ORAAH 2.8.2 Installation Guide

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction

This guide explains how to install ORAAH (Oracle R Advanced Analytics for Hadoop), formerly known as ORCH, on a client and on the nodes of the Hadoop cluster. These steps are currently validated on generic Apache Hadoop, Cloudera CDH and on Hortonworks HDP clusters.

2 About Client-Side and Hadoop Cluster-Side Setup

You must install ORAAH within the Hadoop cluster and also on a client external to the hadoop cluster.

2.1 ORAAH Installation on a Client

The client side of ORAAH can be installed on Hadoop cluster edge nodes and/or on client hosts that are outside of the Hadoop cluster. R session runs on the client (Linux only).

2.2 ORAAH Installation on Hadoop

On the entire Hadoop cluster, i.e., all nodes that host a YARN Node Manager, you must install ORAAH server components.

3 Package Changes in Release 2.8.2

Release 2.8.2 supports the ORE release 1.5.1. The previous dependency on the OREmodels package is replaced with a dependency on the OREcommon package. Altogether, ORAAH now depends on five ORE packages. You must install each of these packages on the computer that hosts the ORAAH client as well as on the Hadoop nodes running the YARN Node Manager:

- OREbase
- OREcommon
- OREembed
- OREserver
- OREstats

ORAAH also depends on Intel MKL libraries.These libraries are used in ORAAH advanced statistical functions to improve the performance and precision of statistical computations. The Intel MKL libraries need to be installed on the client node and on every Hadoop node that runs YARN Node Manager. ORAAH includes all required Intel MKL libraries that are automatically deployed on all cluster nodes and made available for ORAAH processes.

Since release ORAAH 2.8.2 also depends on MPI (Message Passing Interface) and it is required to be installed on all nodes of the target cluster. ORAAH includes pre-compiled version of mpich, which will be automatically deployed on all nodes of the the cluster during installation procedure. In case if a different version of MPI libraries is preferred or required it can be manully configured in ORAAH, see more details below.

A full set of required libraries is included with the product distribution, and is already installed if you use the automated install scripts.

4 Downloading ORAAH

Download the ORAAH release 2.8.2 and the supporting packages from the Oracle Technology Network: http://www.oracle.com/-technetwork/database/database-technologies/bdc/r-advanalytics-for-hadoop/downloads/index.html

Start by unzipping both archives into a folder of your choice on the system that will function as the ORAAH client. To complete the installation, you will later do the same on the nodes of the Hadoop cluster.

5 Prerequisites and Verification

ORAAH 2.8.2 client installer tests for some necessary requirements before installing ORAAH packages and other libraries. The scripts "*precheck.sh*" tests for the availability of the following:

1. Oracle R Distribution (ORD) - Version 3.3.0
2. Hadoop - *hadoop* command availability (Users can set the environment variable `ORCH_HADOOP_HOME` or `HADOOP_HOME` if *hadoop* is not in `PATH`)
3. Spark - *spark-submit* tool should be accessible (Users can set the environment variable `ORCH_SPARK_HOME` or `SPARK_HOME` if *spark-submit* is not in `PATH`)

The client installer also performs a number of post-installation tests and reports back to user. The script "*postcheck.sh*" does the following checks:

1. ORAAH packages are loaded successfully in R,
2. Accessibility of HDFS in R.

Note
If you wish to skip the pre-install and post-install checks use the switch *-f* with "*install-client.sh*". For more details check the description of "*install- client.sh*" in the section "Automated Installation on Hadoop Systems other than Oracle Big Data Appliance".

6 Automated installation on Oracle Big Data Appliance (BDA) clusters

ORAAH includes a set of installation and uninstallation scripts that automate the installation, upgrade and uninstallation of the product.

The outline of the installation procedure on Oracle BDA machine clusters is as follows:

- Run "*install-client.sh*" script on the client/edge cluster node(s) per script usage details further in this section;
- Run "*install-server.sh*" script to install ORAAH server components on the compute cluster nodes;
- Perform post installation steps described in the section 12 as applicable;
- Perform R environment adjustments as described in the section 13.

Note
The scripts for automated installation on Hadoop are currently compatible with CDH clusters on Oracle Big Data Appliance (BDA) only. Installation on non-BDA Hadoop clusters requires specification of the cluster node names. For details refer section "Automated Installation on Hadoop Systems other than Oracle Big Data Appliance".

Under the "ORAAH-2.8.2-install" folder, one finds the following scripts:

- **install-client.sh** This script installs the client-side packages and libraries required to run ORAAH platform. Run this script only on Hadoop cluster edge nodes and/or on client hosts that are outside of the cluster that will be used for running R and using ORAAH. Running this script is not required on Hadoop cluster compute nodes, with one exception: when a Hadoop compute node is also used as an edge node for running ORAAH client software. Available options:
  - "-y" - Automatically reply "yes" to all script questions (unattended installation mode).

  - "-f" - Skip pre-installation and post-installation checks.
  - <*filename*> - Specify the path of your Renviron.site configuration file if present at a non-default location (other than "*/usr/lib64/R/etc/Renviron.site*").

- **uninstall-client.sh** This script removes all client side ORAAH packages and libraries. If Oracle R Enterprise (ORE) client is detected then user will be prompted to confirm removal of dependant ORE packages in ORAAH. Available options:
  - "-f" - Force uninstallation to continue even if errors are encountered. In *force* mode all ORAAH packages will be removed from all library paths on the client node. So, if you have multiple copies of ORAAH libraries on your client, they all will be removed.
  - <*filename*> - Specify the path of your Renviron.site configuration file if present at a non-default location (other than "*/usr/lib64/R/etc/Renviron.site*").
- **install-server.sh** This script installs server side packages and libraries required to run ORAAH workloads on every compute node of Hadoop cluster (nodes that are under the management of Hadoop's Node Manager). It requires the "dcli" tool to be available and configured on Oracle's BDA Hadoop cluster. The script must be run only on one of the Hadoop cluster nodes. It will automatically install all the required components on the rest of the BDA cluster. Available options:
  - "-y" - Automatically reply "yes" to all script questions (unattended installation mode).
  - <*filename*> - Although this switch is present, you can ignore it. It is not used with Oracle Big Data Appliance installations.
- **uninstall-server.sh** This script removes server-side ORAAH packages and libraries from every node of the Hadoop cluster (except the client node from where this script is initiated) where they had previously been installed. Like the install-server.sh script described above, this script requires the "dcli" tool. Run the script on only one of the Hadoop cluster nodes. It will automatically uninstall ORAAH components on the rest of the cluster. Available options:
  - "-f" - Force continued uninstallation even if errors are encountered.
  - <*filename*> - Ignored.

