



RecordService



# RecordService



## Column-level Security

RecordService provides an abstraction layer between compute frameworks and data storage. It provides row- and column-level security, and other advantages.

## Storage Agnostic Clients

Clients work independently of on-disk storage format. Components swap transparently above or below RecordService.

## Impala Quality Performance

Performance is improved through the optimized scanner, dynamic code generation, and Parquet implementation provided by Impala. Existing MapReduce and Spark jobs gain Impala-quality performance.

## Try RecordService For Yourself

Use the [QuickStart VM](#) to take RecordService for a spin.



[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



# RecordService Overview

RecordService provides an abstraction layer between compute frameworks and data storage. It provides row- and column-level security, and other advantages.

## Topics

The Motivation

The Benefits

The Design

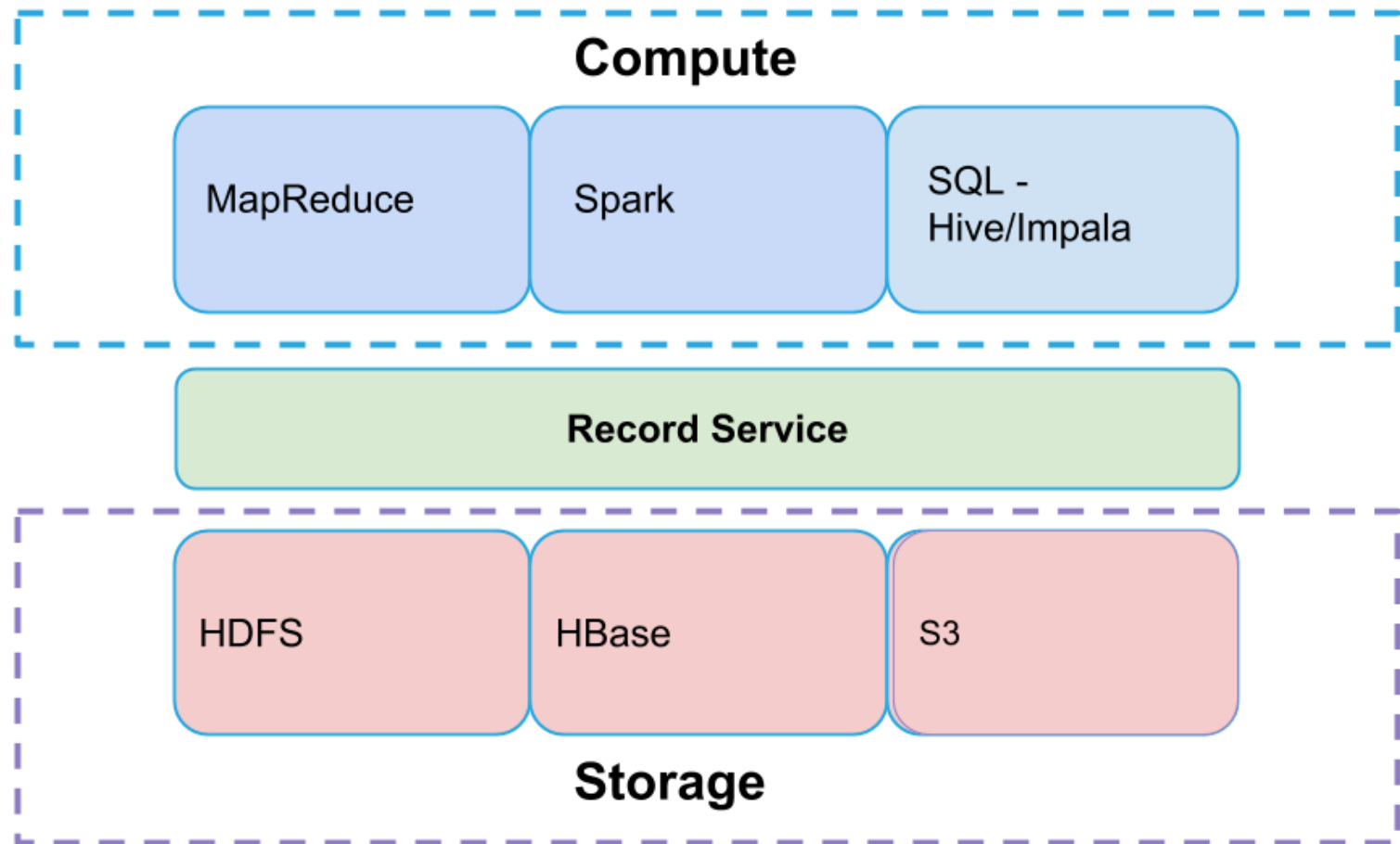
## The Motivation

One of the key aspects of the Hadoop ecosystem is decoupling storage managers (for example, HDFS and Apache HBase) and compute frameworks (for example, MapReduce, Impala and Apache Spark). Although this decoupling allows for far greater flexibility — pick the framework that best solves your problem — it leads to more complexity to ensure that everything works together seamlessly. Furthermore, as Hadoop becomes an increasingly critical infrastructure component for users, the expectations for compatibility, performance,

and security also increase.

RecordService is a new core security layer for Hadoop that sits between the storage managers and compute frameworks to provide a unified data access path.

