka.connect.transforms.RegexRouter",
            "transforms.addTopicPrefix.regex":"(.*)",
            "transforms.addTopicPrefix.replacement":"mysql-debezium-$1"
    }'
```

KAFKA CONNECT

Kafka Connect Rest API

Listing Connector Instances:

```
curl -s -X GET "http://localhost:8083/connectors
```

KAFKA CONNECT

Kafka Connect Rest API

Inspect Config and Status for a

```
curl -i -X GET -H  "Content-Type:application/json" \
        http://localhost:8083/connectors/sink-elastic-orders-00/config
```

KAFKA CONNECT

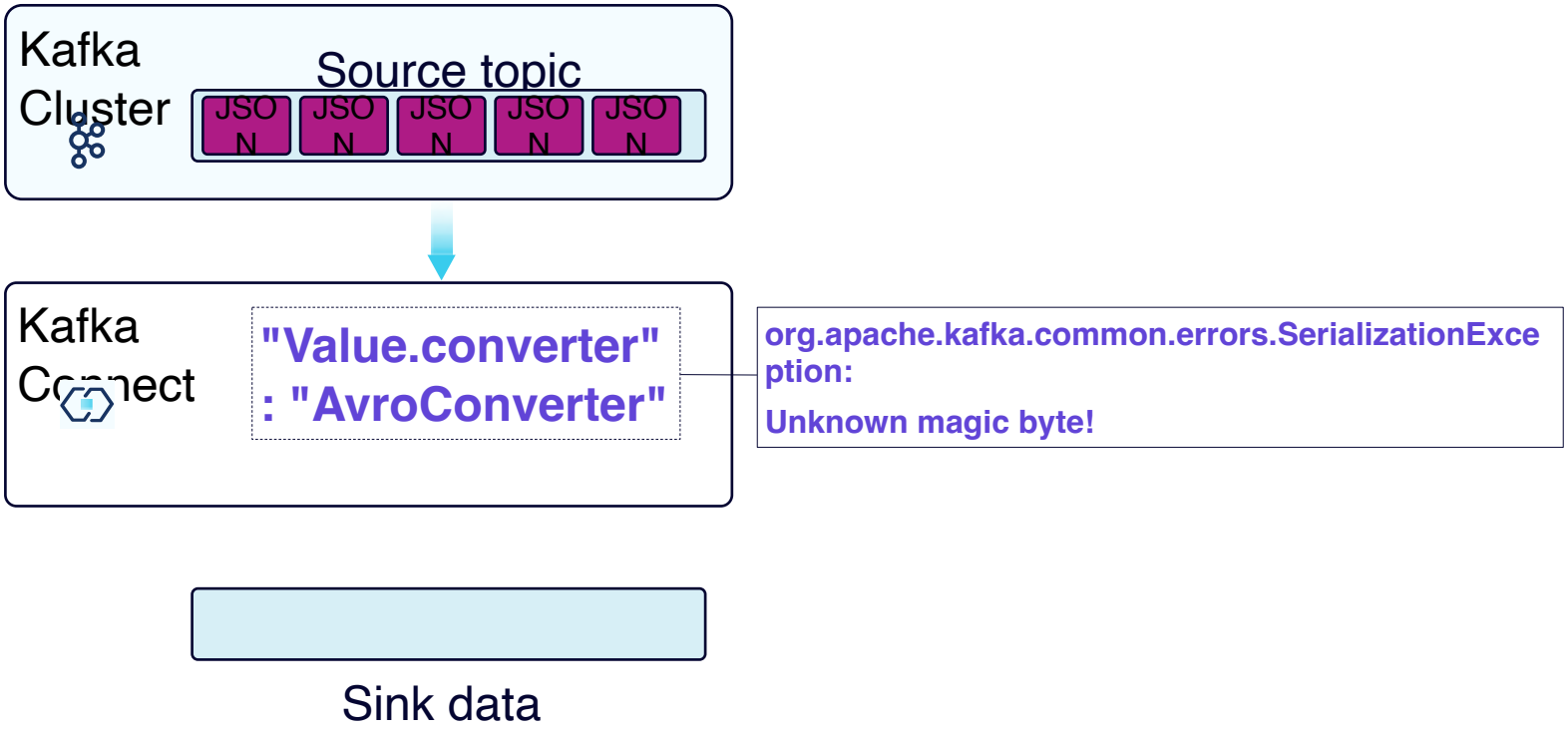
Kafka Connect Rest API

Delete a Connector:

```
curl -s -X DELETE "http://localhost:8083/
connectors/sink-elastic-orders-00"
```

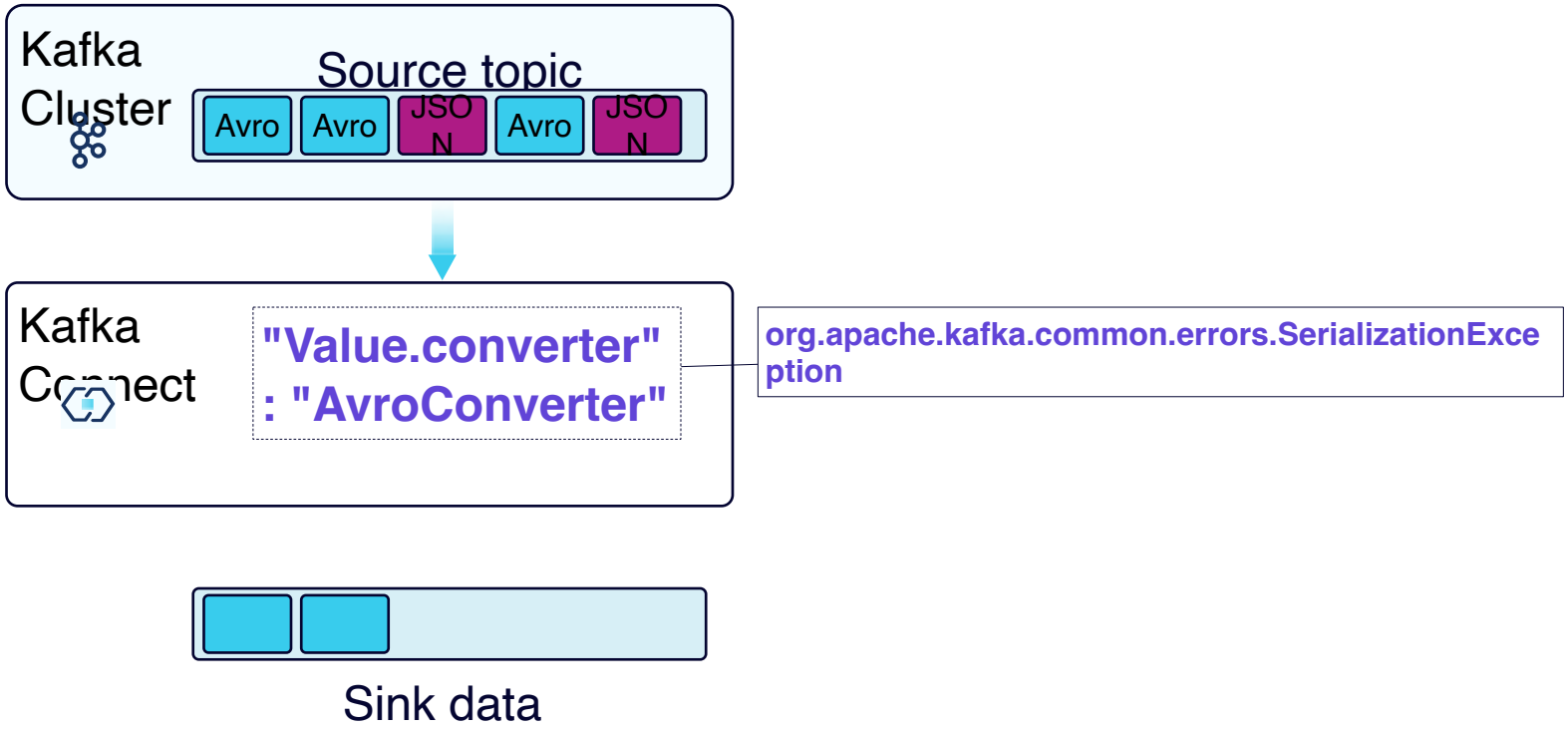
KAFKA CONNECT

Serialization - Wrong Converter



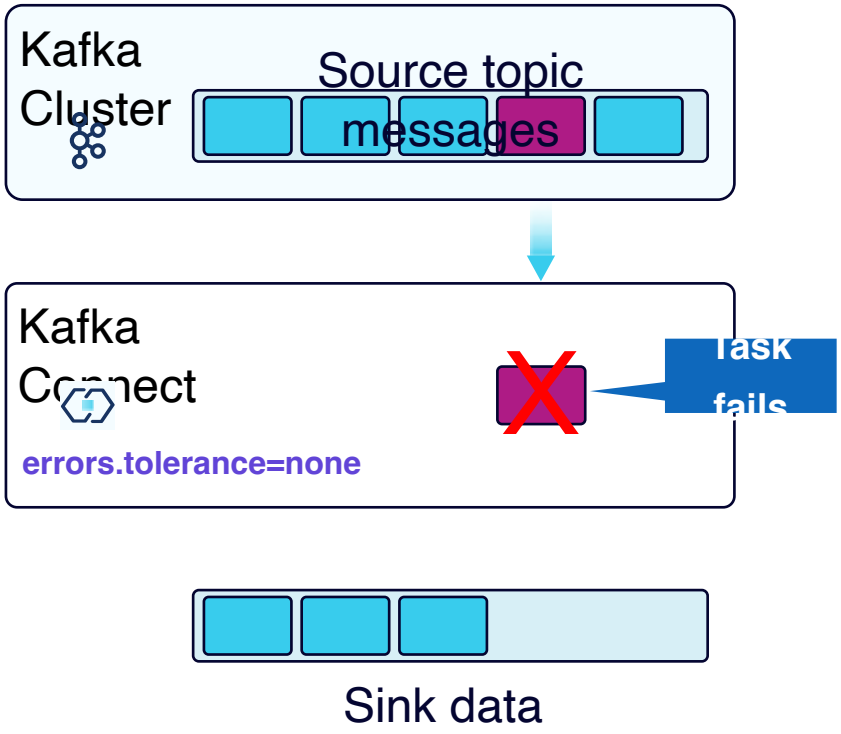
KAFKA CONNECT

Serialization - Multiple format



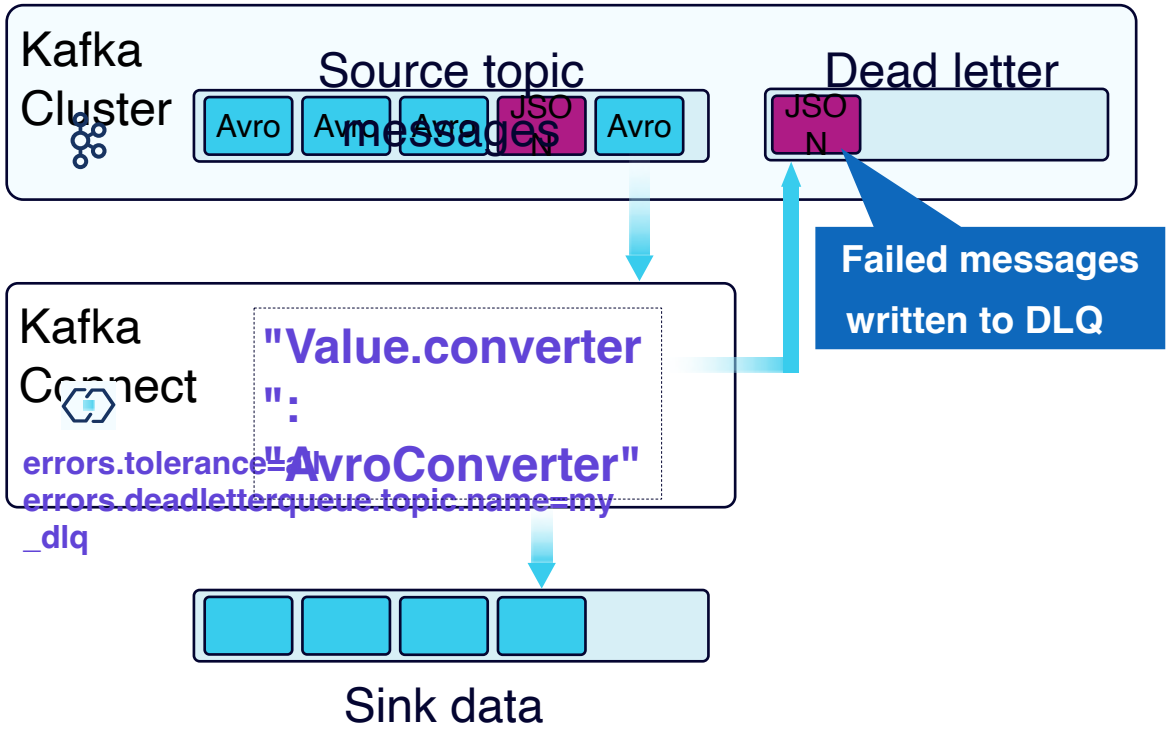
KAFKA CONNECT

Error Tolerances - Fail Fast



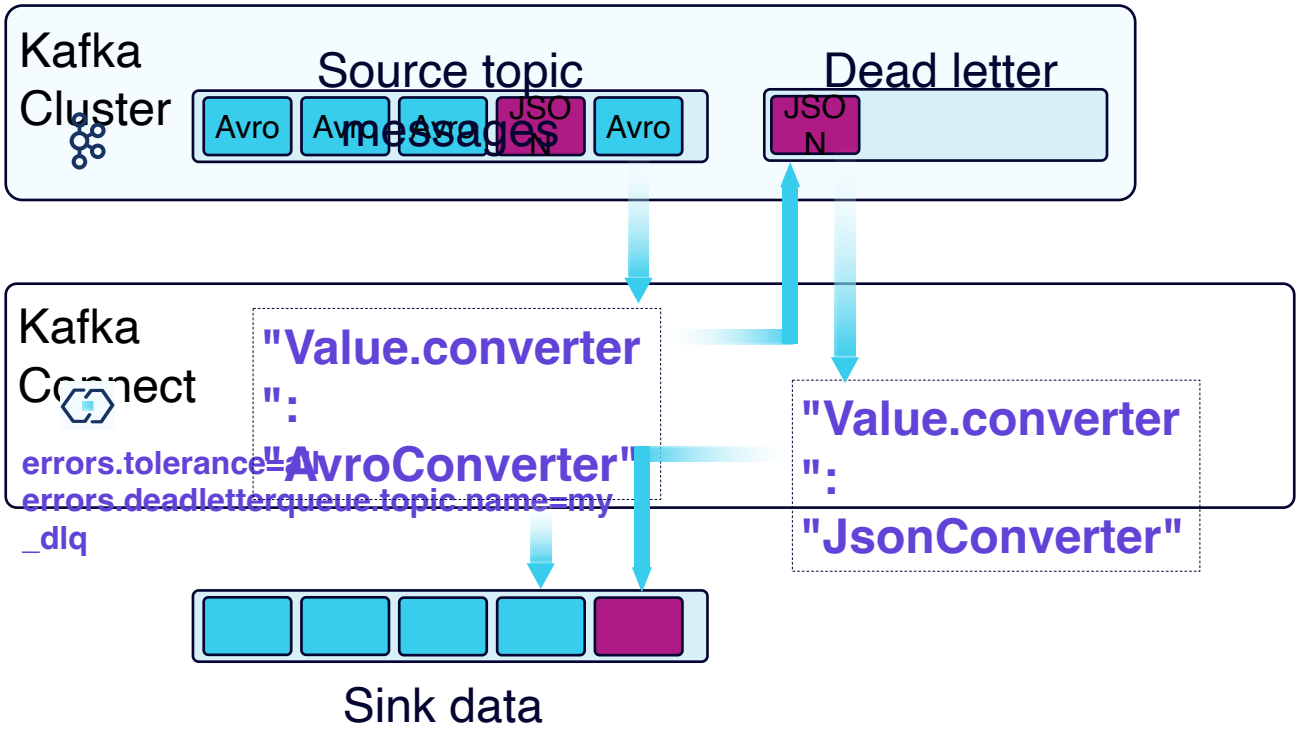
KAFKA CONNECT

Error Tolerances - Dead Letter Queue



KAFKA CONNECT

Reprocessing the Dead Letter Queue

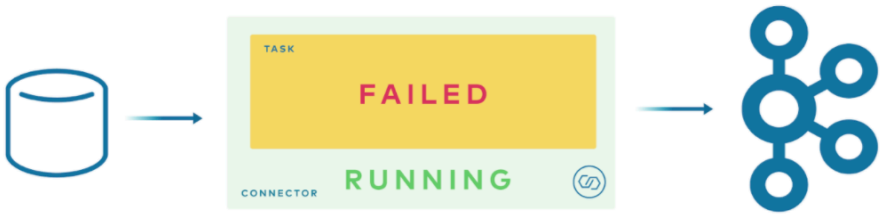


