Installing Hadoop, the Hadoop Adapter for Intel® EE for Lustre*, and the Job Scheduler Integration

Partner Guide

US
April 12, 2016
World Wide Web: http://www.intel.com

Disclaimer and legal information

Contents

About this Document

Document Purpose

This document provides instructions to perform the following tasks:

- Install Apache* Hadoop and the Hadoop adapter for Intel® EE for Lustre* software
- Configure Hadoop, Yarn, and MapReduce
- Install and configure the Job Scheduler Integration. You have the choice of SLURM (Simple Linux Utility for Resource Management) or PBS (portable batch system).

Intended Audience

The intended audience for this guide are partners who are designing storage solutions based on Intel® Enterprise Edition for Lustre* Software. Readers are assumed to be full-time Linux system administrators or equivalent who have:

- experience administering file systems and are familiar with storage components such as block storage, SAN, and LVM
- experience of knowledgeable about Lustre* installation and setup
- proficiency in setting up, administering and maintaining networks
- proficiency in setting up and administering Hadoop.

Conventions Used

Conventions used in this document include:

- # preceding a command indicates the command is to be entered as root
- $ indicates a command is to be entered as a user
- *<variable_name>* indicates the placeholder text that appears between the angle brackets is to be replaced with an appropriate value

Related Documentation

The following documents are pertinent to Intel® Enterprise Edition for Lustre* software. This list is not all-inclusive.

- *Intel Manager® for Lustre* Software User Guide*
- *Intel® Enterprise Edition for Lustre* Partner Installation Guide*
- *Creating a Scalable File Service for Windows Networks using Intel® EE for Lustre* Software*
- *Creating a Monitored Lustre* Storage Solution over a ZFS File System*
- *Hierarchical Storage Management Configuration Guide*
- *Creating an HBase Cluster and Integrating Hive on an Intel® EE for Lustre® File System*
- *Upgrading a Lustre file system to Intel® Enterprise Edition for Lustre* software (Lustre only)*

- *Configuring LNet Routers for File Systems based on Intel* EE for Lustre* Software*
- *Intel® EE for Lustre* Hierarchical Storage Management Framework White Paper*
- *Architecting a High-Performance Storage System White Paper*

Overview

Intel® EE for Lustre* software version 2.2.0.0 *and later* includes the Hadoop* adapter software (although the Hadoop adapter for Lustre is packaged in a separate tar file from Intel® EE for Lustre* software). As background - Hadoop allows users who run MapReduce jobs to bypass storing data in HDFS, and store the MapReduce output directly to Lustre instead. This allows the analytical processes direct access to scientific output instead of transferring data from the compute cluster storage system to another file system. Optimizations have also been made to the shuffle step in MapReduce to take advantage of Lustre's high speed network access to data. Many workloads will see an overall reduction in end-to-end processing time by using the Hadoop adapter with the Intel® EE for Lustre* file system.

The HPC job scheduler integration with MapReduce, also included with Intel® EE for Lustre* software (and also packaged separately from Intel® EE for Lustre* software), allows you to integrate common resource schedulers into your cluster. You have the choice of installing the SLURM (Simple Linux Utility for Resource Management) job scheduler integration or the PBS (portable batch system) job scheduler integration. Instructions are provided in this guide.

Assumptions

This document assumes that Intel® EE for Lustre* software, version 2.2.0.0 *or later*, is already installed and the file system is running. See the Intel® EE for Lustre* Partner Installation Guide for instructions.

Note: References to Lustre throughout this document refer to Intel® EE for Lustre* software.

Software Version Compatibilities

Note: The use of Cloudera Hadoop with Intel® EE for Lustre* software with is not currently supported by Cloudera.

Throughout this document, the string *<VERSION>* is used as placeholder for the correct software version number. Substitute the version number based on the following compatibility information provided next. For the latest information regarding supported Hadoop distributions and adapter compatibilities, please visit the following site:

https://wiki.hpdd.intel.com/login.action;jsessionid=1F2FF698F1321ADE56B63F7B7A57FFDF?os_destination=%2Fpages%2Fviewpage.action%3FpageId%3D25135680

Or contact your Intel® technical support representative.