Important

If you are installing ORAAH together with Oracle R Enterprise (ORE), then install ORAAH only after installing ORE. If you do the reverse and install ORE after ORAAH, then the ORE installation overrides some of the shared R packages with outdated versions not compatible with ORAAH. This causes the ORAAH Hive transparency layer and some of the shared analytics functionality to misbehave, and results in a runtime error. ORAAH validates component versions during the loading sequence in order to detect any version mismatch or incompatibility and will error out in case if any compatibility issues are found.

7 Automated Installation on Hadoop Systems other than Oracle Big Data Appliance

ORAAH includes a separate set of scripts to automate installation, upgrade, and uninstallation of the product on Hadoop systems other than Oracle Big Data Appliance. For automated installation on Oracle's Big Data Appliance (BDA) clusters please refer to "Automated Installation on Hadoop Systems other than Oracle Big Data Appliance".

Under the "ORAAH-2.8.2-install" folder, find the following scripts:

- **install-client.sh** This script installs the client-side packages and libraries required to run the ORAAH platform. Run this script only on Hadoop cluster edge nodes and/or on a client hosts that are outside of the cluster. Running this script is ordinarily not required on Hadoop cluster compute nodes. The only exception is when a Hadoop compute node is also an edge node for running an ORAAH client. Available options:
  - "-y" - Automatically reply "yes" to all script questions (unattended installation mode).
  - "-f" - Skip pre-installation and post-installation checks.
  - <*filename*> - Specify the path of your Renviron.site configuration file if present at a non-default location (other than "*/usr/lib64/R/etc/Renviron.site*").

- **uninstall-client.sh** This script removes all client side ORAAH packages and libraries. If Oracle R Enterprise (ORE) client is detected then user will be prompted to confirm removal of dependant ORE packages in ORAAH. Available options:
  - "-f" - Force uninstallation to continue even if errors are encountered. In *force* mode all ORAAH packages will be removed from all library paths on the client node. So, if you have multiple copies of ORAAH libraries on your client, they all will be removed.
  - *<filename>* - Specify the path of your Renviron.site configuration file if present at a non-default location (other than "*/usr/lib64/R/etc/Renviron.site*").
- **install-server.sh** This script installs server-side packages and libraries required to run ORAAH workloads on every compute node of Hadoop cluster (nodes that are managed by Hadoop's Node Manager). When this script is executed on a Hadoop system other than Oracle Big Data Appliance, it uses "rsync" and "ssh" tools to distribute packages across the cluster and executes remote commands. Run the script only once on one of the Hadoop cluster nodes. Available options:
  - "-y" - Automatically reply "yes" to all script questions (unattended installation mode).
  - *<filename>* - A plain text file that contains host names of all Hadoop cluster nodes where server-side components must be installed. The file format is one host name per line terminated by a new line symbol. Do not include commas or special characters.

Note

This script expects that passwordless SSH access to the cluster nodes is enabled. For your convenience, an "*extra/keyless-ssh.sh*" script is included with ORAAH installation scripts to simplify passwordless SSH setup.

Important

All nodes that are running YARN Node Manager must be listed in this file or ORAAH jobs may fail randomly (a MapReduce or Spark job might eventually hit a node without the proper configuration).

- **uninstall-server.sh** This script removes server side ORAAH packages and libraries from every node of Hadoop cluster (except the client node from where this script is initiated). When this script is executed on Hadoop systems other than Oracle Big Data Appliance, it uses "rsync" and "ssh" tools to remove packages across the cluster and execute remote commands. The script must be run only once on one of the Hadoop cluster nodes. Note that the scripts expects that passwordless ssh access to the cluster nodes is enabled. For your convenience, an "extra/keyless-ssh.sh" script is included with ORAAH installation scripts to simplify password-less ssh setup. Available options:
  - "-f" - Force uninstallation to continue even if errors are encountered.
  - *<filename>* - A plain text file that contains host names of all Hadoop cluster nodes where server-side components must be installed. The file format is one host name per line, terminated by a new line symbol. Do not include commas or special characters. Note that all nodes that are running YARN Node Manager must be listed in this file for a complete uninstallation.

Important

If you are installing ORAAH together with Oracle R Enterprise (ORE), then you need to make sure that ORAAH is installed after ORE. If you install ORE after ORAAH, it will override some of the shared R packages with outdated versions that are not compatible with ORAAH, thus causing ORAAH Hive transparency layer and some of the shared analytics functionality to misbehave with subsequent runtime errors. ORAAH is validating versions of loaded packages to make sure that they were not overwritten. It will generate a fatal error during loading sequence if any version mismatch or incompatibility is detected.

8 Supporting scripts

ORAAH comes with a set of supporting installation and management scripts that can help an ORAAH administrator automate some common supporting.

- **install-packages.sh** This script installs a set of R packages in bulk on every Oracle Big Data Appliance Hadoop cluster node. This script may be used only on Oracle's BDA Hadoop cluster, and requires the "dcli" tool to be available and configured. The script must be run only once on one of the Hadoop cluster nodes. Available options:
  - "-y" - Automatically reply "yes" to all script questions (unattended installation mode).
  - <*filename*> - List of R packages to be installed. The file format is one file path per line, terminated by a newline character. Do not include commas or special characters.
- **uninstall-packages.sh** This script uninstalls a set of R packages from every Oracle Big Data Appliance Hadoop cluster node where they are installed. This script requires Oracle's "dcli" tool and is therefore only for use on Oracle Big Data Appliance. Run the script only once on one of the Hadoop cluster nodes. Available options:
  - "-f" - Force to continue uninstallation even if errors are encountered.
  - <*filename*> - List of R packages to be uninstalled. The file format is one file path per line, terminated by a newline character. Do not include commas or special characters.
- **keyless-ssh.sh** This script can be used to enable passwordless SSH for a set of Hadoop nodes. Passwordless SSH is currently required for automated installation and uninstallation of ORAAH server-side components on all supported Hadoop systems other than Oracle Big Data Appliance.
  - <*filename*> - A plain text file that contains host names of all Hadoop cluster nodes where server-side components must be installed. The file format is one host name per line, terminated by a newline character. Do not include commas or special characters.