## Unified Data Access Path



# The Benefits

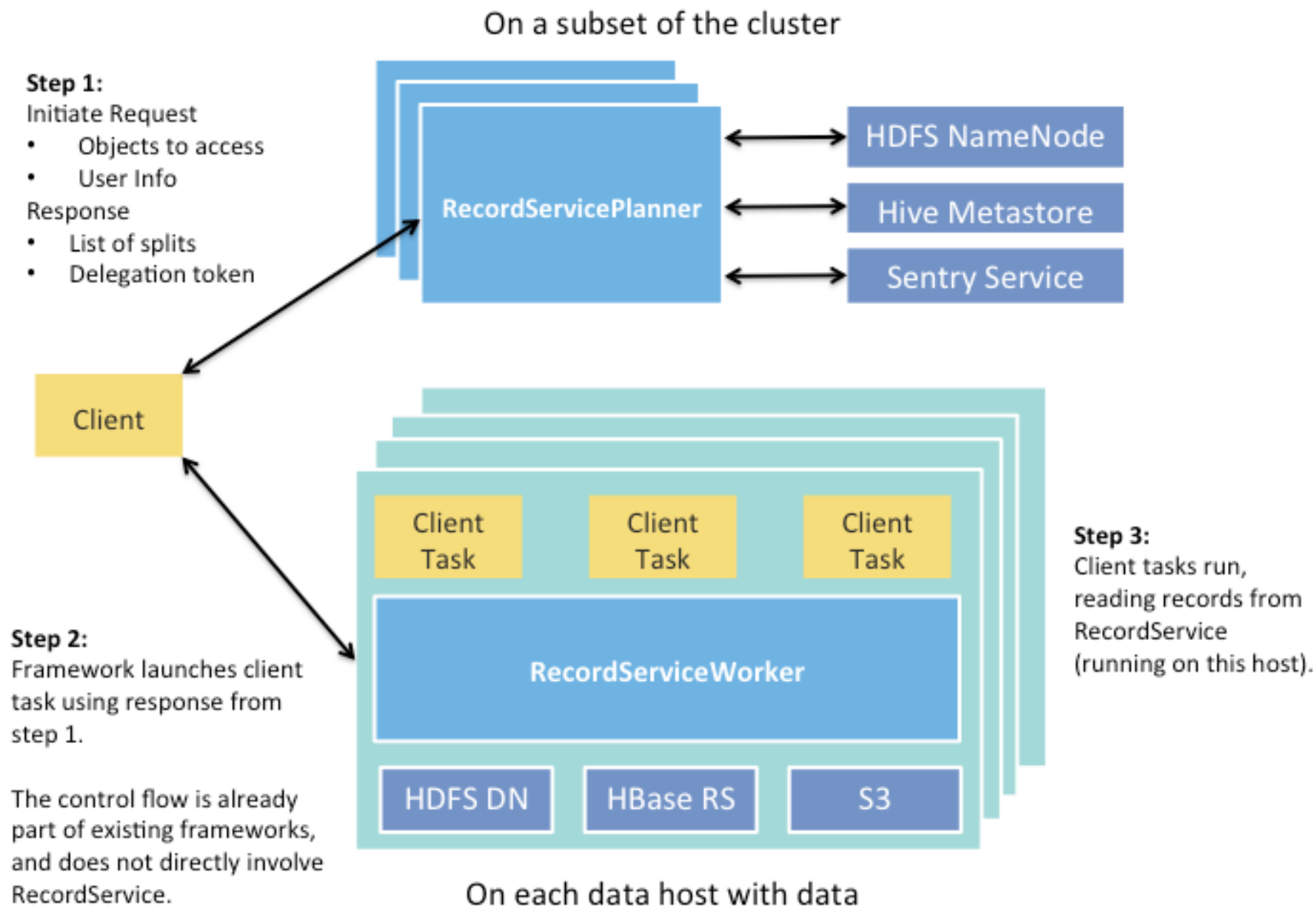
- RecordService provides fine-grained data permissions and enforcement across Hadoop.
- Clients work independently of on-disk storage format.
- The unified data access path provides a single place to implement and test file format–related changes.
- Components swap transparently above or below RecordService.
- Performance is improved through the optimized scanner, dynamic code generation, and Parquet implementation provided by Impala.
- Existing MapReduce and Spark jobs gain Impala-quality performance.
- RecordService can make projections over original source datasets instead of making copies or subsets.

# The Design

RecordService provides the following services.

- RecordServicePlanner — Generates tasks, performs authorization checks, and handles metadata access. Called during input split generation.
- RecordServiceWorker — Executes tasks, and reads and writes to the storage layer.
  - Builds on the highly optimized I/O scheduler and file parsers provided by Impala.
  - Returns rows in a canonical format.
- Thrift APIs.

- Client Integration Libraries — Allow easy migration to RecordService.



[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2015 Cloudera, Inc. All rights reserved.

**RecordService**

# RecordService Beta 0.2.0 Release Notes

This is the documentation for RecordService Beta 0.2.0. For RecordService Beta 0.1.0 documentation, see [RecordService\\_0.1.0.pdf](#).

This release of RecordService is a public beta and should not be run on production clusters. During the public beta period, RecordService is supported through the mailing list [RecordService-user@googlegroups.com](mailto:RecordService-user@googlegroups.com), not through the Cloudera Support Portal.

As you use RecordService during the public beta period, keep in mind the following:

- The RecordService team responds to beta issues as quickly as possible, but cannot commit to issue-resolution or bug-fix delivery times during the public beta period.
- There is no guarantee that a bug will be fixed in a future release.
- The RecordService team does not provide patches for beta releases, and cannot guarantee upgrades from this release to later releases.



- Although multiple releases of beta code might be planned, the contents are not guaranteed. There is no schedule for future beta code releases. Any releases are announced to the user group as they occur.

Topics
New Features in RecordService Beta 0.2.0
Notable Bug Fixes in RecordService Beta 0.2.0
RecordService VM Requirements
Platform and Hardware Support
Storage and File Format Support
Data Type Support
Known Issues
Limitations

## New Features in RecordService Beta 0.2.0

- Support for CDH5.5, including:
  - Sentry Column-Level Authorization.
  - Spark 1.5.
- CSD user experience improvements for Spark and Sentry configuration.
- Performance improvements for loading metadata.

# Notable Bug Fixes in RecordService Beta 0.2.0

- Fix support for multiple planners with path requests.
- Path requests do not contain the connected user in some cases, causing requests to fail with authorization errors.
- `SpecificMutableRow Exception` while running spark-shell with RecordService.
- Port conflict when two `recordservices` are running on the same host.
- Update `task_size` to use total bytes of scan ranges.
- Fail plan request when worker membership is empty.

## RecordService VM Requirements

RecordService VM requires VirtualBox version 4.3 or 5. You can download a free copy of VirtualBox at <https://www.virtualbox.org/wiki/Downloads>.

## Platform and Hardware Support

RecordService supports the following software and hardware configurations when running on your own Hadoop cluster:

- CDH 5.4 and higher
- Server support: RHEL5 and RHEL6, Ubuntu LTS, SLES, and Debian

- Intel Nehalem (or later) or AMD Bulldozer (or later) processor
- 64 GB memory
- For optimal performance, run with 12 or more disks, or use SSD.

## Storage and File Format Support

RecordService supports reading HDFS or S3 of the following file formats:

- Parquet
- Text
- Sequence file
- RC
- Avro

## Data Type Support

RecordService supports the following data types:

- INT (8-64 bits)
- CHAR/VARCHAR
- BOOL

- FLOAT
- DOUBLE
- DECIMAL
- STRING
- TIMESTAMP

RecordService does not support the following data types:

- BLOB/CLOB
- Nested Types

## Known Issues

### **Saving machine state and restarting the VM can result in no registered workers**

After restarting the VM from a saved state, you might receive the following message when attempting to run RecordService applications.

```
Exception in thread "main" java.io.IOException:  
com.cloudera.recordservice.core.RecordServiceException:  
TRecordServiceException(code:INVALID_REQUEST, message:Worker membership is  
empty. Please ensure all RecordService Worker nodes are running.
```

You can verify that the membership is 0 by looking at `http://quickstart.cloudera:11050/membership`.

