FABRIC UPGRADE PROCEDURE TO V7.2

This document describes:

- How to upgrade Fabric to the present version: from the latest **V7.1.1** to **V7.2.**
- How to re-implement the modified product features.

Notes:

- This document does not cover the Fabric server topology changes, such as nodes addition, DCs, change of replication factor and consistency level.
- It is a must to perform the Fabric upgrade procedure in the testing environments prior to applying it on your production deployment.
- Sanity tests must be performed upon the completion of the upgrade procedure, such as performing a few GET commands and conducting other checks per the sanity procedure defined in your project.

SOFTWARE UPGRADE PROCEDURE

Preliminary Step

Download the latest release of Fabric V7.2 package and copy it to the server(s).

Stop Fabric

Take the following steps in the specified order:

1. If your project has an iidFinder:

Stop the iidFinder on all nodes.

Wait until Kafka lags are zero in the relevant Kafka topics:

- Delta_cluster_<LU_name> topics
- DeltaPriority_cluster_<LU_name> topics

Run the following command to verify that the lag on the above topics is zero:

```
/opt/apps/k2view/apps/kafka/bin/kafka-consumer-groups --bootstrap-server <internal Kafka server IP> --group <Kafka interface group ID> --describe | awk '{if ($6>0) print $0}'
```

Investigate the remaining messages in the Delta tables and clean them, by taking the following steps per each LU:

- Run MIGRATE command on all distinct IIDs.
- Check the results in order to decide how to proceed with the failed entity messages.
- Clean the Delta table.

2. Stop Fabric on all nodes.

Open the Package

Perform the following steps:

Rename the Fabric directory as shown, type the specific Fabric version in the indicated location:

```
cp -r config config_$(k2fabric -version |awk '{print $2}'|head -n1)
```

```
mv fabric $(k2fabric -version |awk '{print $2}'|head -n1)
```

Extract (un-TAR) the Fabric directories from the upgrade package (extract only the directories) as shown. The specific file name will depend on the specific Fabric version:

```
tar -zxvf k2fabric-server-fabric-<package_name>.tar.gz fabric apps
```

Adding **apps** after **fabric** installs the correct Java version and creates a soft link to Java.

Run Upgrade Scripts (where applicable)

When upgrading to a version whose number is not consecutive to the version you are using, run all the upgrade scripts of all versions in between. For example, when upgrading from the latest V6.4 to V7.0, run all the upgrade scripts in the following order: 6.4 -> 6.5 -> 7.0.

The versions that have the upgrade scripts between the latest 6.4 and 7.1 are: 6.5.8, 7.0, 7.1.

The upgrade scripts should run from one Fabric node only. After running the scripts, verify that all the changes have been applied.

This step is not relevant for the upgrade from the latest **V7.1.1** to **V7.2**.

Verify Upgrade Success

Use the following command to verify that the upgrade procedure has been completed successfully:

```
k2fabric -version
```

The result will display the Fabric package number, for example:

```
Tag fabric- 7.0.0_188 at revision = 3fc12ed1c6839614a9fd27e2894828bcd160988f
```

If there are local files on the server (such as local JARs), their names will be displayed here.

Configuration Changes

Prior to performing the changes, backup your project configuration files, such as config.ini, etc. Please read the below configuration changes summary per each version.

To Version 7.2

- The .NET Studio has introduced a **new encryption mechanism, AES 256,** for all the project's passwords (used in the interfaces and environments). The encryption upgrade is done automatically upon project opening. In order to apply this change, the environment and the project must be deployed.
  - An automatic passwords upgrade, during the opening or importing of a project, can be **disabled** by setting the following property in the Studio app config to **false**: *AutoUpgradePasswordsOnProjectOpen.*
  - If the automatic upgrade is disabled, the passwords will be upgraded separately on each interface, when opening the interface in the Studio and saving it.

- Note that after upgrading the project's passwords encryption, older versions of Studio or Fabric won't be able to read/decrypt the passwords of the upgraded project. The passwords will have to be re-typed when opening a project in an older Studio version.
- This feature can be disabled by setting the following property in the Studio app config to false: *EnableUpgradePasswordsEncryption*.

➢ **Support SQLite and Postgres as Fabric System DB**: Cassandra, SQLite, or PostgreSQL can now serve as a Fabric System DB.

When keeping Cassandra as a default Fabric System DB, no configuration changes are required. Below are changes of parameter names in the config.ini:

- The existing [cassandra_entity_storage] section in **config.ini** has been renamed to [system_db_entity_storage].
- The existing DEFAULT_GLOBAL_STORAGE_TYPE parameter in the [fabric] section and the existing MDB_DEFAULT_SCHEMA_CACHE_STORAGE_TYPE parameter in the [fabricdb] section have now been set to the new default value: SYSTEM_DB.
- The new [system_db] section has been added to config.ini and it holds the System DB settings. The valid values are CASSANDRA (default), SQLITE and POSTGRESQL.
  - When the SYSTEM_DB_TYPE is either CASSANDRA or POSTGRESQL, it impacts both the Storage and the System DB types, unless the Storage type has been altered by using the DEFAULT_GLOBAL_STORAGE_TYPE parameter.
  - When the SYSTEM_DB_TYPE is set to SQLITE, the System DB will be SQLite, but the Storage will be located at **$FABRIC_HOME/storage/entities**.
- SYNC_CASSANDRA_SYSTEM_AUTH parameter has been replaced in the config.ini by a new parameter - READ_ONLY_AUTHENTICATORS. By default, this parameter is set to false.
- The impact of the System DB update on the **node.id** file settings:
  - When the System DB is set to either SQLite or PostgreSQL, the dc_name will get a default LOCAL_DC value.
  - When the System DB is set to SQLite, the effective_ip will be set by default to 127.0.0.1.
- Note that when moving from Cassandra to another System DB type (SQLite or PostgreSQL), no data transition is provided. You need to re-create your system data (jobs, audit, etc.).

➢ **CommonDB**:

- You can now configure the distribution storage type of the CommonDB messages via the new parameter **MESSAGES_INTERNAL_BROKER_TYPE** defined in the [common_area_config] section of config.ini. By default, it is set to **PUB_SUB** to improve the performance of the long snapshot / transaction loading process.
- The existing [common_area_cassandra_config] section in **config.ini** has been renamed to [common_area_system_db_config].

- As part of the **PUBSUB infrastructure** improvement, the topic is now created only by the Publisher and not by the Subscriber. As a result, if the Subscriber starts polling from an un-existing topic, the warnings will be thrown to the log. This change impacts CommonDB tables without a population. To suppress the warnings in the log, add the following line to the **logback.xml**:
  - <logger name="org.apache.kafka.clients.NetworkClient" level="ERROR" />

Implementation Changes

Please see the below implementation changes summary for each version.

To Version 7.2

N/A

Start Fabric

Take the following steps in the specified order:

1. Start the first Fabric node and wait until it has been successfully completed.
2. Start all other nodes.
3. Deploy the project.
4. If your project has an iidFinder, re-start the iidFinder on all nodes.

Note that the above steps may vary per your project's runbook. For example, you may first restart the iidFinder on a single node only, verify that the process has been successfully completed and then proceed to restarting the iidFinder on all other nodes.