9 Manual installation

When ORAAH is installed on a Hadoop machine cluster other than Oracle Big Data Appliance machine cluster, the automated server side installation/uninstallation scripts may fail for various reasons. If this happens, you can perform the Hadoop side of the installation manually. The automated client-side installation/uninstallation scripts should work on all platforms.

The following steps describe how to manually deploy all server-side components on any Hadoop cluster.

Copy the following files to every node of your Hadoop cluster:

```
OREserver_1.5.1_R_x86_64-pc-linux-gnu.tar.gz
OREcommon_1.5.1_R_x86_64-pc-linux-gnu.tar.gz
mpich_install.zip
mkl/*
lib/*
```

Install R packages on every node of the cluster:

```
R --vanilla CMD INSTALL OREcommon_1.5.1_R_x86_64-pc-linux-gnu.tar.gz
R --vanilla CMD INSTALL OREserver_1.5.1_R_x86_64-pc-linux-gnu.tar.gz
```

Copy the MKL libraries to R's library directory:

```
cp mkl/* /usr/lib64/R/lib
```

Copy the MPI libraries to R's library directory:

```
unzip -q -o mpich_install.zip -d /usr/lib64/R/lib"
```

Copy the ORAAH libraries to R’s library directory:

```
cp lib/* /usr/lib64/R/lib
```

If the "OREserver" and "OREcommon" packages have not been installed on any of the cluster nodes where a mapper/reducer tasks could run, then two of the analytics, namely, "orch.glm" and "orch.lm," will error out. The following error can then be seen in the mapper task log file:

```
Error in loadNamespace(name) : there is no package called ’OREserver’:
Calls: source ... tryCatch -> tryCatchList -> tryCatchOne -> <Anonymous>
In addition: Warning message:
In library(package, lib.loc = lib.loc, character.only = TRUE, logical.return = TRUE, :
there is no package called ’OREserver’
Execution halted
```

If MPI is not installed properly on the driver node, the MPI job (such as `orch.elm`, `orch.helm`, `orch.dspca`, `orch.dssvd`) will fail immediately. If MPI is not installed properly on one or more of worker nodes, MPI worker launch attempts will be failing on that node and potentially retried by a resource manager. If same partition fails to start MPI worker more than the resource manager configured number of attempts, the whole job will be rejected.

It is not advisable to attempt MPI jobs with partial MPI availability throughout the cluster, as the probability of at least one MPI stage failure within computation significantly increases with the number of MPI stage submissions by an algorithm. If at least one MPI stage fails, the whole algorithm fails and may attempt again as configured with `orch.mpi.options()`.

10 Installing ORAAH With a Non-Oracle R Distribution

If you choose to use an alternative to Oracle’s R distribution, then install one additional library — "*libOrdBlasLoader.so*". This library is required by ORAAH’s statistical packages. If this library is not installed, then you will encounter failures when running most ORAAH analytics.

To install the library, copy the library file libOrdBlasLoader.so, which is included in ORAAH distribution zip file, into one of the directories listed in `LD_LIBRARY_PATH` or (the preferred way) into R’s "lib" directory (which by default is "*/usr/lib64/R/lib*").

```
cp libOrdBlasLoader.so /usr/lib64/R/lib
```

The same should be done on every node of Hadoop cluster where R is installed:

```
dcli -C -f libOrdBlasLoader.so -d /tmp/ libOrdBlasLoader.so
dcli -C "cp /tmp/libOrdBlasLoader.so /usr/lib64/R/lib"
```

Note
Note that you need root privileges to install this library.

If you use Oracle’s R distribution (available via http://public-yum.oracle.com/ for OEL5 and OEL6), then this additional installation is not required. The libOrdBlasLoader.so library is already included with the Oracle R distribution.

11 Other Dependencies

ORAAH has a dependency on the "rJava" and the "RJDBC" R packages. Both packages come in the "ORAAH-2.8.2-Supporting" zip file and are required on the ORAAH client host only (but not on the Hadoop cluster compute nodes) .

These packages should be installed automatically by the installation scripts, but if the scripts are not executed or fail, then you must install these packages manually.

You can download the packages from CRAN:

- rJava: https://cran.r-project.org/web/packages/rJava/index.html
- RJDBC: https://cran.r-project.org/web/packages/RJDBC/index.html

If needed, install them on the client node as follows:

```
R CMD javareconf
R CMD INSTALL rJava_0.9-9_R_x86_64-pc-linux-gnu.tar.gz
R CMD INSTALL RJDBC_0.2-6_R_x86_64-pc-linux-gnu.tar.gz
```

ORAAH also has an indirect dependency on the "png" and "DBI" packages. As with other required dependencies, the packages should be automatically installed from "ORAAH-2.8.1-Supporting" zip file; but if not, they must be installed manually. You can also download these packages from CRAN:

- png: http://cran.r-project.org/web/packages/png/index.html
- DBI: https://cran.r-project.org/web/packages/DBI/index.html

Install these packages on the ORAAH client node:

```
R CMD INSTALL png_0.1-7_R_x86_64-pc-linux-gnu.tar.gz
R CMD INSTALL DBI_0.6-1_R_x86_64-pc-linux-gnu.tar.gz
```

12 Post-Installation Steps – Setting ORAAH Configuration Variables for the Environment

ORAAH works with a wide variety of Hadoop distributions and versions, but it needs to "know" how to interact with the particular Hadoop distribution (and the specific version of the distribution) that you are using.

Important

Even when library(ORCH) loads successfully, the configuration is not complete. To fully connect ORAAH to HDFS, HIVE, and Spark on the Hadoop cluster, it is critical that after the installation you set all of the ORAAH environment variables described in this section. These variables must be set using a method that makes them available to all user R sessions.

There are environment variables that override correspondent Hadoop native environment values that specify Hadoop's component home path. For example, if both HADOOP_HOME and ORCH_HADOOP_HOME environment variables are defined, then ORAAH prioritizes the use of the ORCH_HADOOP_HOME variable.

Errors Indicating that ORAAH Requires More Configuration to Work With Your Hadoop Environment:

The installation automatically configures ORAAH's Hadoop Abstraction Layer (HAL) to work with most versions of the Cloudera Distribution of Hadoop (CDH), although ORAAH connectivity to Cloudera Hive and Spark does require some manual configuration steps.

