Configuring Run-time Engines for Big Data Management

© Copyright Informatica LLC 2017. Informatica, the Informatica logo, and Big Data Management are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at https://www.informatica.com/trademarks.html.

Abstract

If you installed Informatica Big Data Management and you did not configure the run-time engines during the installation, you can configure the engines later. This article explains how to configure the Blaze engine, the Spark engine, and the Hive engine.

Supported Versions

- Informatica Big Data Management® 10.1

Table of Contents

Overview

You can configure a Blaze engine, a Spark engine, and a Hive engine to run mappings on a Hadoop cluster.

When you run a big data mapping, you can choose to run the mapping in the Informatica environment or the Hadoop environment. The native environment is the Informatica environment. When you choose the native run-time engine, the Data Integration Service runs mappings on the Informatica domain. When you choose a run-time engine in the Hadoop environment, the Data Integration Service pushes the mapping run processing to the cluster.

To run mappings on the cluster, choose from the following run-time engines:

Blaze engine

Informatica run-time engine. The Data Integration Service submits jobs to the Blaze engine executor on the Informatica domain. The Blaze engine executor communicates between the Data Integration Service and the Blaze engine components on the Hadoop cluster.

Spark engine

Apache run-time engine. The Data Integration Service sends a mapping application to the Spark executor on the Informatica domain. The Spark executor submits the job to the Hadoop cluster to run.

Hive engine

A Hive driver that receives HiveQL queries from the Data Integration Service on the Informatica domain. The Hive driver converts the HiveQL queries to MapReduce jobs and sends the jobs to the Hadoop cluster.

Informatica recommends that you select all engines to run mappings in the Hadoop environment. The Data Integration Service uses a proprietary rule-based methodology to determine the best engine to run the mapping. The rule-based methodology evaluates the mapping sources and the mapping logic to determine which engine to run a mapping.

Blaze Engine Configuration

You can configure the Blaze engine to run mappings in the Hadoop environment.

To configure the Blaze engine with Informatica Big Data Management, perform the following tasks:

1. Open ports for the Blaze engine.
2. Prepare the Hadoop Cluster for the Blaze engine.
3. Allocate Cluster Resources for Blaze
4. Configure virtual memory limits.
5. Configure dynamic partitioning.

Open the Required Ports for the Blaze Engine

When you create the Hadoop connection, specify the minimum and maximum port range that the Blaze engine can use. Then open the ports on the cluster for the Blaze engine to use to communicate with the Informatica domain.

The default port for Blaze monitoring is 9080.

The default port range for the Blaze engine is 12300 to 12600.

Note: If the Hadoop cluster is behind a firewall, work with your network administrator to open the range of ports that the Blaze engine uses.

Prepare the Hadoop Cluster for the Blaze Engine

To run mappings on the Blaze engine, you need to create a Blaze engine user and log directories. You also need to grant permissions to the user.

Create a Blaze User Account

On all nodes in the Hadoop cluster, create an operating system user account for the Blaze engine user. For example, run the following command:

```
useradd blaze
```

Create Blaze Engine Directories and Grant Permissions

Create the following directories on the Hadoop cluster:

Local services log directory

Create a local services log directory on all nodes in the cluster and grant permissions to the Blaze user account. The hadoopEnv.properties file on the domain contains an entry for this directory. The file uses an environment variable, $HADOOP_NODE_INFA_HOME, that gets set to the Big Data Management installation directory. The default installation directory is `/opt/Informatica`. For example, run the following commands:

```
hadoop fs mdkir -p /opt/Informatica/blazeLogs
hadoop fs -chmod 777 /opt/Informatica/blazeLogs
```

If you use a different directory name, you must update the following property in the hadoopEnv.properties file: `infagrid.node.local.root.log.dir`

Aggregated HDFS log directory

Create a log directory on HDFS to contain aggregated logs for local services. Grant permissions to the Blaze user account. For example, run the following commands:

```
hadoop fs mdkir -p /var/log/hadoop-yarn/apps/informatica

hadoop fs -chmod 777 /var/log/hadoop-yarn/apps/informatica
```

The hadoopEnv.properties file on the domain contains an entry for this directory. If you use a different directory name, you must update the following property in the hadoopEnv.properties file: `infacal.hadoop.logs.directory`

HDFS temporary working directory

Create a working directory on HDFS for the Blaze engine and grant permissions to the Blaze user account. For example, run the following commands:

```
hadoop fs mkdir -p /blaze/workdir

hadoop fs -chmod 777 /blaze/workdir
```

When you configure connection properties through the Hadoop Configuration Manager or through the Developer tool, you need to provide the path to this directory. Alternatively, you can create this directory when you create the connection.