Note: In this document, the *Hadoop adapter for Lustre* and the *hadoop-lustre-plugin* refer to the same software.

Target Hadoop Distribution	Hadoop Adapter for Lustre[1]	HAM[2] Version	Job Scheduler Versions	Java Version (Recommended)
hadoop-2.5.0-apache	hadoop-lustre-plugin-3.0.0	hadoop-hpc-scheduler-3.0.0	SLURM 2.5.6 or PBS 2.5.6	Java Runtime Environment 1.7.0+

*HAM refers to the HPC adapter for Mapreduce job schedulers.

Topology

Before installing and configuring Hadoop, you need to identify your existing Lustre cluster and its server roles. You also need to plan the topology of the Hadoop cluster and those server roles. The following tables offer an example list of servers and their roles. For our Hadoop cluster, we are showing one machine running *ResourceManager*, four machines running *NodeManager*, and one machine running *HistoryManager*. Your configuration may be different with respect to the number of NodeManagers and the HistoryManager, which is optional.

Note: Hostnames are required for later configuration in this process; these are examples only that we will use in this document. Be sure to make note of your hostnames and role assignments.

Lustre cluster

Hostname	Roles
st31-mds[1-2] (2 nodes)	Metadata Server
st31-oss[1-2] (2 nodes)	Object Storage Server

Hadoop cluster

Hostname	Hadoop-Yarn Roles
st31-cli1	ResourceManager
st31-cli[2-5] (4 nodes)	NodeManager
st31-cli6	Historyserver

Installing Apache Hadoop

Perform the following steps:

1. Download Apache Hadoop tar from the repository:

   #wget http://apache.fastbull.org/hadoop/common/hadoop-<*VERSION*>/hadoop-<*VERSION*>tar.gz

Untar the archive in /usr/lib:

   #tar –zxvf hadoop-<*VERSION*>.tar.gz

   The HADOOP_HOME will be /usr/lib/hadoop-<*VERSION*>

Verify any existing java installation using the following command:

   /usr/sbin/alternatives --config java

Download from Oracle Java JDK 7 and install it:

   #rpm -ivh jdk-7u45-linux-x64.rpm

Verify that Oracle Java is configured correctly:

   #java –version

   java version "1.7.0_45"

   Java(TM) SE Runtime Environment (build 1.7.0_45-b13)

   Java HotSpot(TM) 64-Bit Server VM (build 24.55-b03, mixed mode)

Add java.sh script to /etc/profile.d with this content:

   export JAVA_HOME=/usr/java/jdk1.7.0_45

Please verify that all the nodes have the correct hostname. For example, on our node st31-cli1, the output of the command hostname would be: st31-cli1

Mount the Lustre File System on Hadoop

We are running Apache in a non-secure mode as root user.

1. Mount the Lustre file system on each Hadoop node using the following commands. (Addresses are examples only.)

   #mkdir /mnt/lustre

mount –t lustre st31-mds1@tcp1:st31-mds2@tcp1: /lustre/mnt/lustre

Verify success using the df –h command on each node.

Create the root tree for Hadoop on a single Hadoop node:

#mkdir /mnt/lustre/hadoop-apache

Use alternatives to manage different Hadoop configurations on all Hadoop nodes:

```
# cp -r /usr/lib/hadoop-<VERSION>/etc/hadoop /etc/hadoop/conf.apache
#alternatives --install /etc/hadoop/conf hadoop-conf
/etc/hadoop/conf.apache 70
#alternatives --set hadoop-conf /etc/hadoop/conf.apache
#alternatives --config hadoop-conf
```

Installing the Hadoop Adapter for Lustre on Apache Hadoop

For additional information, see http://hadoop.apache.org/docs/r2.3.0/hadoop-project-dist/hadoop-common/ClusterSetup.html

Prepare Lustre directory for Hadoop

These instructions need to be performed on all hadoop nodes.

1. A non-root user with access to the Lustre file system must have user ID (UID) and group ID (GID) matches between the Lustre metadata server (MDS) and clients. Hadoop uses yarn as a non-root user to access underlying file system. A yarn user must exist on Lustre* MDS and Hadoop nodes.
2. Verify that yarn UID and GID are identical on Lustre MDS and Hadoop nodes (where Lustre is mounted). Use usermod and groupmod to make necessary adjustments.
3. Create a directory on Lustre that will act as the root directory for Hadoop I/O. All paths on the Lustre file system will be resolved relative to this base directory. For example, if /mnt/lustre/hadoop is the Hadoop root directory, then the absolute path /inputs/data will translate to /mnt/lustre/hadoop/inputs/data.

