HiveServer2 Overview

Introduction

HiveServer2 (HS2) is a service that enables clients to execute queries against Hive. HiveServer2 is the successor to HiveServer1 which has been deprecated. HS2 supports multi-client concurrency and authentication. It is designed to provide better support for open API clients like JDBC and ODBC.

HS2 is a single process running as a composite service, which includes the Thrift-based Hive service (TCP or HTTP) and a Jetty web server for web UI.

HS2 Architecture

The Thrift-based Hive service is the core of HS2 and responsible for servicing the Hive queries (e.g., from Beeline). Thrift is an RPC framework for building cross-platform services. Its stack consists of 4 layers: Server, Transport, Protocol, and Processor. You can find more details about the layers at https://thrift.apache.org/docs/concepts.

The usage of those layers in the HS2 implementation is described below.

Server

HS2 uses a TThreadPoolServer (from Thrift) for TCP mode, or a Jetty server for the HTTP mode.

The TThreadPoolServer allocates one worker thread per TCP connection. Each thread is always associated with a connection even if the connection is idle. So there is a potential performance issue resulting from a large number of threads due to a large number of concurrent connections. In the future HS2 might switch to another server type for TCP mode, for example TThreadedSelectorServer. Here is an article about a performance comparison between different Thrift Java servers.

Transport

HTTP mode is required when a proxy is needed between the client and server (for example, for load balancing or security reasons). That is why it is supported, as well as TCP mode. You can specify the transport mode of the Thrift service through the Hive configuration property hive.server2.transport.mode.

Protocol

The Protocol implementation is responsible for serialization and deserialization. HS2 is currently using TBinaryProtocol as its Thrift protocol for serialization. In the future other protocols may be considered, such as TCompactProtocol, based on more performance evaluation.

Processor

Process implementation is the application logic to handle requests. For example, the ThriftCLIService.ExecuteStatement() method implements the logic to compile and execute a Hive query.

Dependencies of HS2

- Metastore
  The metastore can be configured as embedded (in the same process as HS2) or as a remote server (which is a Thrift-based service as well). HS2 talks to the metastore for the metadata required for query compilation.

- Hadoop cluster
  HS2 prepares physical execution plans for various execution engines (MapReduce/Tez/Spark) and submits jobs to the Hadoop cluster for execution.

You can find a diagram of the interactions between HS2 and its dependencies here.

JDBC Client

The JDBC driver is recommended for the client side to interact with HS2. Note that there are some use cases (e.g., Hadoop Hue) where the Thrift client is used directly and JDBC is bypassed.

Here is a sequence of API calls involved to make the first query:

- The JDBC client (e.g., Beeline) creates a HiveConnection by initiating a transport connection (e.g., TCP connection) followed by an OpenSession API call to get a SessionHandle. The session is created from the server side.
- The HiveStatement is executed (following JDBC standards) and an ExecuteStatement API call is made from the Thrift client. In the API call, SessionHandle information is passed to the server along with the query information.
- The HS2 server receives the request and asks the driver (which is a CommandProcessor) for query parsing and compilation. The driver kicks off a background job that will talk to Hadoop and then immediately returns a response to the client. This is an asynchronous design of the ExecuteStatement API. The response contains an OperationHandle created from the server side.
- The client uses the OperationHandle to talk to HS2 to poll the status of the query execution.

Source Code Description

The following sections help you locate some basic components of HiveServer2 in the source code.

Server Side

- **Thrift IDL file for TCLIService**: https://github.com/apache/hive/blob/master/service-rpc/if/TCLIService.thrift.
- **TCLIService.Iface implemented by**: *org.apache.hive.service.cli.thrift.ThriftCLIService* class.
- **ThriftCLIService subclassed by**: *org.apache.hive.service.cli.thrift.ThriftBinaryCLIService* and *org.apache.hive.service.cli.thrift.ThriftHttpCLIService* for TCP mode and HTTP mode respectively.
- **org.apache.hive.service.cli.thrift.EmbeddedThriftBinaryCLIService class:** Embedded mode for HS2. Don't get confused with embedded metastore, which is a different service (although the embedded mode concept is similar).
- **org.apache.hive.service.cli.session.HiveSessionImpl class**: Instances of this class are created on the server side and managed by an *org.apache.accumulo.tserver.TabletServer.SessionManager* instance.
- **org.apache.hive.service.cli.operation.Operation class**: Defines an operation (e.g., a query). Instances of this class are created on the server and managed by an *org.apache.hive.service.cli.operation.OperationManager* instance.
- **org.apache.hive.service.auth.HiveAuthFactory class**: A helper used by both HTTP and TCP mode for authentication. Refer to Setting Up HiveServer2 for various authentication options, in particular Authentication/Security Configuration and Cookie Based Authentication.

Client Side

- **org.apache.hive.jdbc.HiveConnection class*:*** Implements the *java.sql.Connection* interface (part of JDBC*).* An instance of this class holds a reference to a *SessionHandle* instance which is retrieved when making Thrift API calls to the server.
- **org.apache.hive.jdbc.HiveStatement class:** Implements the *java.sql.Statement* interface (part of JDBC). The client (e.g., Beeline) calls the *HiveStatement.execute()* method for the query. Inside the *execute*() method, the Thrift client is used to make API calls.
- **org.apache.hive.jdbc.HiveDriver class**: Implements the *java.sql.Driver* interface (part of JDBC). The core method is *connect*() which is used by the JDBC client to initiate a SQL connection.

Interaction between Client and Server

- **org.apache.hive.service.cli.SessionHandle class:** Session identifier. Instances of this class are returned from the server and used by the client as input for Thrift API calls.
- **org.apache.hive.service.cli.OperationHandle class**: Operation identifier. Instances of this class are returned from the server and used by the client to poll the execution status of an operation.

Resources

How to set up HS2: Setting Up HiveServer2

HS2 clients: HiveServer2 Clients

User interface: Web UI for HiveServer2

Metrics: Hive Metrics

Cloudera blog on HS2: http://blog.cloudera.com/blog/2013/07/how-hiveserver2-brings-security-and-concurrency-to-apache-hive/