## Workaround

Restart RecordService by running the following command on the VM:

```
sudo service recordservice-server restart
```

## RecordService client configurations are not properly propagated to Spark jobs

RecordService configuration options are not propagated to Spark jobs using the RecordService custom service descriptor (CSD). All configuration options must be specified in the job or through Cloudera Manager safety valves for Spark.

## Workaround

- Apply configuration options using the Spark configuration safety valve: **Spark -> Configuration -> Spark (Standalone) Client Advanced Configuration Snippet (Safety Valve) for spark-conf/spark-defaults.conf**

```
spark.recordservice.planner.hostports=<comma separated list of planner host:ports>
```

- If the cluster is Kerberized, also set:

```
spark.recordservice.kerberos.principal=<Kerberos principal>
```

- Save changes and deploy the client configuration.

## digest-md5 library not installed on cluster, breaking delegation tokens

The digest-md5 library is not installed by default in parcel deployments.

## Workaround

To install the library on RHEL 6, use the following command-line instruction:

```
sudo yum install cyrus-sasl-md5
```

## Short circuit reads not enabled

### Workaround

In Cloudera Manager, open the HDFS configuration page and search for *shortcircuit*. There are two configurations named **Enable HDFS Short Circuit Read**. One defaults to *true* and one to *false*. Set both values to *true*.

# Limitations

## Security Limitations

- RecordService only supports simple single-table views (no joins or aggregations).
- SSL support has not been tested.
- Oozie integration has not been tested.

## Storage/File Format Limitations

- No support for write path.
- Unable to read from Kudu or HBase.

## **Operation and Administration Limitations**

- No diagnostic bundle support.
- No metrics available in Cloudera Manager.

## **Application Integration Limitations**

- Spark DataFrame is not well tested.
- 

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.

**RecordService**

# Getting Started with RecordService Beta

You can try RecordService beta in multiple ways. First, download the server via the virtual machine (VM) or by downloading the RecordServiceClient parcel or package.

RecordService beta must be installed only on a test cluster, because it is beta software.

## Using the Beta VM

The RecordService beta VM provides a single-node Hadoop cluster, including Impala, MapReduce/YARN, Spark, Sentry, and RecordService. Using the VM is the easiest way to see RecordService in action, and it does not require that you use an existing Hadoop cluster.

See [Using the RecordService VM](#)

## Using Your Own Cluster



You can install and run RecordService on your own cluster built on CDH 5.4 or later.

See [Download and Install RecordService on Your CDH Cluster](#).

## Configuring RecordService

While you should not need to change default settings, there are several properties you can modify to customize your RecordService instance.

See [Configuring RecordService](#).

## Running Examples

To integrate the RecordService into your existing MapReduce and Spark jobs, use the RecordService Client libraries. The client repository contains many example applications you can run right away.

See [RecordService Examples](#).

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.

**RecordService**

# Installing RecordService on Your CDH Cluster

Use these instructions to install RecordService in an existing Hadoop cluster. You can also use the RecordService VM if you want to quickly install and explore the features of RecordService.

See [Download and Install RecordService VM](#).

## Topics

Download RecordService

Installing RecordService as an Add-on Service in Cloudera Manager

Installing the RecordService CSD

Installing RecordService as a Parcel

Deploying RecordService Using Cloudera Manager

Deploying Recommended RecordService Roles

Running a Job Using RecordService

Installing RecordService from a Package (Not Recommended)

# Download RecordService

Use these instructions to install RecordService. Cloudera recommends that you install Cloudera Manager on your cluster, so that you can download and install RecordService as an add-on service in Cloudera Manager.

## Installing RecordService as an Add-on Service in Cloudera Manager

You can install RecordService as an [Add-on Service](#) in Cloudera Manager. First, install both the custom service descriptor (CSD) and the parcel in Cloudera Manager; then, add RecordService as an add-on service from the home page in Cloudera Manager. See

[http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm\\_mc\\_addon\\_services.html](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm_mc_addon_services.html).

## Installing the RecordService CSD

Follow these steps to install the RecordService CSD.

1. Download the CSD from <http://archive.cloudera.com/beta/recordservice/csd>.
2. Upload the CSD to `/opt/cloudera/csd` in the Cloudera Manager server.
3. Change the owner and group for the JAR using the following command-line instruction:

```
chown cloudera-scm:cloudera-scm /opt/cloudera/csd/RECORD_SERVICE-0.2.0.jar
```

4. Update the permissions on the file using the following command-line instruction:

```
chmod 644 /opt/cloudera/csd/RECORD_SERVICE-0.2.0.jar
```

5. Restart the Cloudera Manager server:

1. As the root user on the Cloudera Manager server, run `service cloudera-scm-server restart`.

2. Log in to the Cloudera Manager Admin Console and restart the Cloudera Manager Service.

6. Check whether the CSD successfully installed in `http://{cm-server}:7180/cmf/csd/refresh`. Search for the following entry:

```
{ csdName: "RECORD_SERVICE-0.2.0",  
  serviceType: "RECORD_SERVICE",  
  source: "/opt/cloudera/csd/RECORD_SERVICE-0.2.0.jar",  
  isInstalled: true  
}
```

See

[http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm\\_mc\\_addon\\_services.html](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm_mc_addon_services.html).

## Installing RecordService as a Parcel

Follow these steps to install the RecordService Parcel.

1. Go to `http://{cm-server}:7180/cmf/parcel/status`.
2. If there is no RecordService parcel listed on the status page, click **Edit Settings** and add the RecordService repository URL `http://archive.cloudera.com/beta/recordservice/parcels/latest` to **Remote Parcel Repository URLs**.
3. **Download** the RecordService parcel.
4. **Distribute** the parcel.

5. **Activate** the parcel. Cloudera Manager asks you to restart the entire cluster, but you only need to start/restart RecordService.

See [http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm\\_ig\\_parcels.html](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm_ig_parcels.html)

## Deploying RecordService Using Cloudera Manager

Follow these steps to start RecordService for a cluster from Cloudera Manager.