Grant Permissions on the Hive Source Database

Grant the Blaze user account CREATE TABLE permission on the Hive source database. The CREATE TABLE permission is required in the following situations:

- The Hive source table uses SQL standard-based authorization.
- A mapping contains a Lookup transformation with an SQL override.

Allocate Cluster Resources for Blaze

Update properties in the yarn-site.xml file to verify that the cluster allocates sufficient memory and resources for the Blaze engine.

The yarn-site.xml file is in the following location:

```
/etc/hadop/conf
```

Update the following properties:

yarn.nodemanager.resource.memory-mb

The maximum RAM available for each container. Set the maximum memory on the cluster to increase resource memory available to the Blaze engine.

Configure this property on the following nodes:

- Run-time nodes. Set to at least 10 GB.
- Management node. Set to at least 13 GB.

yarn.nodemanager.resource.cpu-vcores

The number of virtual cores for each container. The number might correspond to the number of physical cores, but you can increase the value to allow for more processing.

Configure this property on the following nodes:

- Run-time nodes. Set to at 6.
- Management node. Set to 9.

yarn.nodemanager.resource.cpu-vcores

The minimum RAM available for each container. Set the minimum memory to allow the VM to spawn sufficient containers. Configure this property to be no less than 4 GB of the maximum memory on the run-time and management nodes.

Configure Virtual Memory Limits

Remove virtual memory limits for every node in the Hadoop cluster.

Configure the following property on the yarn-site.xml file:
yarn.nodemanager.vmem-check-enabled

Determines virtual memory limits. Set the value to false to disable virtual memory limits. For example:

```
<property>
    <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
    <description>Enforces virtual memory limits for containers.</description>
</property>
```

Configure Dynamic Partitioning

If the mappings that run on the Hive engine use Hive dynamic partitioned tables, configure dynamic partition variables in the hive-site.xml file.

Open the following file on the machine that runs the Data Integration Service: `/<BigDataManagementInstallationDirectory>/Informatica/services/shared/hadoop/<Hadoop_distribution_name>/conf/hive-site.xml`.
Configure the following Hive environment variables for dynamic partitioning:

hive.exec.dynamic.partition

Set this property to `TRUE` to enable dynamic partitioned tables.

exec.dynamic.partition.mode

Set this property to `nonstrict` to allow all partitions to be dynamic.

Spark Engine Configuration

To configure the Spark run-time engine, perform the following tasks:

1. Configure Spark properties to increase performance.
2. Enable dynamic allocation.

The Spark engine can run mappings on all Hadoop distributions.

Configure Performance Properties

You can configure Spark properties to improve performance of mappings that run on the Spark run-time engine.

1. Open the `hadoopEnv.properties` file and back it up.

   The `hadoopEnv.properties` file is on the machine that runs the Data Integration Service. The file is in the following location: `<Informatica installation directory>/services/shared/hadoop/<Hadoop_distribution_name>_<version_number>/infaConf/`

2. Configure the following properties:

Property	Value	Description
spark.dynamicAllocation.enabled	TRUE	Enables dynamic resource allocation. Required when you enable the external shuffle service.
spark.shuffle.service.enabled	TRUE	Enables the external shuffle service. Required when you enable dynamic allocation.
spark.scheduler.maxRegisteredResourcesWaitingTime	15000	The number of milliseconds to wait for resources to register before scheduling a task. Reduce this from the default value of 30000 to reduce any delay before starting the Spark job execution.
spark.scheduler.minRegisteredResourcesRatio	0.5	The minimum ratio of registered resources to acquire before task scheduling begins. Reduce this from the default value of 0.8 to reduce any delay before starting the Spark job execution.

3. Locate the spark.executor.instances property and place a # character at the beginning of the line to comment it out.

   Note: If you enable dynamic allocation for the Spark engine, Informatica recommends that you comment out this property.