However, ORAAH's ability to auto-detect the Hadoop distribution and self-configure accordingly does not yet extend to other other Hadoop distributions, such as Hortonworks HDP, MapR, Apache. If are using one of these, or your own custom-built Hadoop environment, then at startup you will likely encounter an error indicating that ORAAH does not recognize some feature of the environment. Errors of this type can also occur with some versions of CDH. Here are some examples output logs for specifically 2.8.0 release version.

```
Oracle R Connector for Hadoop 2.8.0
Info: using native C base64 encoding implementation
Info: Hadoop distribution is unknown
Error: unsupported version 2.2.0-cdh5.0.0-beta-2 of Hadoop
Info: use "ORCH_HAL_VERSION" envvar to define HAL version
DBG: 22:41:22 [FA] HAL was not initialized
```

```
Error : .onLoad failed in loadNamespace() for 'ORCHcore', details:
call: NULL
error: execution aborted
Error: package â could not be loaded
```

or:

```
Oracle R Connector for Hadoop 2.8.0
Info: using native C base64 encoding implementation
Info: Hadoop distribution is Cloudera's CDH v5.0.0
Info: using auto-detected ORCH HAL v4.2
Info: HDFS workdir is set to "/user/oracle"
Error: unrecognized response from Hadoop
Error: mapReduce is not ready, hadoop.*() may fail
Info: HDFS is functional
Error: Failed to connect to Hadoop cluster.
Loading required package: ORCHstats
<...>
```

These errors indicate that ORAAH does not know how to interact this particular Hadoop distribution, and must be manually configured using special OS environment variables.

How to Set Up Persistent ORAAH Environment Variables:

There are many options to preserve the configuration between R sessions. The preferred method is to put the configuration variables into "*/usr/lib64/R/etc/Renviron.site*" as root. This enables any R session to pick up environmental variables.

Note
This file does not exist by default in R (do not confuse it with the existing "*/usr/lib64/R/etc/Renviron*").

Other approaches to ensure persistent environment include:

1. Store variables in the user's startup scripts (under .cshrc, .bashrc or .profile, depending on the OS and shell interpreter). The variables are then loaded each time a user starts an R session.
2. Use Sys.setenv() R base functions during an R session to configure the variables before loading ORAAH.

The sections that follow list the most important ORAAH configuration variables. For the complete list of ORAAH configuration options use `help("ORCH-config")` after loading "ORCHcore" package.

12.1 ORCH_HADOOP_HOME

This ORAAH environment variable enables you to override auto-detection of a Hadoop home path, or the home path of Hadoop components used by ORAAH. Set it before starting R and before loading the ORCH library.

Important
Setting this environment overrides any Hadoop native environment values that specify Hadoop's component home path. For example, if both the HADOOP_HOME and ORCH_HADOOP_HOME environment variables are set, then ORAAH prioritize use of ORCH_HADOOP_HOME.

Other supported Hadoop components home environment variables are:

- `ORCH_HDFS_HOME`
- `ORCH_HIVE_HOME`
- `ORCH_MAHOUT_HOME`
- `ORCH_SQOOP_HOME`
- `ORCH_OLH_HOME`

12.2 ORCH_HAL_VERSION

ORCH_HAL_VERSION overrides auto-detection of the Hadoop version, and forces use of a version of the ORAAH Hadoop Abstraction Layer that you specify. You can set this ORAAH environment variable before starting R and loading the ORCH library.

Supported versions are:

- **1** = Apache/IDC/Hortonworks 1.*
- **2** = Cloudera CDH3u*
- **3** = Cloudera CDH4.* with MR1
- **4** = Cloudera CDH4.[0-3] with MR1
- **4.1** = Cloudera CDH4.4 with MR1
- **4.2** = Cloudera CDH5.x with MR2 or Hortonworks 2.x

How Auto-Detection Works:

- If ORCH_HAL_VERSION is set:
  1. If ORAAH auto-detection cannot identify the Hadoop version in the environment, then ORAAH applies the user-selected HAL and upon load of the ORCH library displays a message stating the ORCH_HAL_VERSION in use.
  2. If auto-detection does identify the Hadoop version in the environment, and determines that it is not consistent with ORCH_HAL_VERSION, then a warning message is issued upon loading the ORCH library. However, the load proceeds, and the version specified by ORCH_HAL_VERSION is used.
- If ORCH_HAL_VERSION is not set (the default), then auto-detection either succeeds or fails in selecting a matching HAL. If ORAAH cannot identify the Hadoop distribution or version, then it issues an error message and remains in an error state (not initialized). This state prevents HDFS and mapReduce operations from functioning correctly. You must unload ORAAH, set the correct value of ORCH_VAL_VERSION, and reload ORAAH.

Note
If ORCH_HAL_VERSION is set to an invalid value, then an error message is issued when loading ORAAH, and the value is ignored. ORAAH will continue to operate as if the variable was not set. You can unload ORAAH, set the correct value of ORCH_VAL_VERSION, and reload ORAAH in order to correct this.

Note
You can override the HAL version when you are testing ORAAH against a new Hadoop distribution. In this case, ORAAH loads and initializes, but you may encounter failures when invoking ORAAH API functions. ORAAH does not provide any functional guarantees as this case.

12.3 ORCH_JAR_MR_VERSION

ORCH_JAR_MR_VERSION overrides auto-detection of a Hadoop mapReduce API version and specifies use of an appropriate version of the ORAAH Hadoop JAR library.

Set this ORAAH environment variable before starting R and loading the ORCH library.

Settings for supported versions are:

- **1** = mapReducer v1.
- **2** = mapReducer v2, aka YARN.

If ORAAH cannot auto-detect the Hadoop version and HAL, then the mapReduce version default to version 2.

12.4 ORCH_STREAMING_LIB

ORCH_STREAMING_LIB overrides auto-detection of Hadoop's Streaming Java library path and lets you specify a custom path to the streaming JAR file. The path should be specified including the library file name.

Set this ORAAH environment variable before starting R and loading the ORCH library.

Important
Attention. Setting this environment variable will override any Hadoop native environment value that specify streaming library path. For example, if both HADOOP_STREAMING_JAR and ORCH_STERAMING_LIB environment variables are set, then ORAAH uses its own ORCH_HADOOP_HOME variable.