1. Add a service from the Cloudera Manager home page.
2. Customize role:
  - RecordService Planner and Worker: Select hosts with both the role of DN and NN.
  - RecordService Planner: Select hosts with the role of NN.
  - RecordService Worker: Select hosts with the role of DN.  
**Note:** Only one role is allowed on a single node.
3. Review and modify configuration settings, such as *log dir* and *planner port*.
  - If Sentry is enabled in the cluster, add the sentry configuration to the field **Configuration Snippet (Safety Valve) for sentry-site.xml**. You can find your Sentry configurations either from the Cloudera Manager Sentry process or Sentry process directory (`/var/run/cloudera-scm-agent/process/*-sentry-SENTRY_SERVER/sentry-site.xml`). Here is a sample:

```
<property>
  <name>sentry.service.server.principal</name>
  <value>sentry/_HOST@principal</value>
</property>
<property>
  <name>sentry.service.security.mode</name>
  <value>kerberos</value>
```

```
</property>
<property>
  <name>sentry.service.client.server.rpc-address</name>
  <value>hostname</value>
</property>
<property>
  <name>sentry.service.client.server.rpc-port</name>
  <value>portnum</value>
</property>
<property>
  <name>hive.sentry.server</name>
  <value>server1</value>
</property>
```

4. Start the service.
5. Go to the debug page hostname:11050 to verify that the service started properly.
6. Go to the RecordService cluster page in Cloudera Manager.
7. From **Actions** choose **Deploy Client Configuration**.
8. Run the following command on the host where you run the RecordService client:

```
export HADOOP_CONF_DIR=/etc/recordservice/conf
```

See

[http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm\\_mc\\_add\\_service.html?scroll=cmug\\_topic\\_5\\_1](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm_mc_add_service.html?scroll=cmug_topic_5_1).

## Deploying Recommended RecordService Roles

**RecordService Planner:** Clients (MapReduce and Spark) connect to the Planner to submit requests. The Planner performs authorization checks and generates all tasks to execute for the overall request. For the beta release, run only one Planner node located on a non-DataNode.

**RecordService Worker:** Read data from the underlying storage layer (HDFS/S3) and construct records. Worker roles cannot directly process client requests. Run a Worker on every DataNode in the cluster to ensure locality.

**RecordService Planner Worker:** Acts as both a Planner and a Worker. Provides flexibility for small clusters.

**RecordService Gateway:** Contains configuration information, such as the location of the RecordService Planner. Deploy to all Gateway nodes, and any other nodes that run client jobs that do not already have a RecordService Planner or Worker role deployed.

## Running a Job Using RecordService

You can verify the RecordService server installation by running examples from the client JAR.

1. Download the client library and example tarball. <http://archive.cloudera.com/beta/recordservice/client-dist/recordservice-client-0.2.0-cdh5.5.x-bin.tar.gz>
  - You can also build the client library yourself from the client repository. <https://github.com/cloudera/RecordServiceClient>.
  - Client libraries are also available directly from the Cloudera public Maven repository.
2. To verify the server installation, run client examples in your clusters.



- Log in to one of the nodes in your cluster, and load test data:

```
> wget -q --no-clobber \  
  https://s3-us-west-1.amazonaws.com/recordservice-vm/tpch.tar.gz  
> tar -xzf tpch.tar.gz  
> hadoop fs -mkdir -p /test-warehouse/tpch.nation  
> hadoop fs -put -f tpch/nation/* /test-warehouse/tpch.nation/  
> impala-shell -f create-tbls.sql
```

See <https://github.com/cloudera/RecordServiceClient/blob/master/tests/create-tbls.sql>.

- Run a MapReduce job for RecordCount on tpch.nation:

```
> hadoop jar /path/to/recordservice-examples-0.2.0-cdh5.5.x.jar \  
  com.cloudera.recordservice.examples.mapreduce.RecordCount \  
  "SELECT * FROM tpch.nation" "/tmp/recordcount_output"
```

- Start spark-shell with the RecordService JAR:

```
> path/to/spark/bin/spark-shell \  
  --conf spark.recordservice.planner.hostports=planner_host:planner_port \  
  --jars /path/to/recordservice-spark-0.2.0-cdh5.5.x.jar  
> scala> import com.cloudera.recordservice.spark._  
  import com.cloudera.recordservice.spark._  
> scala> val data = sc.recordServiceRecords("select * from tpch.nation") \  
  data: org.apache.spark.rdd.RDD[Array[org.apache.hadoop.io.Writable]] = \  
  RecordServiceRDD[0] at RDD at RecordServiceRDDBase.scala:57  
> scala> data.count()  
res0: Long = 25
```

See <http://github.com/cloudera/RecordServiceClient/tree/master/java/examples-spark>

## Installing RecordService from a Package (Not

# Recommended)

Follow these steps to install RecordService from a package.

1. Download the package from <http://archive.cloudera.com/beta/recordservice>.
2. Install Hadoop, Hive, Impala, Sentry, ZooKeeper, and any other application you want to use.
3. Install the RecordServicePlanner using the following command-line instruction:

```
./recordserviced -hostname=hostname -recordservice_planner_port=12050 \  
-recordservice_worker_port=0 -recordservice_webserver_port=11050 \  
-webserver_doc_root=path/to/package/lib/recordservice \  
-log_dir=path/to/log/dir -abort_on_config_error=false \  
-lineage_event_log_dir=path/to/log/dir \  
-audit_event_log_dir=path/to/log/dir -profile_log_dir=path/to/log/dir \  
-v=1 -mem_limit=8G
```

4. Install the worker using the following command-line instruction.

```
./recordserviced -hostname=hostname -recordservice_planner_port=0 \  
-recordservice_worker_port=13050 -recordservice_webserver_port=11050 \  
-webserver_doc_root=path/to/package/lib/recordservice \  
-log_dir=path/to/log/dir -abort_on_config_error=false \  
-lineage_event_log_dir=path/to/log/dir \  
-audit_event_log_dir=path/to/log/dir -profile_log_dir=path/to/log/dir \  
-v=1 -mem_limit=8G
```

5. Install both planner and worker on one node:

```
./recordserviced -hostname=hostname -recordservice_planner_port=12050 \  
-recordservice_worker_port=13050 -recordservice_webserver_port=11050 \  
-webserver_doc_root=path/to/package/lib/recordservice \  
-log_dir=path/to/log/dir -abort_on_config_error=false \  
-lineage_event_log_dir=path/to/log/dir \  
-audit_event_log_dir=path/to/log/dir -profile_log_dir=path/to/log/dir \  
-v=1 -mem_limit=8G
```

## 6. Set additional parameters:

- Add the following parameters if the cluster is Kerberized:

```
-principal=kerberos_principle -keytab_file=the/path/to/record_service.keytab
```

- Add the Sentry configuration file, if applicable:

```
-sentry_config=path/to/sentry-site.xml/
```

---

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



# Configuring RecordService

## Topics

Client Configurations

Server Configurations

Kerberos Configuration

Sentry Table Configuration

Delegation Token Configuration

## Client Configurations

While it should not be necessary to change the default configuration, you have the option modifying RecordService properties.