KAFKA CONNECT

Troubleshooting Kafka Connect

Your Connect worker is running, your source connector is running

=>**but no data is being ingested.**



KAFKA CONNECT

Troubleshooting Kafka Connect

Getting task status:

```
curl -s "http://localhost:8083/connectors/source-debezium-orders-00/status"
| jq '.tasks[0].trace'

"org.apache.kafka.connect.errors.ConnectException\n\tat
io.debezium.connector.mysql.AbstractReader.wrap(AbstractReader.java:230)\n\tat
io.debezium.connector.mysql.AbstractReader.failed(AbstractReader.java:197)\n\tat
io.debezium.connector.mysql.BinlogReader$ReaderThreadLifecycleListener.onCommunicationF
ailure(BinlogReader.java:1018)\n\t
at
com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.ja
va:950)\n\tat
com.github.shyiko.mysql.binlog.BinaryLogClient.connect(BinaryLogClient.java:580)\n\tat
com.github.shyiko.mysql.binlog.BinaryLogClient$7.run(BinaryLogClient.java:825)\n\tat
java.lang.Thread.run(Thread.java:748)\nCaused by: java.io.EOFException\n\tat
com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.read(ByteArrayInputStream.java:1
90)\n\tat
com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.readInteger(ByteArrayInputStream
.java:46)\n\tat
com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserial
ize(EventHeaderV4Deserializer.java
:35)\n\tat
com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserial
ize(EventHeaderV4Deserializer.java
:27)\n\tat
com.github.shyiko.mysql.binlog.event.deserialization.EventDeserializer.nextEvent(EventD
eserializer.java:212)\n\tat
io.debezium.connector.mysql.BinlogReader$1.nextEvent(BinlogReader.java:224)\n\tat
com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.ja
va:922)\n\t... 3 more\n"
```

KAFKA CONNECT

Troubleshooting Kafka Connect

The Log is the Source of Truth:

There are different ways to access the log, depending on how you are running Connect:

- If you are just running the Confluent CLI locally, the command is `confluent local services connect log`
- If you are using Docker, it's `docker logs`, plus the name of the container
- If you are running completely vanilla Connect using Apache Kafka, you can just read the log files with `cat`, or more likely `tai`
  by installation)

KAFKA CONNECT

Troubleshooting Kafka Connect