   After editing, the line appears as follows:

   ```
   #spark.executor.instances=100
   ```

Configure Dynamic Resource Allocation for Spark

You can dynamically adjust the resources that an application occupies based on the workload. You can configure dynamic resource allocation for mappings that run on the Spark engine.

Note: Hadoop distribution vendors publish information on configuring dynamic resource allocation for their cluster environments. Check their documentation for additional information for this task.

Configuring Dynamic Resource Allocation on Amazon EMR Clusters

1. Copy the Spark shuffle .jar file from the Hadoop distribution library on the cluster to the following directory on each of the cluster nodes where the YARN manager is running:

   ```
   /usr/lib/hadoop-yarn/lib
   ```

2. On each of the cluster nodes where Yarn node manager is running, open the following file for editing:

   ```
   /etc/hadoop/conf/yarn-site.xml
   ```

3. Add the following properties and values to yarn-site.xml:

   ```
   <property>
       <name>yarn.nodemanager.aux-services</name>
       <value>mapreduce_shuffle,spark_shuffle</value>
   </property><property>
       <name>yarn.nodemanager.aux-services.spark_shuffle.class</name>
       <value>org.apache.spark.network.yarn.YarnShuffleService</value>
   </property>
   ```

4. On all nodes where the node manager runs, restart the yarn node manager service.
5. Save and close the file.

Configuring Dynamic Resource Allocation on Cloudera Clusters

1. On the machine where the Data Integration Service runs, back up the following file and then open it for editing: `<InformaticaInstallationDir>/services/shared/hadoop/cloudera_cdh<version_number>/infaConf/HadoopEnv.properties`
2. Configure the following properties:

Property	Set Value to
spark.dynamicAllocation.enabled	TRUE
spark.shuffle.service.enabled	TRUE

3. Locate the spark.executor.instances property and comment it out.

   The following example shows the syntax:

   ```
   #spark.executor.instances=100
   ```
4. On the cluster name node, use a command window to browse to the following directory: `/opt/cloudera/parcels/CDH-<version>/lib/hadoop-yarn/lib/`. Verify that one of the following .jar files is present:
   - `spark-1.6.0-cdh<version_number>-yarn-shuffle.jar`
   - `spark-yarn-shuffle.jar`

   If either file is present, you do not need to update the classpath. If neither file is present, use the .jar file bundled with the Big Data Management download. Use the Ambari or Cloudera cluster configuration browser to update the yarn.application.classpath property to include one of the following values:
   - For Spark 1.6x:

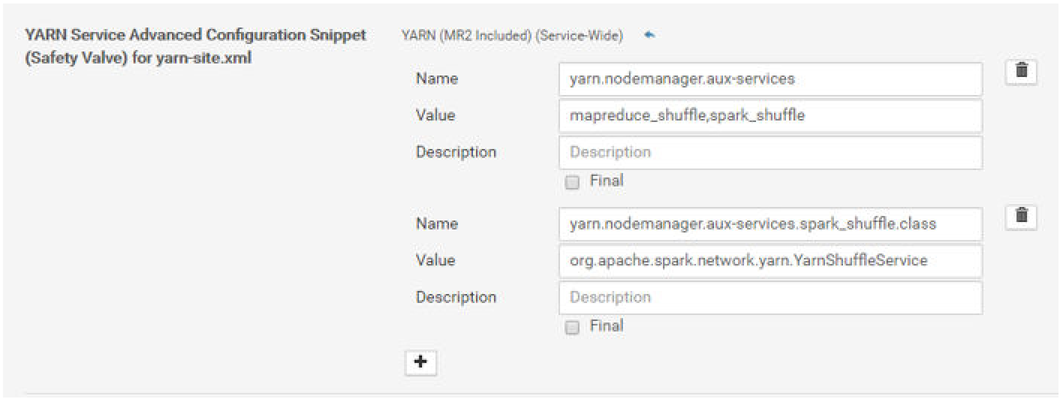
     ```
     /opt/Informatica/services/shared/hadoop/cloudera_cdh5u8_custom/spark/lib/spark-1.6.0-cdh5.8.0-yarn-shuffle.jar
     ```
   - For Spark 2.0x:

     ```
     /opt/Informatica/services/shared/spark/lib_spark_2.0.1_hadoop_2.6.0/yarn/spark-2.0.1-yarn-shuffle.jar
     ```
5. In the cluster configuration manager interface, browse to the following configuration screen: **YARN Service Advance Configuration Snippet (Safety Valve) for yarn-site.xml**.