To change any of the setting listed below:

1. In Cloudera Manager, navigate to the RecordService configuration page.
2. Search for `Safety Valve`.
3. In the search results, look for **RecordService (Beta) Client Advanced Configuration Snippet (Safety Valve) for recordservice-conf/recordservice-site.xml**.
4. Add or change the value in the field provided. For example, to change `recordservice.task.fetch.size` to `1000`, add the following code:
 

```
<property>
  <name>recordservice.task.fetch.size</name>
  <value>1000</value>
</property>
```
5. Click **Save Changes**.
6. From the **Actions** menu, choose **Deploy Client Configuration**.

For more information, see [Modifying Configuration Properties Using Cloudera Manager](#).

These are the configuration settings you can (optionally) adjust in your RecordService instance.

CATEGORY	PARAMETER	DESCRIPTION	DEFAULT VALUE
Connectivity	recordservice.planner.hostports	Comma separated list of planner service host/ports.	localhost:12050
Connectivity	recordservice.kerberos.principal	Kerberos principal for the planner service. Required if using Kerberos.	

Connectivity	<code>recordservice.planner.retry.attempts</code>	Maximum number of attempts to retry RecordService RPCs with Planner.	3
Connectivity	<code>recordservice.planner.retry.sleepMs</code>	Sleep between retry attempts with Planner in milliseconds.	5000
Connectivity	<code>recordservice.planner.connection.timeoutMs</code>	Timeout when connecting to the Planner service in milliseconds.	30000
Connectivity	<code>recordservice.planner.rpc.timeoutMs</code>	Timeout for Planner RPCs in milliseconds.	120000
Connectivity	<code>recordservice.worker.retry.attempts</code>	Maximum number of attempts to retry RecordService RPCs with a Worker.	3
Connectivity	<code>recordservice.worker.retry.sleepMs</code>	Sleep in milliseconds between retry attempts with Worker.	5000
Connectivity	<code>recordservice.worker.connection.timeoutMs</code>	Timeout when connecting to the Worker service in milliseconds.	10000
Connectivity	<code>recordservice.worker.rpc.timeoutMs</code>	Timeout for Worker RPCs in milliseconds.	120000
Performance	<code>recordservice.task.fetch.size</code>	Configures the maximum number of records returned when fetching results from the RecordService. If not	5000

		<p>set, the server default is used.</p> <p>Note: This might need to be adjusted according to the type of workloads (MR, Spark, etc), due to the differences in the data processing speed.</p>	
Resource Management	recordservice.task.memlimit.bytes	Maximum memory the server should use per task. Tasks exceeding this limit are aborted. If not set, the server process limit is used.	-1 (Unlimited)
Resource Management	recordservice.task.plan.maxTasks	Hint for maximum number of tasks to generate per PlanRequest. This is not strictly enforced by the server, but is used to determine if task combining should occur. This value might need to be set for large datasets.	-1 (Unlimited)
Resource Management (Advanced)	recordservice.task.records.limit	Maximum number of records returned per task.	-1 (Unlimited)
Logging (Advanced)	recordservice.worker.server.enableLogging	Enable server logging (logging level from Log4j).	FALSE

## Server Configurations

The properties listed on the Cloudera Manager RecordService Configuration page are the ones Cloudera considers the most reasonable to change. However, adjusting these values should not be necessary. Very advanced administrators might consider making minor adjustments.

## Kerberos Configuration

No special configuration is required via Cloudera Manager. Enabling Kerberos on the cluster configures everything.

## Sentry Table Configuration

Sentry is configured for you in the RecordService VM. This section describes how to configure Sentry in a non-VM deployment.

### Prerequisite

Follow CDH documentation to install Sentry and enable it for Hive (and Impala, if applicable).

See

[http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/sg\\_sentry\\_service\\_install.html](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/sg_sentry_service_install.html).

## Configure Sentry with RecordService

1. Enable RecordService to read policy metadata from Sentry:
  - In Cloudera Manager, navigate to the **Sentry Configuration** page.



- In **Admin Groups**, add the user *recordservice*.
  - In **Allowed Connecting Users**, add the user *recordservice*.
2. Save changes.
3. Enable Sentry for RecordService.
- In Cloudera Manager, navigate to **RecordService Configuration**.
  - Select the **Sentry-1** service.
  - In the **Configuration Snippet (Safety Valve) for sentry-site.xml** field, enter the following settings.

```
<property>
  <name>sentry.service.server.principal</name>
  <value>sentry/_HOST@principal</value>
</property>

<property>
  <name>sentry.service.security.mode</name>
  <value>kerberos</value>
</property>

<property>
  <name>sentry.service.client.server.rpc-address</name>
  <value>hostname</value>
</property>

<property>
  <name>sentry.service.client.server.rpc-port</name>
  <value>portnum</value>
</property>

<property>
  <name>hive.sentry.server</name>
  <value>server1</value>
</property>
```

4. Save changes.
5. Restart the Sentry and RecordService services.

## Delegation Token Configuration

No special configuration is required with Cloudera Manager. This is enabled automatically if the cluster is kerberized.

RecordService persists state in Zookeeper, by default, under the /recordservice Zookeeper directory. If this directory is already in use, you can configure the directory with `recordservice.zookeeper.znode`. This is a Hadoop style XML configuration that you can add to the advanced service configuration snippet.

---

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



# RecordService Examples

You can find the source code for RecordService examples in the RecordService Client GitHub repository.

<https://github.com/cloudera/RecordServiceClient/tree/master/java/examples>

Instructions for running the examples are stored in the repository with the source code.

## Topics

Hadoop and MapReduce Examples

RecordService Spark Examples

## Hadoop and MapReduce Examples

Example	Description
RSCat	This example shows how you can output tabular data for any dataset readable by RecordService. It demonstrates a standalone Java application built on the core libraries without using a computation framework such as MapReduce or Spark.