Note: All commands below must run as root user.

4. The Hadoop root directory must be accessible to all users.

   # chmod ugo+rwx /mnt/lustre/hadoop

5. Add the group “hadoop” to the Hadoop root directory's Access Control List so that Hadoop daemons are able to access job token files on Lustre:

   # setfacl -R -m group:hadoop:rwx /mnt/lustre/hadoop

6. The above step however only sets permissions on existing directory, but newly created directories will not inherit them. For that, we need to set the default mask (notice the extra -d option):

   # setfacl -R -d -m group:hadoop:rwx /mnt/lustre/hadoop

7. Verify that access is correct by running the following command on all nodes; it should return information about Lustre file system

   # sudo -u yarn df /mnt/lustre/hadoop

Install the Hadoop Adapter for Lustre

These instructions need to be performed on all hadoop nodes.

1. If you are installing on systems running Intel® EE for Lustre* software version 2.4.n.n or later, then *download* the file *hadoop-lustre-plugin-<HAL Version>.jar* to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore, the folder to download to is: `/usr/lib/hadoop-<version>/share/hadoop/common/lib`.

   If you are installing on systems running Intel® EE for Lustre* software version 2.2 or 2.3, then this jar file is already downloaded: *copy* the file *hadoop-lustre-plugin-<HAL Version>.jar* to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore:

#cp /root/install/ieel-<version>/hadoop/hadoop-lustre-plugin-<version>.jar /usr/lib/hadoop-<version>/share/hadoop/common/lib/

2. Change permissions on this file:

   ```
   #chmod 644 /usr/lib/hadoop-<version>/share/hadoop/common/lib/
   hadoop-lustre-plugin-<version>.jar
   ```

3. Add the hadoop.sh script to /etc/profile.d with this content:

   ```
   #HADOOP VARIABLES START
   export HADOOP_INSTALL=/usr/lib/hadoop-<VERSION>
   export PATH=$PATH:$HADOOP_INSTALL/bin
   export PATH=$PATH:$HADOOP_INSTALL/sbin
   export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
   export HADOOP_COMMON_HOME=$HADOOP_INSTALL
   export HADOOP_HDFS_HOME=$HADOOP_INSTALL
   export YARN_HOME=$HADOOP_INSTALL
   export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
   export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
   ```

```
export HADOOP_CONF_DIR=/etc/hadoop/conf
#HADOOP VARIABLES END
```

Edit Configuration Files

Next, for each Hadoop node, we need to edit the following three configuration files located in the directory /etc/hadoop/conf.

- core-site.xml
- yarn-site.xml
- mapred-site.xml

These files are edited based on the Hadoop-Yarn role for that node.

core-site.xml

On all nodes, make these changes to the file core-site.xml.

Property	Value	Description
fs.defaultFS	lustre:///	Configure Hadoop to use Lustre as the default file system.
fs.lustre.imp	org.apache.hadoop.fs.LustreFileSystem	Configure Hadoop to use Lustre file system
fs.AbstractFileSystem.lustre.impl	org.apache.hadoop.fs.LustreFileSystem$LustreFs	Configure Hadoop to use Lustre class
fs.root.dir	<lustre mount point>/<new-directory-name> (i.e.: /mnt/lustre/hadoop)	Hadoop root directory on Lustre mount point.

Following are the contents of a sample core-site.xml file.

```
<configuration>
<property>
  <name>fs.defaultFS</name>
  <value>lustre:///</value>
</property>

<property>
  <name>fs.lustre.impl</name>
  <value>org.apache.hadoop.fs.LustreFileSystem</value>
</property>

<property>
  <name>fs.AbstractFileSystem.lustre.impl</name>
  <value>org.apache.hadoop.fs.LustreFileSystem$LustreFs</value>
```

```
</property>

<property>
  <name>fs.root.dir</name>
  <value>/mnt/lustre/hadoop-apache</value>
</property>

<property>
  <name>lustre.stripe.count</name>
  <value>1</value>
</property>

<property>
  <name>lustre.stripe.size</name>
  <value>4194304</value>
</property>

<property>
  <name>fs.block.size</name>
  <value>1073741824</value>
</property>
</configuration>
```

Mapred-site.xml

Make the configuration changes listed in the following table to the file mapred-site.xml. Make these changes on all Hadoop nodes.