12.5 ORCH_CLASSPATH

This ORAAH environment variable enables you to override CLASSPATH environment variable. You can set the CLASSPATH used by ORAAH using this variable. Set it before starting R and before loading the ORCH library.

"rJava" does not support wildcard characters in CLASSPATH environment variable. Which forced users to add exact list of jar libraries files in the CLASSPATH before they could use Apache Spark analytics. To make configuration easier, ORAAH library loader supports wildcard characters in CLASSPATH. For example, having a path "*/usr/lib/hadoop/lib/*.jar*" in CLASSPATH adds all jars from "*/usr/lib/hadoop/lib*" to rJava JVM's CLASSPATH. This makes the process of getting started with ORAAH's Spark features easier and faster. You can use wildcards with ORCH_CLASSPATH too, if you are going to use ORCH_CLASSPATH instead of CLASSPATH for ORAAH configuration.

Important
Setting this environment variable overrides default CLASSPATH environment value. So, if both ORCH_CLASSPATH and CLASSPATH environment variables are set, then ORAAH prioritize use of ORCH_CLASSPATH.

12.6 ORCH_MPI_LIBS and ORCH_MPI_MPIEXEC

In ORAAH 2.8.2 we have introduced a new package "ORCHmpi". This new package has distributed MPI-backed algorithms which run over the Apache Spark framework.

"ORCHmpi" needs MPI libraries made available to ORAAH either by making MPI available system-wide, or by setting ORCH related environment variables on the client node. For more information on setting up MPI, check help(ORCH_MPI_LIBS) and help(ORCH_MPI_MPIEXEC). If MPI is configured properly, orch.mpiAvailable() and orch.scalapackAvailable() functions will return TRUE.

Another environment variable ORCH_MPI_MAX_GRID_SIZE is used to specify the maximum number of MPI workers (not counting the leader process) that MPI computation may spawn on the cluster per submission. It is recommended to set this to an integer value, with maximum being no more than 60 percent of available cluster CPU cores.

Important
The new ORAAH server installer ("*install-server.sh*") copies a pre-built MPI package *mpich-3.3a2* on the cluster and adds this default configuration to your "*Renviron.site*". However, it was pre-built for Oracle Linux 6 and if your cluster is based on Oracle Linux 7, we recommend you build MPI libraries for your cluster and make it available to ORAAH as described above.

12.7 ORCH_JAVA_XMX, ORCH_JAVA_XMS and ORCH_JAVA_MAX_PERM

You can set these ORCH environment variable before starting R and loading the ORCH library. It enables you to set the flag "-Xmx", "-Xms" and "-XX:MaxPermSiz" for the ORAAH client's Java Virtual Machine (JVM).

ORCH_JAVA_XMX flag specifies the maximum memory allocation pool for a JVM, which means that your JVM will be able to use a maximum of Xmx amount of memory. The default value of this flag for ORAAH is 1GB.

ORCH_JAVA_XMS flag specifies the initial memory allocation pool for a JVM, which means that your JVM will be able to use an initial size of Xms amount of memory. The default value of this flag for ORAAH is 256 MB.

ORCH_JAVA_MAX_PERM flag specifies the size for Permanent Generation, which is is where the classes, methods, internalized strings, and similar objects used by the JVM are stored. The default value of this flag for ORAAH is 256MB.

This memory flag can be specified in multiple sizes, such as kilobytes (k), megabytes (m), gigabytes (g) and so on. See ORAAH reference manual for more details and examples.

13 "*Renviron.site*" file in $R_HOME/etc.

R loads environment variables from a several different files before the R Session is started. For a server with the expected multi-user sessions connecting to R, creating and using Renviron.site is recommended.

How R searches for Environment Settings Files

R searches for site and user files to process for setting environment variables. The environment variable R_ENVIRON points to the site file. If R_ENVIRON is not set, then "*$R_HOME/etc/Renviron.site*" is used, if it exists (In fact, Renviron.site does not exist in a "factory-fresh" installation). The name of the user-specific environment file can also be specified by the R_ENVIRON_USER environment variable. If this is not set, the files searched for are "*.Renviron*" in the current directory, and then the same file in the user's home directory (in that order).

There is also a file "*$R_HOME/etc/Renviron*", which is read very early in the start-up processing. It contains environment variables set by R during the configuration process. Values in that file can be overridden in site or user environment files. Do not change "*$R_HOME/etc/Renviron*" itself. Note that this is distinct from "*$R_HOME/etc/Renviron.site*".

Creating Renviron.site

The recommendation is to create a new Renviron.site file with settings that are related to ORAAH, its HIVE and Spark configuration requirements, and other environmental variables that might be needed if using ORAAH in a mixed Oracle Database environment (or when an Oracle Database Client it also configured on the same node).

This file should be put into "*$R_HOME/etc*", or at the default path, or "*/usr/lib64/R/etc/Renviron.site*".

The following examples show sample Renviron.site files for a Cloudera Distribution of Hadoop cluster, a Hortonworks Distribution of Hadoop Cluster, and Oracle's Big Data Lite Virtual machine. In the case of the Cloudera cluster, the installation is assuming that Parcels are used. In the case of the Hortonworks cluster, it assumes that Apache Ambari was used for setup. In the case of the Oracle Big Data Lite VM, it was configured not using parcels.

Important
Carefully inspect each folder and file location. Depending on the release of CDH, HDP or other Hadoop cluster release, you might have to search for the appropriate files. Also, the correct name of the file is Renviron.site with a capital R. Finally, this file needs to exist in each node that has R installed in the cluster.

14 Appendix 1: Examples of "*Renviron.site*"

Below you can find example of "Renviron.site" file taken from a "live" setup of ORAAH on different Hadoop distribution. Please use it for refence or as a base template when setting up ORAAH on your cluster.