---

SumQueryBenchmark	This demonstrates running a sum over a column and pushing the scan to RecordService. It shows how you can use RecordService to accelerate scan-intensive operations.
Terasort	This is a port of the Hadoop <a href="#">Terasort</a> benchmark test ported to RecordService. See README in the Terasort package for more details. This also demonstrates how to implement a custom InputFormat using the RecordService APIs.
MapredColorCount / MapreduceAgeCount / MapReduceColorCount	These examples are ported from Apache Avro. They demonstrate the steps required to port an existing Avro-based MapReduce job to use RecordService.
RecordCount/WordCount	More MapReduce applications that demonstrate some other InputFormats included in the client library, including TextInputFormat and RecordServiceInputformat. Useful for existing MapReduce jobs already using TextInputFormat.
Reading data from a View, and Enforcing Sentry Permissions (uses RecordCount)	This example demonstrates how an MR job can now read data even when the user only has permission to see part of the data in a file (table). See <a href="https://github.com/cloudera/RecordServiceClient/tree/master/java/examples#how-to-enforce-sentry-permissions-with-mapreduce">https://github.com/cloudera/RecordServiceClient/tree/master/java/examples#how-to-enforce-sentry-permissions-with-mapreduce</a>
com.cloudera.recordservice.examples.avro	Unmodified from the Apache Avro examples, these utilities help you generate sample data.

---

## RecordService Spark Examples

The following examples can be found at

<https://github.com/cloudera/RecordServiceClient/tree/master/java/examples-spark>.

Example	Description
Query1/Query2	Examples that demonstrate RecordService native RDD Resilient Distributed Dataset (RDD) integration using RecordServiceRDD.
WordCount	The Hadoop WordCount example built on top of the RecordService equivalent of <code>textFile()</code> .
SchemaRDDExample	Another example of the native RDD integration, this time using <code>SchemaRecordServiceRDD</code> .
TeraChecksum	This example uses <code>hadoopFile()</code> with the RecordService <code>InputFormats</code> . This is a port of the TeraChecksum MapReduce job, written in Spark.
TpcdsBenchmark	This demonstrates the SparkSQL integration running a portion of the tpcds benchmark.
DataFrameExample	An example that demonstrates DataFrames and RecordService working together.
How to use RecordService with Spark shell	Examples of using spark-shell to interact with RecordService in a variety of ways.
Reading Data from a View and Enforcing Sentry Permissions	This example demonstrates how an MR job may now read data even when the user only has permission to see part of the data in a file (table). See <a href="#">ReadMe.md</a>

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



RecordService



# Using the RecordService VM

## Topics

Downloading RecordService VM

Installing the RecordService VM

Testing Your VM Configuration

Configuring RecordService Environment Variables

Running Examples on the VM

Example: Using RecordService to Control Column-level Access

Example: Using RecordService to Control Row-level Access

Troubleshooting the VM Configuration

Unable to ssh to the VM

Verify VM is Listed Correctly in Hosts
Verify Known Hosts
Verify Workers Are Running
Debugging the VM
Restarting a service
Debugging Via Logs
Dubugging Via Webpage
Other Service Variables

## Downloading RecordService VM

Follow these steps to download the RecordService VM.

1. Install VirtualBox. The VM works with VirtualBox version 4.3 on Ubuntu 14.04 and VirtualBox version 5 on OSX 10.9. Download VirtualBox for free at <https://www.virtualbox.org/wiki/Downloads>.
2. Clone the recordservice-quickstart repository onto your local disk from <https://github.com/cloudera/recordservice-quickstart>.

## Installing the RecordService VM



**Note:** If you have previously installed the VM on your host machine, follow the instructions in [Verify VM is Listed Correctly in Hosts](#) and [Verify Known Hosts](#).

---

Follow these steps to install the RecordService VM.

1. In a terminal window, navigate to the root of the RSQuickstart Git repository.
2. Run the script `install.sh`.

This script downloads an `ova` file and loads it into VirtualBox. The script might ask you to enter a password, because it edits your `/etc/hosts` file to give the VM a stable IP address, `quickstart.cloudera`. When the script completes, the running VM functions as a RecordService server.

## Testing Your VM Configuration

Test that the VM is running and IP forwarding is configured properly.

1. Enter the command `ssh cloudera@quickstart.cloudera`.
2. Enter the password `cloudera`.

If you cannot ssh to the VM, see [Troubleshooting the VM Configuration](#).

Successfully connecting through ssh verifies that you can log in to the VM.

## Configuring RecordService Environment Variables

1. On your host machine, navigate to the root of your RecordServiceClient repository, `$RECORD_SERVICE_HOME`.
2. Run `source config.sh`.
3. Navigate to the `recordservice-quickstart` directory.
4. Run `source vm_env.sh`.
5. Navigate to `$RECORD_SERVICE_HOME/java`.
6. Test your environment with the command `mvn test -DargLine="-Duser.name=recordservice"`

The VM is preconfigured with sample data to execute the tests. The *recordservice* user has access to the data via Sentry.

The VM is not secured with LDAP or Kerberos. If you want to change the security configuration, you can add roles in Sentry through `impala-shell`. If you have `impala-shell` on your host machine, you can connect to the VM by issuing the following command.

```
impala-shell -i quickstart.cloudera:21000 -u impala
```

You can also connect from within `impala-shell`.

```
CONNECT quickstart.cloudera:21000;
```

## Running Examples on the VM

The following examples demonstrate how to use RecordService to implement column- and row-level access in Hadoop. Additional examples are described in the topic [RecordService Examples](#).

## Example: Using RecordService to Control Column-level Access

RecordService provides column-level security. You can restrict users in a group to a subset of columns in a dataset. This allows you to maintain a single, secure dataset that can be viewed and updated by users with specific access rights to only the columns they need.

For example, this schema describes a table that stores information about employees.

<b>column_name</b>	<b>column_type</b>
firstname	string
lastname	string
position	string
department	string
salary	long
phone	long

Suppose you want your employees to have access to the names and phone numbers, but not the position or salary info. You can create a group that allows users to see only the columns you want them to see.

To assign access with column-level restrictions, you create a role, assign the role to a group, and then grant permissions to the role.