Property	Value	Description
mapreduce.map.speculative	false	Turn off map tasks speculative execution (this is incompatible with Lustre currently)
mapreduce.reduce.speculative	false	Turn off reduce tasks speculative execution (this is incompatible with Lustre currently)
mapreduce.job.map.output.collector.class	org.apache.hadoop.mapred.SharedFsPlugins$MapOutputBuffer	Defines the MapOutputCollector implementation to use, specifically for Lustre, for shuffle phase
mapreduce.job.reduce.shuffle.consumer.plugin.class	org.apache.hadoop.mapred.SharedFsPlugins$Shuffle	Name of the class whose instance will be used to send shuffle requests by reducetasks of this job

Following are the contents of a sample mapred-site.xml file.

<property>

```
 <name>mapreduce.map.speculative</name>
 <value>false</value>
</property>

<property>
 <name>mapreduce.reduce.speculative</name>
 <value>false</value>
</property>

<property>
 <name>mapreduce.job.map.output.collector.class</name>
<value>org.apache.hadoop.mapred.SharedFsPlugins$MapOutputBuffer</value>
</property>

<property>
 <name>mapreduce.job.reduce.shuffle.consumer.plugin.class</name>
 <value>org.apache.hadoop.mapred.SharedFsPlugins$Shuffle</value>
</property>

<property>
 <name>mapreduce.framework.name</name>
 <value>yarn</value>
</property>

<property>
 <name>mapreduce.input.fileinputformat.split.minsize</name>
 <value>2147483648</value>
</property>
```

Yarn-site.xml

Make the following configuration changes to the file yarn-site.xml. Make these changes on all Hadoop nodes.

Following is an example of yarn-site.xml located in /etc/hadoop/conf

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

<property>
```

```
    <name>yarn.nodemanager.aux-
services.mapreduce_shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>

  <property>
    <name>yarn.log-aggregation-enable</name>
    <value>true</value>
  </property>

<property>
      <name>yarn.resourcemanager.hostname</name>
      <value>st31-cli1</value>
  </property>

<property>
      <name>yarn.scheduler.minimum-allocation-mb</name>
      <value>512</value>
</property>

<property>
      <name>yarn.scheduler.maximum-allocation-mb</name>
      <value>16384</value>
</property>

<property>
      <name>yarn.nodemanager.resource.memory-mb</name>
      <value>18432</value>
</property>
```

Directory Creation and Configuration Deployment

Copy the files core-site.xml, mapred-site.xml, and yarn-site.xml to all the Hadoop nodes in /etc/hadoop/conf

1. Create the temp directory:

```
#$YARN_HOME/bin/hadoop fs -mkdir /tmp
#$YARN_HOME/bin/hadoop fs -chmod -R 1777 /tmp
```

Prepare the tree for the history server:

```
#$YARN_HOME/bin/hadoop fs -mkdir -p /user/history
#$YARN_HOME/bin/hadoop fs -chmod -R 1777 /user/history
#$YARN_HOME/bin/hadoop fs -chown mapred:hadoop /user/history
```

Verify the Hadoop Adapter Configuration

1. Verify that the Lustre file system is working correctly using these command as root:

    # hadoop fs -mkdir -p /test1/test2

Assuming that fs.root.dir is set to /mnt/lustre/hadoop, the directory test1 should have been created under /mnt/lustre/hadoop and directory test2 should have been created under test1. The command:

    # hadoop fs -ls /test1

    should return directory information /test1/test2

Start YARN and the MapReduce JobHistory Server

After making the required configuration changes above, only the YARN daemons need to be started in order to run MapReduce jobs. Because Lustre replaces HDFS as the default file system, HDFS daemons are not required.

Note: Be sure to always start ResourceManager before starting NodeManager services.

1. As root user on the ResourceManager system, enter the command:

    # $HADOOP_HOME/sbin/yarn-daemon.sh start resourcemanager

As root user on each NodeManager system, enter the command:

    # $HADOOP_HOME/sbin/yarn-daemon.sh start nodemanager

As root user on the MapReduce JobHistory Server system, enter the command:

    # $HADOOP_HOME/sbin/mr-jobhistory-daemon.sh start historyserver

Enter this command to verify the topology::

    #$YARN_HOME/bin/yarn node –list

Run a Test MapReduce Job

Run a test MapReduce job using one of the examples included within the Hadoop distribution:

$HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-*<VERSION>*.jar pi 4 1000

Installing and Configuring an HPC Job Scheduler

HPC job schedulers are often used in high performance computing environments to allocate resources and to start or monitor tasks. When integrating Hadoop clients with Lustre storage, it is beneficial to use an existing HPC job scheduler rather than using YARN to schedule Hadoop jobs directly. This allows administrators to initiate and manage scheduling the same

way between HPC and Hadoop clients. The section describes how to install Intel's HPC job scheduler integration software for Mapreduce, and how to integrate common resource schedulers into your cluster.

You have the choice of installing SLURM (Simple Linux Utility Resource Management) or PBS (portable batch system). See the appropriate section:

- Installing and Configuring the Job Scheduler Integration for SLURM
- Installing and Configuring the Job Scheduler Integration for PBS

Installing and Configuring the Job Scheduler Integration for SLURM

Prerequisites

This section describes how to install and configure the HPC job scheduler integration with Map Reduce to support SLURM.

The following software must be installed.

- Intel® Enterprise Edition for Lustre* software, version 2.2.0.0 or later.
- Apache Hadoop version 2.5.
- The Hadoop* Adapter for Intel® EE for Lustre* Software.
- SLURM version 2.5.6.

Install the Job Scheduler Integration for SLURM

If you are installing on systems running Intel® EE for Lustre* software version 2.4.0.0 or later, then *download* the file *hadoop-hpc-scheduler-<VERSION>.jar* to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore, the folder to download to is: `/usr/lib/hadoop-<version>/share/hadoop/common/lib`.

If you are installing on systems running Intel® EE for Lustre* software version 2.2 or 2.3, then this jar file is already downloaded; *copy* the file *hadoop-hpc-scheduler-<VERSION>.jar* . to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore:

```
#cp /root/install/ieel-<version>/hadoop/ hadoop-hpc-scheduler-
<VERSION>.jar /usr/lib/hadoop-<version>/share/hadoop/common/lib/.
```

Note: This is *not* the same jar file as that used to support PBS.

Configure the Job Scheduler Integration

Configuration is implemented by the configuration file yarn-site.xml. Following is the configuration change required in yarn-site.xml to run MapReduce jobs with the adapter:

```
<property>
<description>RPC class implementation</description>
<name>yarn.ipc.rpc.class</name>

<value>org.apache.hadoop.yarn.hpc.HadoopYarnHPCRPC</value>
</property>
```

You should now be able to run any MapReduce job just like you would with YARN. You are not required to have any of the YARN daemons (node or resource managers) running.

For an extended list of available configuration options, see the next section.

Job Scheduler Integration for SLURM, Configuration File

This file contains a range of configuration options. The following is an example only.

```
<?xml version="1.0"?>
<!-- Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License. You
may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0 Unless required by
applicable law or agreed to in writing, software distributed under
the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES
OR CONDITIONS OF ANY KIND, either express or implied. See the
License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file. -->

<configuration>
<!-- HPC configurations -->

<property>
<description>RPC class implementation</description>
<name>yarn.ipc.rpc.class</name>
<value>org.apache.hadoop.yarn.hpc.HadoopYarnHPCRPC</value>
</property>

<property>
<description>HPC Application Client class
implementation</description>
<name>yarn.application.hpc.client.class</name>
<value>org.apache.hadoop.yarn.hpc.slurm.SlurmApplicationClient</valu
```

```
e></property>

<property>
<description>HPC Application Master class
implementation</description>
<name>yarn.application.hpc.applicationmaster.class</name>
<value>org.apache.hadoop.yarn.hpc.slurm.SlurmApplicationMaster</valu
e></property>