   Add the following properties:

Name	Value
yarn.nodemanager.aux-services	mapreduce_shuffle,spark_shuffle
yarn.nodemanager.aux-services.spark_shuffle.class	org.apache.spark.network.yarn.YarnShuffleService

The following image shows the configuration screen:


YARN Service Advanced Configuration Snippet (Safety Valve) for yarn-site.xml

YARN (MR2 Included) (Service-Wide)

Name	yarn.nodemanager.aux-services
Value	mapreduce_shuffle,spark_shuffle
Description	Description
	Final
Name	yarn.nodemanager.aux-services.spark_shuffle.class
Value	org.apache.spark.network.yarn.YarnShuffleService
Description	Description
	Final

6. Restart the Yarn service on the cluster.

   When the Yarn service restarts, look for the following message in the Yarn nodemanager log:

   ```
   org.apache.spark.network.yarn.YarnShuffleService: Started YARN shuffle service for
   Spark on port 7337.
   ```

   In the application log on the cluster, look for the following message:

   ```
   INFO util.Utils: Using initial executors = 0, max of
   spark.dynamicAllocation.initialExecutors, spark.dynamicAllocation.minExecutors and
   spark.executor.instances
   ```

Configuring Dynamic Resource Allocation on Hortonworks Clusters

1. On the machine where the Data Integration Service runs, back up the following file and then open it for editing: `<InformaticaInstallationDir>/services/shared/hadoop/hortonworks_<version_number>/infaConf/HadoopEnv.properties`
2. Configure the following properties:

Property	Set Value to
spark.dynamicAllocation.enabled	TRUE
spark.shuffle.service.enabled	TRUE

3. Locate the property spark.executor.instances and comment it out.

   The following example shows the syntax:

   ```
   #spark.executor.instances=100
   ```

4. On the cluster name node, use a command window to browse to the following directory: `usr/hdp/<Current version>`. Verify that one of the following .jar files is present:

   - `/usr/hdp/<Current version>/spark/aux/spark-1.6.2.2.5.0.0-1245-yarn-shuffle.jar`
   - `/usr/hdp/<Current version>/spark2/aux/spark-2.0.0.2.5.0.0-1245-yarn-shuffle.jar`

   If either file is present, you do not need to update the classpath. If neither file is present, use the .jar file bundled witht he Big Data Management download. Use the Ambari or Cloudera cluster configuration browser to update the yarn.application.classpath property to include the following value:

   ```
   /opt/Informatica/services/shared/spark/lib_spark_2.0.1_hadoop_2.6.0/yarn/spark-2.0.1-
   yarn-shuffle.jar
   ```

5. In the Ambari cluster configuration browser, select the YARN service and click the **Advanced** tab.

Add the following property in the **Node Manager** section:

Property	Value
yarn.nodemanager.aux-services	mapreduce_shuffle,spark_shuffle,spark2_shuffle

Add the following properties in the **Advanced yarn-site** section:

Property	Value
yarn.nodemanager.aux-services.spark_shuffle.classpath	{stack_root}}/${hdp.version}/spark/aux/*
yarn.nodemanager.aux-services.spark_shuffle.class	org.apache.spark.network.yarn.YarnShuffleService

6. Restart the YARN service on the cluster.

   When the YARN service restarts, look for the following message in the cluster console:

   ```
   org.apache.spark.network.yarn.YarnShuffleService: Started YARN shuffle service for
   Spark on port <number>.
   ```

   In the application log on the cluster, look for the following message:

   ```
   Using initial executors = 0, max of spark.dynamicAllocation.initialExecutors,
   spark.dynamicAllocation.minExecutors and spark.executor.instances
   ```

Configure Dynamic Resource Allocation on Azure HDInsight and IBM Big Insights

1. On the machine where the Data Integration Service runs, back up the following file and then open it for editing: `<InformaticaInstallationDir>/services/shared/hadoop/hortonworks_<version_number>/infaConf/HadoopEnv.properties`
2. Configure the following properties:

Property	Set Value to
spark.dynamicAllocation.enabled	TRUE
spark.shuffle.service.enabled	TRUE

3. Locate the property spark.executor.instances and comment it out.

   The following example shows the syntax;

   ```
   #spark.executor.instances=100
   ```
4. Locate the Spark shuffle .jar file and note the location. The file is located in the following path: `/opt/Informatica/services/shared/spark/lib_spark_2.0.1_hadoop_2.6.0/spark-network-shuffle_2.11-2.0.1.jar`.
5. Add the Spark shuffle .jar file location to the classpath of each cluster node manager.
6. Edit the yarn-site.xml file in each cluster node manager.
   a. Change the value of the `yarn.nodemanager.aux-services` property as follows:

      ```
      <property>
          <name>yarn.nodemanager.aux-services</name>
          <value>mapreduce_shuffle,spark_shuffle</value>
      </property>
      ```

b. Add the following property-value pairs:

```
yarn.nodemanager.aux-
services.spark_shuffle.class=org.apache.spark.network.yarn.YarnShuffleService
```

Hive Engine Configuration

To configure the Hive engine with Informatica Big Data Management, perform the following tasks:

- Configure dynamic partitioning.
- Configure Hive connection properties for MapR.
- Edit hive-site.xml to use the Hive engine on a MapR cluster.

Configure Dynamic Partitioning

If the mappings that run on the Hive engine use Hive dynamic partitioned tables, configure dynamic partition variables in the hive-site.xml file.

Open the following file on the machine that runs the Data Integration Service: `/<BigDataManagementInstallationDirectory>/Informatica/services/shared/hadoop/<Hadoop_distribution_name>/conf/hive-site.xml`.
Configure the following Hive environment variables for dynamic partitioning:

hive.exec.dynamic.partition

Set this property to `TRUE` to enable dynamic partitioned tables.

exec.dynamic.partition.mode

Set this property to `nonstrict` to allow all partitions to be dynamic.

Configure Hive Connection Properties for MapR

To use Hive to run mappings on a MapR cluster, configure the Data Access Connection String with the service principal name on the machine that runs the Data Integration Service.

1. In the Administrator tool, browse to the **Connections** tab and browse to the **HiveServer2 Connection Properties** area.
2. Configure the following connection properties:

Property	Value
Metadata Connection String	`jdbc:hive2://<domain_host>:<port_number>/default;principal=<service_principal_name>`
Data Access Connection String	`jdbc:hive2://<domain_host>:<port_number>/default;principal=<service_principal_name>` **Note:** You can retrieve the service principal name from the MapR Control System browser.

3. In the **Environment Variables** area, configure the following property to define the Kerberos authentication protocol:

Property	Value
JAVA_OPTS	`-Dhadoop.login=<MAPR_ECOSYSTEM_LOGIN_OPTS> -Dhttps.protocols=TLSv1.2` where <`MAPR_ECOSYSTEM_LOGIN_OPTS`> is the value of the MAPR_ECOSYSTEM_LOGIN_OPTS property in the `/opt/mapr/conf/env.sh` file.

Edit hive-site.xml to Use the Hive Run-Time Engine on a MapR Cluster

To run mappings using Hive on a MapR cluster, edit the file hive-site.xml file on the machine that runs the Data Integration Service. The file is in the following location: `<Informatica installation directory>/services/shared/hadoop/mapr_<version>/conf/hive-site.xml`.
Perform the following steps:

1. Locate the hive.server2.authentication property and change the value as follows:

```
<property>
<name>hive.server2.authentication</name>
<value>kerberos</value>
<description> </description>
</property>
```

2. Add the following property:

```
<property>
<name>hive.metastore.sasl.enabled</name>
<value>true</value>
<description> </description>
</property>
```

3. Open the file `/opt/mapr/conf/hive-site.xml` on the cluster and copy the following properties:

   - hive-metastore.principal
   - hive-metastore.keytab

   Paste both properties to `<Informatica installation directory>/services/shared/hadoop/mapr_<version>/conf/hive-site.xml`.

Author

Ellen Chandler
Principal Technical Writer