- Connect to the RecordService VM using the SSH command `ssh cloudera@quickstart.cloudera`.
- Enter the password `cloudera`.
- Restart Sentry using the following-command line instruction: `sudo service sentry-store restart`
- Currently, you have access to the entire table. Use Impala to select all records from the `rs.employees` table:

```
$ impala-shell
[quickstart.cloudera:21000] > use rs;
Query: use rs
[quickstart.cloudera:21000] > select * from rs.employees;
Query: select * from rs.employees
```

firstname	lastname	position	department	salary	phonenum
Peter	Aaron	WATER RATE TAKER	WATER MGMNT	88968	5551962
Rufi	Bhola	TRAFFIC SIGNAL REPAIRMAN	TRANSPORTN	95888	5551543
Faruk	Bota	POLICE OFFICER	POLICE	80778	5551860
Mary	Bradley	POLICE OFFICER	POLICE	80778	5551638
Joseph	Chu	ASST TO THE ALDERMAN	CITY COUNCIL	70764	5551218
Sam	Cing	CHIEF CONTRACT EXPEDITER	GENERAL SERVICES	84780	5551969
Carol	Cloud	CIVIL ENGINEER IV	WATER MGMNT	104736	5551219
Mackay	Dalford	POLICE OFFICER	POLICE	46206	5551516
Drake	Desmond	GENERAL LABORER - DSS	STREETS & SAN	40560	5551551
Sachen	Dhoot	ELECTRICAL MECHANIC	AVIATION	91520	5551500
Albert	Encino	FIRE ENGINEER	FIRE	90456	5551834
Arthur	Excalibur	POLICE OFFICER	POLICE	86520	5551485

Eric	Freeman	FOSTER GRANDPARENT	FAMILY & SUPPORT	2756	5551706
Fred	Gobi	CLERK III	POLICE	43920	5551597
Ivan	Gorbachev	INVESTIGATOR - IPRA II	IPRA	72468	5551978
Theodore	Henry	POLICE OFFICER	POLICE	69684	5551904
Cranston	Horton	POLICE OFFICER	POLICE	80778	5551924
Bobbi	Ingerwen	FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC	FIRE	98244	5551294
Franz	Isenglass	POLICE OFFICER	POLICE	80778	5551249
Jenny	Jakuti	FIREFIGHTER/PARAMEDIC	FIRE	87720	5551656
Karl	Karloff	ENGINEERING TECHNICIAN VI	WATER MGMNT	106104	5551911
Kathryn	Kretchmer	FIREFIGHTER-EMT	FIRE	91764	5551875
Loren	Lakshmi	LIEUTENANT	FIRE	110370	5551082
Dudley	Less	SENIOR ENVIRONMENTAL INSPECTOR	HEALTH	76656	5551407
Mahood	Mahmut	CROSSING GUARD	POLICE	16692	5551892
Wanda	Myers	GENERAL LABORER - DSS	STREETS & SAN	40560	5551644
Trey	Mystique	ELECTRICAL MECHANIC-AUTO-POLICE MTR MNT	GENERAL SERVICES	91520	5551376
Manuel	Nickels	POLICE OFFICER	POLICE	86520	5551380
Sheldon	Overton	PARAMEDIC	FIRE	54114	5551551
Lana	Park	MOTOR TRUCK DRIVER	STREETS & SAN	71781	5551530
Franny	Periodico	LIBRARY ASSOCIATE - HOURLY	PUBLIC LIBRARY	24835	5551413
Perry	Polite	CIVIL ENGINEER IV	WATER MGMNT	104736	5551891
Mike	Processer	SENIOR PROGRAMMER/ANALYST	DoIT	104736	5551139
Quincy	Quintado	ENGINEERING TECHNICIAN V	BUSINESS AFFAIRS	96672	5551406
Richard	Ramstadt	POLICE OFFICER	POLICE	46206	5551517
Chandra	Sambrosa	SENIOR COMPANION	FAMILY & SUPPORT	2756	5551896
Amber	Sikh	SUPERVISING TRAFFIC CONTROL AIDE	OEMC	55800	5551384
Thomas	Spelt	SANITATION LABORER	STREETS & SAN	72384	5551807
Boli	Tiku	POLICE OFFICER	POLICE	92316	5551921
Clark	Trent	AMBULANCE COMMANDER	FIRE	123948	5551998
Noreen	Umbrella	POOL MOTOR TRUCK DRIVER	STREETS & SAN	16151	5551173
Conrad	Valvolvy	FIREFIGHTER-EMT	FIRE	85680	5551908
June	Vendi	SEWER BRICKLAYER	WATER MGMNT	88566	5551022
Auicula	Ventricular	TREE TRIMMER	STREETS & SAN	74464	5551512
Solomon	Wally	POLICE OFFICER	POLICE	89718	5551750
Winifred	Wonderman	ELECTRICAL MECHANIC	AVIATION	91520	5551990
Elvin	Yahuli	POLICE OFFICER	POLICE	86520	5551675
Aloysius	Zeke	FIREFIGHTER-EMT	FIRE	95460	5551601

Anderson	Zephyr	PERSONAL COMPUTER OPERATOR III	HEALTH	66684	5551717
Zelda	Zion	FIREFIGHTER-EMT	FIRE	99920	5553970

-----

Fetches 50 row(s) in 3.75s

- Use Impala to set permissions for a group of users of `rs.employees`. First, create a role named *demorole*. Next, add the role to the *demogroup* you created before starting Impala. Grant the *select* privilege to *demorole* for only the columns `firstname`, `lastname`, and `phonenumber` from the `rs.employees` table.

```
[quickstart.cloudera:21000] > create role demorole;
Query: create role demorole

Fetches 0 row(s) in 0.40s
[quickstart.cloudera:21000] > grant role demorole to group demogroup;
Query: grant role demorole to group demogroup

[quickstart.cloudera:21000] > GRANT SELECT(firstname, lastname, phonenumber) ON TABLE rs.employees TO ROLE demorole;
Query: GRANT SELECT(firstname, lastname, phonenumber) ON TABLE rs.employees TO ROLE demorole;

Fetches 0 row(s) in 0.11s
```

- Exit Impala.
- From the home directory, execute the following command. This is a trivial example class that counts the number of records in the table. Since the command specifies the *salary* column, to which the *demouser* does not have access, the command fails.

```
[cloudera@quickstart ~]$ sudo su demouser hadoop jar \
./recordservice-client-0.2.0-cdh5.5.x/lib/recordservice-examples-0.2.0-cdh5.5.x.jar \
com.cloudera.recordservice.examples.mapreduce.RecordCount \
"select lastname, salary from rs.employees" "/tmp/count_salary_output"

15/12/08 17:41:26 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
```

```
. . .  
RecordServiceException: TRecordServiceException(code:INVALID_REQUEST, message:Could not plan request., detail:At  
. . .
```

- Now run the same command, but specify the `firstname`, `lastname`, and `phonenumber` columns.

```
[cloudera@quickstart ~]$ sudo su demouser hadoop jar \  
./recordservice-client-0.2.0-cdh5.5.x/lib/recordservice-examples-0.2.0-cdh5.5.x.jar \  
com.cloudera.recordservice.examples.mapreduce.RecordCount \  
"select firstname, lastname, phonenumber from rs.employees" \  
"/tmp/count_phonelist_output"  
  
15/12/08 17:42:25 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032  
. . .  
File Output Format Counters  
  Bytes Written=3
```

- View the results using Hadoop.

```
hadoop fs -cat /tmp/count_phonelist_output/part-r-00000
```

The result returned is a row count of 50.