14.1 Cloudera Distribution of Hadoop 5.14.2.

```
############### HOME DIR's################
# If you have an Oracle CLient Configured:
# ORACLE_HOME=/usr/lib/oracle/12.1/client64
# ORACLE_HOSTNAME=localhost

CDH_VERSION=5.14.2

# Optional Settings to skip verification of HDFS and Map Reduce functionality.
# Recommended to be set to 0 (default is 1 if not set) after the configuration
# has been successful to speed up initialization.
ORCH_HDFS_CHECK=0
ORCH_MAPRED_CHECK=0

############### HOME DIRs #################
R_HOME=/usr/lib64/R
JAVA_HOME=/usr/java/default
HADOOP_HOME=/opt/cloudera/parcels/CDH/lib/hadoop
HIVE_HOME=/opt/cloudera/parcels/CDH/lib/hive
HADOOP_MAPRED_HOME=/opt/cloudera/parcels/CDH/lib/hadoop-mapreduce
SQOOP_HOME=/opt/cloudera/parcels/CDH/lib/sqoop
PIG_HOME=/opt/cloudera/parcels/CDH/lib/pig
IMPALA_HOME=/opt/cloudera/parcels/CDH/lib/impala
YARN_HOME=/opt/cloudera/parcels/CDH/lib/hadoop-yarn
SPARK_HOME=/opt/cloudera/parcels/SPARK2/lib/spark2

# OPTIONAL: Point to Impala JDBC driver version 4 client library directory.
# NOTE: These libaries should be present on all nodes of the cluster if you wish
# to use Impala tables as input to Spark analytics in ORAAH. If they are present
# only on your client node, you can still connect to Impala and perform table
# operations.
# IMPALA_JAR=/opt/example/impalajdbc

# OPTIONAL: pointers to folders for different Oracle Big Data Appliance or
# Oracle Big Data Connectors and Oracle NoSQL components. Depending on the
# folders the products are installed.
OLH_HOME=/opt/oracle/oraloader-3.4.0-h2
KVHOME=/u01/nosql/kv-ee
OSCH_HOME=/u01/connectors/osch
COPY2BDA_HOME=/u01/orahivedp

########## CONF DIRs #############
HADOOP_CONF_DIR=/etc/hadoop/conf
HIVE_CONF_DIR=/opt/cloudera/parcels/CDH/lib/hive/conf
SPARK_CONF_DIR=/etc/spark2/conf

ORCH_STREAMING_LIB=/opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar

LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib:/usr/lib64/R/lib:/usr/lib64/R/library/ ↩
    rJava:/usr/lib64/R/library/RImpala

HADOOP_CLASSPATH=$COPY2BDA_HOME/jlib/*:$OLH_HOME/jlib/*:$HIVE_CONF_DIR:$OSCH_HOME/jlib/*: ↩
    $ORACLE_HOME/jdbc/lib/*:$KVHOME/lib/kvstore.jar:/usr/lib/hive-hcatalog/share/hcatalog/ ↩
    hive-hcatalog-core.jar

PATH=/usr/lib64/qt-3.3/bin:/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/ ↩
    bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/oracle/bda/bin:/usr/lib/oracle/12.1/client64/ ↩
    bin:/usr/java/default/bin:/home/oracle/bin:$ORCH_HADOOP_HOME:$ORCH_STREAMING_LIB: ↩
    $SQOOP_HOME:$OLH_HOME:$SPARK_HOME:$SPARK_JAVA_OPTS:$HADOOP_HOME
```

```
HADOOP_JARS=${HADOOP_HOME}/hadoop-common.jar:${HADOOP_HOME}/hadoop-auth.jar:${HADOOP_HOME}/ ↩
    client/hadoop-hdfs.jar:${HADOOP_HOME}/client/slf4j-api.jar:${HADOOP_HOME}/client/slf4j- ↩
    log4j12.jar:${HADOOP_HOME}/client/log4j.jar:${HADOOP_HOME}/client/guava.jar:${ ↩
    HADOOP_HOME}/client/commons-collections.jar:${HADOOP_HOME}/client/commons-configuration. ↩
    jar:${HADOOP_HOME}/client/commons-lang.jar:${HADOOP_HOME}/client/commons-cli.jar:${ ↩
    HADOOP_HOME}/client/commons-logging.jar:${HADOOP_HOME}/client/commons-io.jar:${ ↩
    HADOOP_HOME}/client/htrace-core4.jar:${HADOOP_HOME}/client/protobuf-java.jar:${ ↩
    HADOOP_HOME}/client/snappy-java.jar:${HADOOP_HOME}/client/hadoop-mapreduce-client-common ↩
    .jar:${HADOOP_HOME}/client/hadoop-mapreduce-client-app.jar:${HADOOP_HOME}/client/hadoop- ↩
    mapreduce-client-core.jar:${HADOOP_HOME}/client/hadoop-mapreduce-client-shuffle.jar:${ ↩
    HADOOP_HOME}/client/hadoop-mapreduce-client-jobclient.jar:${HADOOP_HOME}/client/hadoop- ↩
    yarn-api.jar:${HADOOP_HOME}/client/hadoop-yarn-common.jar:${HADOOP_HOME}/client/hadoop- ↩
    yarn-client.jar:${HADOOP_HOME}/client-0.20/jackson-core-asl.jar:${HADOOP_HOME}/client ↩
    -0.20/jackson-mapper-asl.jar

SPARK2_JARS=${SPARK_HOME}/jars/*.jar

HIVE_JARS=${HIVE_HOME}/lib/hive-jdbc*.jar:${HIVE_HOME}/lib/hive-exec*.jar:${HIVE_HOME}/lib/ ↩
    hive-service*.jar:${HIVE_HOME}/lib/libfb*.jar:${HIVE_HOME}/lib/libthrift*.jar:${ ↩
    HIVE_HOME}/lib/log4j-*.jar:${HIVE_HOME}/lib/zookeeper*.jar:${HIVE_HOME}/lib/httpclient*. ↩
    jar:${HIVE_HOME}/lib/httpcore*.jar:${HIVE_HOME}/lib/hive-shims-common.jar

CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:${ ↩
    SPARK2_JARS}:${HIVE_JARS}

SPARK_DIST_CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:$ ↩
    {IMPALA_JAR}/*

# Paths to your MPI installation.
# Note: By default they are set by ORAAH client installer to point to pre-built
# mpich 3.3a2 libraries which are distributed with ORAAH as part of server
# installation.
ORCH_MPI_LIBS=/usr/lib64/R/lib/mpich33a2/lib
ORCH_MPI_MPIEXEC=/usr/lib64/R/lib/mpich33a2/bin/mpiexec

# END OF Renviron.site
```