“Task is being killed and will not recover until manually restarted”

```
ache.kafka.connect.runtime.WorkerSourceTask)
[2019-05-07 14:39:13,115] INFO WorkerSourceTask{id=source-debezium-orders-00-0} Finished commitOffsets successfully in 28 ms (org.apache.
kafka.connect.runtime.WorkerSourceTask)
[2019-05-07 14:39:13,116] ERROR WorkerSourceTask{id=source-debezium-orders-00-0} Task threw an uncaught and unrecoverable exception (org.
apache.kafka.connect.runtime.WorkerTask)
org.apache.kafka.connect.errors.ConnectException
   at io.debezium.connector.mysql.AbstractReader.wrap(AbstractReader.java:230)
   at io.debezium.connector.mysql.AbstractReader.failed(AbstractReader.java:197)
   at io.debezium.connector.mysql.BinlogReader$ReaderThreadLifecycleListener.onCommunicationFailure(BinlogReader.java:1018)
   at com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.java:950)
   at com.github.shyiko.mysql.binlog.BinaryLogClient.connect(BinaryLogClient.java:580)
   at com.github.shyiko.mysql.binlog.BinaryLogClient$7.run(BinaryLogClient.java:825)
   at java.lang.Thread.run(Thread.java:748)
Caused by: java.io.EOFException
   at com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.read(ByteArrayInputStream.java:190)
   at com.github.shyiko.mysql.binlog.io.ByteArrayInputStream.readInteger(ByteArrayInputStream.java:46)
   at com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserialize(EventHeaderV4Deserializer.java:35)
   at com.github.shyiko.mysql.binlog.event.deserialization.EventHeaderV4Deserializer.deserialize(EventHeaderV4Deserializer.java:27)
   at com.github.shyiko.mysql.binlog.event.deserialization.EventDeserializer.nextEvent(EventDeserializer.java:212)
   at io.debezium.connector.mysql.BinlogReader$1.nextEvent(BinlogReader.java:224)
   at com.github.shyiko.mysql.binlog.BinaryLogClient.listenForEventPackets(BinaryLogClient.java:922)
   ... 3 more
[2019-05-07 14:39:13,121] ERROR WorkerSourceTask{id=source-debezium-orders-00-0} Task is being killed and will not recover until manually restarted
 (org.apache.kafka.connect.runtime.WorkerTask)
```

There is something about the network connection or about the byte-level IO between the source database and the co

Conclusion

Quality Network for Education and Technology

XIN CHÂN THÀNH CẢM ƠN!