## Example: Using RecordService to Control Row-level Access

You can also define a view and assign access that restricts a user to certain rows in the data set.

Create a view that restricts the rows returned. For example, where *position* is not `POLICE OFFICER`.

```
[quickstart.cloudera:21000] > use rs;  
[quickstart.cloudera:21000] > create view rs.no_police as select * from rs.employees where position <> "POLICE OFFICER"  
Query: create view rs.no_police as select * from rs.employees where position <> "POLICE OFFICER"
```

Assign that view to *demorole*.

```
[quickstart.cloudera:21000] > grant select on table rs.no_police to role demorole;  
Query: grant select on table rs.no_police to role demorole
```

- Run a query against the `rs.no_police` view.

```
[cloudera@quickstart ~]$ sudo su demouser hadoop jar \  
./recordservice-client-0.2.0-cdh5.5.x/lib/recordservice-examples-0.2.0-cdh5.5.x.jar \  
com.cloudera.recordservice.examples.mapreduce.RecordCount \  
"select firstname, lastname, phonenumber from rs.no_police" \  
"/tmp/count_no_police_output"  
  
15/12/08 17:50:18 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
.  
.  
.  
File Output Format Counters  
  Bytes Written=3
```

- View the results using Hadoop.

```
hadoop fs -cat /tmp/count_no_police_output/part-r-00000
```

The result returned is a row count of 38.

Additional examples are described in the [examples](#) topic.

## Troubleshooting the VM Configuration



## Unable to ssh to the VM

- Ensure that the ssh daemon is running on your machine.
- Ensure that the RecordService VM is running. In your terminal, enter the following command:

```
VBoxManage list runningvms
```

You should see “rs-demo” listed as a running VM.

## Verify VM is Listed Correctly in Hosts

Check that the VM is listed correctly in your `/etc/hosts` file. If you open the file, you should see a line that lists an IP address followed by `quickstart.cloudera`. You can check the VM’s IP with the following command:

```
VBoxManage guestproperty get rs-demo /VirtualBox/GuestInfo/Net/0/V4/IP
```

## Verify Known Hosts

If you’ve used a Cloudera QuickStart VM before, your known hosts file might already have an entry for `quickstart.cloudera` registered to a different key. Delete any reference to `quickstart.cloudera` from your known hosts file, which is usually found in `~/.ssh/known_hosts`.

## Verify Workers Are Running

If you receive an error message similar to the following, your worker nodes are likely not running:

```
Exception in thread "main" java.io.IOException:  
com.cloudera.recordservice.core.RecordServiceException:  
TRecordServiceException(code:INVALID_REQUEST, message:  
Worker membership is empty. Please ensure all RecordService Worker nodes are running.)
```

You can correct the problem by restarting the RecordService server using the following command:

```
sudo service recordservice-server restart
```

## Debugging the VM

### Restarting a service

To restart a service, use the standard RHEL service model.

```
service <service-name> start|stop|restart
```

You can view all of the installed services `/etc/init.d` directory.

### Debugging Via Logs

RecordService logs are in the `/var/log/recordservice` directory. You can find most service logs in the `/var/log` directory.


# Dubugging Via Webpage

View the RecordService debug page on your host machine at `quickstart.cloudera:11050`.

## Other Service Variables

To view the default execution environment for a service, look for its file in the `/etc/default` directory.

---

 Share on Twitter

 Share on Facebook

 Share on Google+

[Cloudera](#) [Source Code](#) [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



RecordService



## RecordService FAQ

### **Q: Where and how are permissions managed?**

**A:** Sentry handles policy metadata. See [Sentry Configuration](#).

In the current version of Sentry, permissions on views allow for fine-grained access control — restricting access by column and row.

### **Q: What kind of data can we use RecordService for?**

**A:** Hive Metastore Tables only. Support might be added in the future for other schema sources, depending on customer demand.

### **Q: What happens if you try to access data controlled by RecordService without using RecordService?**

**A:** Sentry's HDFS Sync feature ensures that the files are locked such that only users with full access to all of the values in a file (with Sentry permissions for the entire table) are allowed to read the files directly. See

## [Configuring the Sentry Service.](#)

### **Q: Do I need to be running Sentry to evaluate the RecordService?**

**A:** No. You can deploy RecordService without Sentry, in which case no authorization checks will happen. This can be useful to evaluate functionality and performance.

### **Q: Are there any APIs to discover the permissions that are set?**

**A:** Hue can show this, as well as SHOW commands in Hive or Impala CLI.

### **Q: What is the RecordService security model? Can you purposely restrict views into the data?**

**A:** Yes, you can control permissions per view. Setting the active role, is not currently supported.

### **Q: Why does RecordService implement its own schema (which seems to be a copy of Hive's schema)?**

**A:** The client API is layered so that it does not have to pull in all dependencies. As you move higher in the client API, you get access to more standard Hadoop objects. In this case, you get a recordservice-hive JAR that returns Hive Schema objects.

### **Q: Can you list tables through RecordService, or do you use the Hive metastore to get tables, and then ask RecordService for the schema?**

**A:** You cannot list tables through RecordService. You have to use the Hive Metastore, or use a tool such as

Hue.

You need only the fully qualified table name to read from a table, so the client does not need to pass the table metadata to RS.

**Q: How does accessing a path directly compare to querying the Hive metastore?**

**A:** RecordService infers the schema. If the path contains a self-describing file, such as Avro or Parquet, it uses that. For files like CSV, RecordService defaults to a `STRING` schema. In a future release RecordService might infer schema from files, but the security rules for paths are still under consideration. For now, RecordService only supports reading from tables defined in the Hive Metastore.

---

[Cloudera](#)   [Source Code](#)   [Mailing List](#)

© 2016 Cloudera, Inc. All rights reserved.



RecordService



## Getting Involved

You are welcome to comment and contribute to the RecordService project in any of these ways.

### Mailing list

If you have questions, comments, reflections, inspirations, suggestions, or need help getting started, please contact the RecordService team via our mailing list. We want to hear from you! [recordservice-user@googlegroups.com](mailto:recordservice-user@googlegroups.com)

### Discussion forum

<http://community.cloudera.com/t5/Beta-Releases/bd-p/Beta>

### Contributions

<http://github.com/cloudera/RecordServiceClient/>

## Documentation

<http://cloudera.github.io/RecordServiceClient/>

## Bug Reporting

File JIRA issue at <https://issues.cloudera.org/projects/RS>.

## Server Implementation

<http://github.com/cloudera/RecordService/>

---

[Cloudera](#)   [Source Code](#)   [Mailing List](#)