14.2 Hortonworks Distribution of Hadoop 2.6.4.0-91.

```
# Set the correct HAL version for Hortonworks 2.6.4
ORCH_HAL_VERSION=4.2

############### HOME DIRs #################
# If using and Oracle Database Client
# ORACLE_HOME=/usr/lib/oracle/12.1/client64
R_HOME=/usr/lib64/R
JAVA_HOME=/usr/lib/jvm/java
HADOOP_HOME=/usr/hdp/current/hadoop-client
HADOOP_MAPRED_HOME=/usr/hdp/current/hadoop-mapreduce-client
SQOOP_HOME=/usr/hdp/current/sqoop-client
HIVE_HOME=/usr/hdp/current/hive-client
YARN_HOME=/usr/hdp/current/hadoop-yarn-client
SPARK_HOME=/usr/hdp/current/spark2-client

# OPTIONAL: Point to Impala JDBC driver version 4 client library directory.
# NOTE: These libaries should be present on all nodes of the cluster if you wish
# to use Impala tables as input to Spark analytics in ORAAH. If they are present
# only on your client node, you can still connect to Impala and perform table
# operations.
# IMPALA_JAR=/opt/example/impalajdbc
# Optional Settings to skip verification of HDFS and Map Reduce functionality.
# Recommended to be set to 0 (default is 1 if not set) after the configuration
# has been successfull to speed up initialization.
ORCH_HDFS_CHECK=0
ORCH_MAPRED_CHECK=0

########## CONF DIRs #############
HADOOP_CONF_DIR=/etc/hadoop/conf
HIVE_CONF_DIR=/etc/hive/conf
SPARK_CONF_DIR=${SPARK_HOME}/conf

LD_LIBRARY_PATH=/usr/lib64/R/lib:/usr/lib64/R/library/rJava

ORCH_STREAMING_LIB=${HADOOP_MAPRED_HOME}/hadoop-streaming.jar

HADOOP_CLASSPATH=${HIVE_CONF_DIR}

PATH=/usr/lib64/qt-3.3/bin:/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/ ↩
    bin:/usr/local/sbin:/usr/sbin:/sbin:${JAVA_HOME}/bin:${SQOOP_HOME}:${SPARK_HOME}/bin:${ ↩
    HADOOP_HOME}/bin

HADOOP_JARS=${HADOOP_HOME}/hadoop-common.jar:${HADOOP_HOME}/hadoop-auth.jar:${HADOOP_HOME ↩
    }/../hadoop-hdfs/hadoop-hdfs.jar:${HADOOP_HOME}/../hadoop-hdfs-client/hadoop-nfs.jar:${ ↩
    HADOOP_HOME}/lib/slf4j-api-1.7.10.jar:${HADOOP_HOME}/lib/slf4j-log4j12-1.7.10.jar:${ ↩
    HADOOP_HOME}/lib/log4j-1.2.17.jar:${HADOOP_HOME}/lib/guava-11.0.2.jar:${HADOOP_HOME}/lib ↩
    /commons-collections-3.2.2.jar:${HADOOP_HOME}/lib/commons-configuration-1.6.jar:${ ↩
    HADOOP_HOME}/lib/commons-lang-2.6.jar:${HADOOP_HOME}/lib/commons-cli-1.2.jar:${ ↩
    HADOOP_HOME}/lib/commons-logging-1.1.3.jar:${HADOOP_HOME}/client/commons-io-2.4.jar:${ ↩
    HADOOP_HOME}/client/htrace-core-3.1.0-incubating.jar:${HADOOP_HOME}/lib/protobuf-java ↩
    -2.5.0.jar:${HADOOP_HOME}/lib/snappy-java-1.0.4.1.jar:${HADOOP_MAPRED_HOME}/hadoop- ↩
    mapreduce-client-common.jar:${HADOOP_MAPRED_HOME}/hadoop-mapreduce-client-app.jar:${ ↩
    HADOOP_MAPRED_HOME}/hadoop-mapreduce-client-core.jar:${HADOOP_MAPRED_HOME}/hadoop- ↩
    mapreduce-client-shuffle.jar:${HADOOP_MAPRED_HOME}/hadoop-mapreduce-client-jobclient.jar ↩
    :${HADOOP_HOME}/../hadoop-yarn/hadoop-yarn-api.jar:${YARN_HOME}/hadoop-yarn-common.jar:$ ↩
    {YARN_HOME}/hadoop-yarn-client.jar:${HADOOP_HOME}/lib/jackson-core-asl-1.9.13.jar:${ ↩
    HADOOP_HOME}/client-0.20/jackson-mapper-asl-1.9.13.jar

SPARK2_JARS=${SPARK_HOME}/jars/*.jar

HIVE_JARS=${HIVE_HOME}/lib/hive-jdbc.jar:${HIVE_HOME}/lib/hive-exec.jar:${HIVE_HOME}/lib/ ↩
    hive-service.jar:${HIVE_HOME}/lib/libfb*.jar:${HIVE_HOME}/lib/libthrift*.jar:${HIVE_HOME ↩
    }/lib/log4j-*.jar:${HIVE_HOME}/lib/zookeeper*.jar:${HIVE_HOME}/lib/httpclient*.jar:${ ↩
    HIVE_HOME}/lib/httpcore*.jar:${HIVE_HOME}/lib/hive-shims-common.jar

CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:${ ↩
    SPARK2_JARS}:${HIVE_JARS}

SPARK_DIST_CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:$ ↩
    {IMPALA_JAR}/*

# Paths to your MPI installation.
# Note: By default they are set by ORAAH client installer to point to pre-built
# mpich 3.3a2 libraries which are distributed with ORAAH as part of server
# installation.
ORCH_MPI_LIBS=/usr/lib64/R/lib/mpich33a2/lib
ORCH_MPI_MPIEXEC=/usr/lib64/R/lib/mpich33a2/bin/mpiexec

# END OF Renviron.site
```