<property>
<description>HPC Application Container Manager class implementation
</description>
<name>yarn.application.hpc.containermanager.class</name>
<value>org.apache.hadoop.yarn.hpc.slurm.SlurmContainerManager</value
>
</property>

<property>
<description>Work directory for yarn applications in the HPC
environment. This directory should exist before submitting
application/job to HPC Scheduler.
</description><name>yarn.application.hpc.work.dir</name><value>/tmp/
</value></property>

<property>
<description>Work directory for yarn applications in the HPC
environment. This directory should exist before submitting
application/job to HPC Scheduler. </description>
<name>yarn.log.dir</name>
<value>/tmp/hadoop</value>
</property>

<property>
<description>Local directories for yarn applications in the HPC
environment. If this value is not set or empty, it takes the value
of 'yarn.nodemanager.local-dirs' configuration.</description>
<name>yarn.application.hpc.local.dirs</name>
<value/></property>

<property>
<description>Logs directories for yarn applications in the HPC
environment. If this value is not set or empty, it takes the value
of 'yarn.nodemanager.log-dirs' configuration.</description>
<name>yarn.application.hpc.log.dirs</name>
<value/></property>
```

```
<property>
<description>sbatch slurm command</description>
<name>yarn.application.hpc.command.slurm.sbatch</name>
<value>sbatch</value>
</property>

<property>
<description>scontrol slurm command</description>
<name>yarn.application.hpc.command.slurm.scontrol</name>
<value>scontrol</value>
</property>

<property>
<description>scancel slurm command</description>
<name>yarn.application.hpc.command.slurm.scancel</name>
<value>scancel</value>
</property>

<property>
<description>srun slurm command</description>
<name>yarn.application.hpc.command.slurm.srun</name>
<value>srun</value>
</property>

<property>
<description>squeue slurm command</description>
<name>yarn.application.hpc.command.slurm.squeue</name>
<value>squeue</value>
</property>

<property><description>sinfo slurm command</description>
<name>yarn.application.hpc.command.slurm.sinfo</name>
<value>sinfo</value>
</property>

<property>
<description>Wait time for checking the resource availability in HPC
Resource scheduler</description>
<name>yarn.application.hpc.client-rs.max-wait.ms</name>
<value>2000</value>
</property>

<property>
<description>No of max retries to HPC Resource scheduler when
```

```
resources not
available</description><name>yarn.application.hpc.client-
rs.retries.max</name><value>100</value>
</property>

<property>
  <description>Application Master resource memory in mb.
      It can be specified differently for each
application.</description>
  <name>yarn.application.hpc.am.resource.mb</name>
  <value>1536</value>
</property>

<property>
  <description>Application Master resource CPU vcores.
      It can be specified differently for each
application.</description>
  <name>yarn.application.hpc.am.resource.cpu-vcores</name>
  <value>1536</value>
</property>
</configuration>
```

Installing and Configuring the Job Scheduler Integration for PBS

This section describes how to install and configure the HPC job scheduler integration with Map Reduce to support PBS.

Prerequisites

The following software must be installed and working correctly:

- Intel® Enterprise Edition for Lustre* software, version 2.2.0.0 or later.
- Apache Hadoop
- The Hadoop* Adapter for Intel® EE for Lustre* Software.
- PBS version 2.5.6 or later

Install the Job Scheduler Integration for PBS

If you are installing on systems running Intel® EE for Lustre* software version 2.4 or later, then *download* the file: *ham-2.6.0-ieel-2.2.jar* or *ham-2.7.0-ieel-2.2.jar* to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore, the folder to download to is: `/usr/lib/hadoop-<version>/share/hadoop/common/lib`.

If you are installing on systems running Intel® EE for Lustre* software version 2.2 or 2.3, then the jar file is already downloded; *copy* the file *ham-2.6.0-ieel-2.2.jar* or *ham-2.7.0-ieel-2.2.jar.* to the Hadoop common library folder. Note that HADOOP_HOME is: /usr/lib/hadoop-<version>. Therefore:

```
#cp /root/install/ieel-<version>/hadoop/ hadoop-hpc-scheduler-
<VERSION>.jar /usr/lib/hadoop-<version>/share/hadoop/common/lib/.
```

Note: This is *not* the same jar file as that used to support SLURM.

Configure the Job Scheduler Integration

Configuration is implemented with the configuration file yarn-site.xml. Following is the only configuration change required in yarn-site.xml to run MapReduce jobs or YARN applications with the adapter:

```
<property> <description>RPC class implementation</description>
<name>yarn.ipc.rpc.class</name>
<value>org.apache.hadoop.yarn.hpc.HadoopYarnHPCRPC</value>
</property>

<property> <description>HPC Application Client class
implementation</description>
<name>yarn.application.hpc.client.class</name>
<value>org.apache.hadoop.yarn.hpc.pbs.PBSApplicationClient</value>
</property>

<property> <description>HPC Application Master class
implementation</description>
<name>yarn.application.hpc.applicationmaster.class</name>
<value>org.apache.hadoop.yarn.hpc.pbs.PBSApplicationMaster</value>
</property>

<property>
<description>HPC Application Container Manager class implementation
</description>
<name>yarn.application.hpc.containermanager.class</name>
<value>org.apache.hadoop.yarn.hpc.pbs.PBSContainerManager</value>
</property>
```

You should now be able to run any MapReduce job just like you do with YARN. You are not required to have any of the YARN daemons (node or resource managers) running.

Additional configuration options

Following are additional are additional configuration options. If the user wants to update any of these properties, those properties need to set in yarn-site.xml.

```
<property>
<description>Local directories for yarn applications in the HPC
environment. If this value is not set or empty, it takes the value
of 'yarn.nodemanager.local-dirs' configuration. </description>
<name>yarn.application.hpc.local.dirs</name> <value></value>
</property>

<property> <description>Logs directories for yarn applications in
the HPC environment. If this value is not set or empty, it takes the
value of 'yarn.nodemanager.log-dirs' configuration. </description>
<name>yarn.application.hpc.log.dirs</name> <value></value>
</property>

<property> <description>PBS command to submit Job</description>
<name>yarn.application.hpc.command.pbs.qsub</name>
<value>qsub</value>
</property>

<property>
<description>PBS Command to cancel Job</description>
<name>yarn.application.hpc.command.pbs.qdel</name>
<value>qdel</value>
</property>

<property>
<description>PBS Command to get Job stats</description>
<name>yarn.application.hpc.command.pbs.qstat</name>
<value>qstat</value>
</property>

<property>
<description>PBS Command to alter job properties</description>
<name>yarn.application.hpc.command.pbs.qalter</name>
<value>qalter</value>
</property>

<property>
<description>PBS Command to get pbs nodes</description>
<name>yarn.application.hpc.command.pbs.pbsnodes</name>
```

```
<value>pbsnodes</value>
</property>
```

Run a MapReduce Job

The following example is a single command.

```
yarn jar ../share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar
wordcount /input /output yarn jar ../share/hadoop/mapreduce/hadoop-
mapreduce-client-jobclient-*-tests.jar sleep -m 10 -r 10 -mt 1000 -
rt 1000
```

Run an YARN application

The following example is a single command.

```
yarn jar ../share/hadoop/yarn/hadoop-yarn-applications-
distributedshell-*.jar -queue testqueue -timeout 10000 -
master_memory 1024 -master_vcores 1 -jar
../share/hadoop/yarn/hadoop-yarn-applications-distributedshell-*.jar
-shell_command ls -num_containers 5
```

Appendix 1 – Troubleshooting

The following checklist is helpful for troubleshooting in the event that a job fails:

- Check that the fs.root.dir directory has appropriate permissions.
- If failure is due to a FileNotFoundException, check that the reported path is using the prefix lustre:/
- A wrong prefix indicates that the framework is not loading the correct file system. Verify the configuration is correct and try restarting the daemons.
- Check that the environment variable JAVA_HOME is set correctly.
- Running hadoop jobs as root is not permitted.

```
# yarn jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples-
2.3.0-cdh5.1.2.jar pi 4 1000
Number of Maps  = 4
Samples per Map = 1000
org.apache.hadoop.security.AccessControlException: Permission
denied: user=root, access=WRITE, inode="/user":hdfs:hadoop:drwxr-xr-
x
```