Kafka Connector install notes

1. **Using servers mdbconnectdev1, kconnectdev2**
   1. Linux :
      1. Operating System: Oracle Linux Server 8.6
      2. CPE OS Name: cpe:/o:oracle:linux:8:6:server
      3. Kernel: Linux 4.18.0-372.9.1.el8.x86\_64
   2. Log in as dev1 Assurity123

1. **Download java**
   1. Unzip to /User/bin/Java/
   2. *'sudo tar zxvf openjdk-18.0.1.1\_linux-x64\_bin.tar.gz'*
   3. We are currently using openjdk-18.0.1.1\_linux-x64\_bin.tar.gz

1. **Setup keystore and cert**
   * **add MongoDb ca to ca truststore**
     + **This should be a .crt file with just the certification section in it.**
     + **Run *'sudo /usr/bin/java/jdk-18.0.1.1/bin/keytool -import -trustcacerts -alias mongoDb -file mongoCA.crt -cacerts'* .**

This will add this cert to Java security: /usr/bin/java/jdk-18.0.1.1/lib/security/cacerts.

* Add machine cert: copy .pfx file to config dir:  *'sudo cp ~/dloads/kconnectdev2.pfx /usr/bin/confluent/confluent-7.1.1/config'*
* Add path and pwds to env so that kafka can use them:
  + In /home/dev1/.bashrc add:

*export KAFKA\_OPTS="\*

*-Djavax.net.ssl.keyStore=/usr/bin/confluent/confluent-7.1.1/config/kconnectdev2.pfx \*

*-Djavax.net.ssl.keyStorePassword=Assurity123 \*

*-Djavax.net.ssl.trustStore=/usr/bin/java/jdk-18.0.1.1/lib/security/cacerts \*

*-Djavax.net.ssl.trustStorePassword=changeit"*

Check and see if it applied by running 'env' after relogging in.

1. **Download and unzip the confluent community edition**
   1. Version we are using confluent-7.1.1.tar.gz
   2. In /usr/bin/confluent: *'sudo mkdir /usr/bin/confluent'*
   3. Copy it from the dir you sftp'd the .qz file to the dir you just created - *' sudo cp home/dev1/dloads/confluent-7.1.1.tar.gz /usr/bin/confluent'*
   4. Make sure you are in /usr/bin/confluent then Untar  *'sudo tar zxvf confluent-7.1.1.tar.gz'*
   5. Update the path for the dev1 user envirnment. In /home/dev1/.bashrc add: *PATH="/usr/bin/confluent/confluent-7.1.1/bin:/usr/bin/java/jdk-18.0.1.1/bin:$PATH"*

export PATH

To the end of the file. When you re log in you can type 'env' to see if the path was updated correctly. This path can be updated later if we run into an issue or new install

1. **Connectors Via Confluent\_Hub**
   1. Example cmd forMongodb Source : *./confluent-hub install debezium/debezium-connector-mongodb:1.9.3*

Target path /usr/bin/confluent/confluent-7.1.1/share/confluent-hub-components. Answer 'y' to all the prompt, Including update the example .properties files. There are 4 yes/no questions to answer

1. Example cmd for MongoDbSink

*'confluent-hub install redhatinsights/expandjsonsmt:0.0.7'*

*'confluent-hub install hpgrahsl/kafka-connect-mongodb:1.4.0'*

1. Example cmd for SQL source connector

*confluent-hub install debezium/debezium-connector-sqlserver:1.9.3*

1. Example cmd SQL sink connector

confluent-hub install mongodb/kafka-connect-mongodb:latest

1. **Add configs:**
   1. Make config dir *'sudo mkdir /usr/bin/confluent/confluent-7.1.1/config'*
   2. Note: .properties files are for standalone config, the Json is used to load the configs for all the workers in the cluster.
   3. For Distributed configuration
      1. To run : */usr/bin/confluent/Confluent\*/bin/connector-distributed connect-distributed.properties* (This config should be server agnostic )
      2. When this is run the first time it won't have the proper configs pushed to it
      3. Useful Curl commands:

*curl* [*http://localhost:8083/connector-plugins*](http://localhost:8083/connector-plugins) - this will show the connectors that are currently configs for the connection worker on this machine. It should show the connectors and versions

*curl* [*http://localhost:8083/connectors*](http://localhost:8083/connectors)

*curl* [*http://localhost:8083/connectors?expand=status*](http://localhost:8083/connectors?expand=status)

1. Add or update configs for each connector. Ex:

*(Post is for adding a new connector, Put is for updating a connector)*

*curl -s -X POST -H 'Content-Type: application/json' --data @mongodbSource.json* [*http://localhost:8083/connectors*](http://localhost:8083/connectors)

*curl -s -X POST -H 'Content-Type: application/json' --data @sqlserverSource.json* [*http://localhost:8083/connectors*](http://localhost:8083/connectors)

*curl -s -X PUT -H 'Content-Type: application/json' --data @sqlserverSourceUpdate.json* [*http://localhost:8083/connectors/sql-source-connector/config*](http://localhost:8083/connectors/sql-source-connector/config)

*curl -s -X PUT -H 'Content-Type: application/json' --data @mongodbSource.json* [*http://localhost:8083/connectors/mongodbsource/config*](http://localhost:8083/connectors/mongodbsource/config)

**Notes**

* You may need a key when adding a new topic/connector. You can get this from the confluent cloud web interface.
* Pwd for ssl cert = changeme or changeit

Below are old notes from stand alone proof of concept. They were just a draft and have not been updated but may have some useful info if trying to debug an issue:

* *Download and untar the Debezium mongodb soruce connector (CFM, add link, ex path to install to)*
  + *Version : debezium-debezium-connector-mongodb-1.9.2.zip*
  + *Install to /usr/bin/confluent/confluent-7.1.1/share/java/plugins*
    - *Create plugins dir in ./java. 'sudo mkdir plugins'*
  + *Unzip : 'sudo unzip debezium-debezium-connector-mongodb-1.9.2.zip'*

* *Other connectors ( If you just want to test basic connectivity, certs and paths you can run in stand alone mode to check - see the step on running stand alone.*

* *Install MongoDbSinkCoonector*
* *Mongo db Sink connector*
  + *Copy 'sudo cp ~/dloads/mongodbSinkConnector\_kconnectdev2.properties /usr/bin/confluent/confluent-7.1.1/config'*
  + *Name: mongodbSinkConnector\_kconnectdev2*
  + *Topic: CFM - update these!*
  + *Databse*
  + *Collexction*
  + *Install location from the confluent hub /usr/bin/confluent/confluent-7.1.1/share/confluent-hub-components*
  + *confluent-hub install redhatinsights/expandjsonsmt:0.0.7*

*Configs /usr/bin/confluent/confluent-7.1.1/etc/kafka/*

* *, copy worker file from install (cfm add location in install and add copy of exeample) connect-standalone.properites*

* *To test basic setup - run standalone from config dir:*
  + */usr/bin/confluent/confluent-7.1.1/bin/connect-standalone connect-standalone\_kconnectdev2.properties mongodbSrcConnector\_kconnectdev2.properties*
  + *Let it sit for a while. Eventually you will get an error or will have messages about records being processed (at least the first time running) or 0 records being processed (if it has been run before)*
  + *Check* [*www.confluent.cloud*](http://www.confluent.cloud) *-> env -> cluster -> data integration -> clients. You should see a client ID that matches the name field in the src .properties file. If you do…. Congrats, Kafka Connect on premises is able to talk to cloud cluster.*

* Add configs:
  + Make config dir *'sudo mkdir /usr/bin/confluent/confluent-7.1.1/config'*
  + Note: .properties files are for standalone config, the Json is used to load the configs for all the workers in the cluster.
  + First add the stand connection-standalone\*.properties file. Look through it and see if there are an machine specific names. Update if needed. Pay attention to paths and names - like plugin.path. NOTE: In plugin.path make sure to add the trailing '/'
    - Ex plugin.path=/usr/share/java,/home/dev1/confluent/confluent-7.1.1/share/confluent-hub-components**/**
  + Then do mongodbConnector.properties