14.3 Oracle’s Big Data Lite Virtual Machine 4.5.0.

Note
Update your SPARK_HOME and SPARK_CONF_DIR for Spark 2.2+.

```
# Required settings for the Oracle Database and Oracle R Enterprise
TZ=EDT
ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
ORACLE_SID=cdb
ORACLE_HOSTNAME=localhost

CDH_VERSION=5.7.0

############### HOME DIRs #################
# Configuration for the Oracle Loader for Hadoop
SPARK_HOME=/usr/lib/spark
HADOOP_HOME=/usr/lib/hadoop
OLH_HOME=/u01/connectors/olh
SQOOP_HOME=/usr/lib/sqoop
JAVA_HOME=/usr/java/latest
HIVE_HOME=/usr/lib/hive
R_HOME=/usr/lib64/R
KVHOME=/u01/nosql/kv-ee
OSCH_HOME=/u01/connectors/osch
COPY2BDA_HOME=/u01/orahivedp

########## CONF DIRs #############
HADOOP_CONF_DIR=/etc/hadoop/conf
HIVE_CONF_DIR=/etc/hive/conf
SPARK_CONF_DIR=/etc/spark2/conf

# Optional Settings to skip verification of HDFS and Map Reduce functionality.
# Recommended to be set to 0 after the configuration has been successfull to
# speed up initialization
ORCH_HDFS_CHECK=0
ORCH_MAPRED_CHECK=0

LD_LIBRARY_PATH=/usr/java/latest/jre/lib/amd64/server:/u01/app/oracle/product/12.1.0.2/ ↩
    dbhome_1/lib:/usr/lib64/R/lib:/usr/lib/hadoop/lib/native

PATH=/usr/lib64/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/ ↩
    usr/lib64/R/bin:/u01/Middleware/jdeveloper/jdev/bin:/usr/java/latest/bin:/u01/app/oracle ↩
    /product/12.1.0.2/dbhome_1/bin:/home/oracle/scripts:/opt/bin:/u01/sqlcl/bin:/home/oracle ↩
    /bin:/usr/lib64/R/bin:/u01/Middleware/jdeveloper/jdev/bin:/usr/java/latest/bin:/u01/app/ ↩
    oracle/product/12.1.0.2/dbhome_1/bin:/home/oracle/scripts:/opt/bin

# OPTIONAL: Point to Impala JDBC driver version 4 client library directory.
# IMPALA_JAR=/opt/example/impalajdbc

ORCH_STREAMING_LIB=/usr/lib/hadoop-mapreduce/hadoop-streaming.jar

HADOOP_CLASSPATH=/u01/orahivedp/jlib/*:/u01/connectors/olh/jlib/*:/etc/hive/conf:/u01/ ↩
    connectors/osch/jlib/*:/u01/app/oracle/product/12.1.0.2/dbhome_1/jdbc/lib/*:/u01/nosql/ ↩
    kv-ee/lib/kvstore.jar:/usr/lib/hive-hcatalog/share/hcatalog/hive-hcatalog-core.jar

HADOOP_JARS=${HADOOP_HOME}/hadoop-common.jar:${HADOOP_HOME}/hadoop-auth.jar:${HADOOP_HOME ↩
    }/../hadoop-hdfs/hadoop-hdfs.jar:${HADOOP_HOME}/../hadoop-hdfs/hadoop-hdfs-nfs.jar:${ ↩
    HADOOP_HOME}/lib/slf4j-api-1.7.10.jar:${HADOOP_HOME}/lib/slf4j-log4j12-1.7.10.jar:${ ↩
    HADOOP_HOME}/lib/log4j-1.2.17.jar:${HADOOP_HOME}/lib/guava-11.0.2.jar:${HADOOP_HOME}/lib ↩
    /commons-collections-3.2.2.jar:${HADOOP_HOME}/lib/commons-configuration-1.6.jar:${ ↩
    HADOOP_HOME}/lib/commons-lang-2.6.jar:${HADOOP_HOME}/lib/commons-cli-1.2.jar:${ ↩
    HADOOP_HOME}/lib/commons-logging-1.1.3.jar:${HADOOP_HOME}/client/commons-io.jar:${ ↩
    HADOOP_HOME}/client/htrace-core4.jar:${HADOOP_HOME}/lib/protobuf-java-2.5.0.jar:${ ↩
    HADOOP_HOME}/lib/snappy-java-1.0.4.1.jar:${HADOOP_HOME}/../hadoop-mapreduce/hadoop- ↩
    mapreduce-client-common.jar:${HADOOP_HOME}/../hadoop-mapreduce/hadoop-mapreduce-client- ↩
    app.jar:${HADOOP_HOME}/../hadoop-mapreduce/hadoop-mapreduce-client-core.jar:${ ↩
    HADOOP_HOME}/../hadoop-mapreduce/hadoop-mapreduce-client-shuffle.jar:${HADOOP_HOME}/../ ↩
    hadoop-mapreduce/hadoop-mapreduce-client-jobclient.jar:${HADOOP_HOME}/../hadoop-yarn/ ↩
    hadoop-yarn-api.jar:${HADOOP_HOME}/../hadoop-yarn/hadoop-yarn-common.jar:${HADOOP_HOME ↩
    }/../hadoop-yarn/hadoop-yarn-client.jar:${HADOOP_HOME}/client-0.20/jackson-core-asl.jar: ↩
    ${HADOOP_HOME}/client-0.20/jackson-mapper-asl.jar

SPARK2_JARS=${SPARK_HOME}/jars/*.jar

HIVE_JARS=${HIVE_HOME}/lib/hive-jdbc*.jar:${HIVE_HOME}/lib/hive-exec*.jar:${HIVE_HOME}/lib/ ↩
    hive-service*.jar:${HIVE_HOME}/lib/libfb*.jar:${HIVE_HOME}/lib/libthrift*.jar:${ ↩
    HIVE_HOME}/lib/log4j-*.jar:${HIVE_HOME}/lib/zookeeper*.jar:${HIVE_HOME}/lib/httpclient*. ↩
    jar:${HIVE_HOME}/lib/httpcore*.jar:${HIVE_HOME}/lib/hive-shims-common.jar

CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:${ ↩
    SPARK2_JARS}:${HIVE_JARS}

SPARK_DIST_CLASSPATH=${SPARK_CONF_DIR}:${HADOOP_CONF_DIR}:${HIVE_CONF_DIR}:${HADOOP_JARS}:$ ↩
    {IMPALA_JAR}/*

# Paths to your MPI installation.
# Note: By default they are set by ORAAH client installer to point to pre-built
# mpich 3.3a2 libraries which are distributed with ORAAH as part of server
# installation.
ORCH_MPI_LIBS=/usr/lib64/R/lib/mpich33a2/lib
ORCH_MPI_MPIEXEC=/usr/lib64/R/lib/mpich33a2/bin/mpiexec

# END OF Renviron.site
```

15 Copyright Notice