# Druid Cluster Install Guide

(customized for Metatron Discovery)

### Prerequisites

You will need:

* CentOS 6.x or Higher
* JDK 1.8 (recommended Oracle JDK)
* MySQL 5.5 or Higher
* Apache Zookeeper 3.4.10 or Higher
* Need to Ansible Installation on distribution server
* Hadoop Cluster (2.7.3 recommended)

### DBMS Database & Privileges Setup

Database name : druid

|  |
| --- |
| create database druid CHARACTER SET utf8;  grant all privileges on druid.\* TO druid@localhost identified by 'druid';  grant all privileges on druid.\* TO druid@'%' identified by 'druid';  flush privileges; |

### Zookeeper Installation & Setup

Install Apache Zookeeper to act as a Distributed Coordinator.

### Ansible Installation & Setup

Ansible is used to deploy Druid to clustered servers. Ansible communicates with client computers through SSH, so each server you want to manage should be accessible from the Ansible server without password.

### Hadoop Cluster Installation & Setup

Configure directories in hdfs as below

|  |
| --- |
| * hadoop fs -mkdir -p /druid/storage * hadoop fs -mkdir -p /druid/logs |

## Druid Cluster Installation

\*Need to modify to suit your environment

### Download customized Druid & Unzip

Download and Unzip the binary file. [This Link](https://sktmetatronkrsouthshared.blob.core.windows.net/metatron-public/discovery-dist/latest/druid-0.9.1-latest-hadoop-2.7.3-bin.tar.gz) – for Hadoop 2.7.3

|  |
| --- |
| Example location>  ./druid\_dist/druid-0.9.1-SNAPSHOT.{discovery.version}-hadoop-2.7.3 |

### Download deploy template shell & Unzip

Download and Unzip the binary file. [This Link](https://sktmetatronkrsouthshared.blob.core.windows.net/metatron-public/discovery-dist/druid_dist.tar.gz)

|  |
| --- |
| Example Location>  ./druid\_dist/druid\_dist\_v2 |

### Ansible Host Setting

Ansible keeps track of all of the servers that it known about through a “hosts”. We need to set up this file first before we can begin to communicate with our other computers. Refer to the host file under the ./druid\_dist/druid\_dist\_v2 directory and set it in /etc/ansible/hosts.

### Modify deploy environment shell

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/env.sh |
| #!/bin/sh  TARGET\_SYSTEM="metatron-hadoop01" #System Name  BECOME\_USER="metatron" #User for Druid excution  BECOME\_GROUP="metatron" #User Group for Druid excution  BECOME\_USER\_GROUP="$BECOME\_USER:$BECOME\_GROUP"  TARGET\_DIR="/home/metatron/servers/druid" #Target deploy directory  DIST\_DIR="/home/metatron/servers/druid\_dist" #Distribution directory  ROOT\_TEMP\_DIR="/home/metatron/data1/druid" #Temp directory  TMP\_DIR="$ROOT\_TEMP\_DIR/tmp"  VAR\_DIR="$ROOT\_TEMP\_DIR/var"  SEGMENET\_CACHE\_DIR=("/home/metatron/data1/druid/segment-cache" "/home/metatron/data2/druid/segment-cache" "/home/metatron/data3/druid/segment-cache" "/home/metatron/data4/druid/segment-cache" "/home/metatron/data5/druid/segment-cache" "/home/metatron/data6/druid/segment-cache" "/home/metatron/data7/druid/segment-cache" "/home/metatron/data8/druid/segment-cache" "/home/metatron/data9/druid/segment-cache" "/home/metatron/data10/druid/segment-cache") #Local cache disk directory |

### Create necessary directory by shell

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/setup\_druid.sh |

### Configuring Druid’s property file

The major configuration has already been put in the "bootstrap" directory. Below is a list of things you need to change or update depending on your environment.

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/\_common  /common.runtime.properties |
| # Zookeeper – zookeeper cluster server list  druid.zk.service.host={[zkhost01}:](https://tde.sktelecom.com/wiki/display/METATRON/druid.zk.service.host=zkhost01:)2181,{zkhost02}:2181,{zkhost03}:2181  druid.zk.paths.base=/druid  # Metadata Storage (MySQL or Maria DB connection info.)  druid.metadata.storage.type=mysql  druid.metadata.storage.connector.connectURI=jdbc:mysql://{mysql hostname}:{mysql port}/druid?useUnicode=true&characterEncoding=UTF-8  druid.metadata.storage.connector.user=druid  druid.metadata.storage.connector.password=druid  # Deep storage  # For HDFS:  druid.storage.type=hdfs  druid.storage.storageDirectory=/druid/storage  # Indexing service logs  # For HDFS:  druid.indexer.logs.type=hdfs  druid.indexer.logs.directory=hdfs://{clustername}/druid/logs  # Query Cache, 50MB  druid.cache.type=local  druid.cache.sizeInBytes=52428800  # Service discovery  # Indexing service discovery. Update this if you change your overlord's "druid.service".  druid.selectors.indexing.serviceName=druid/prod/overlord  druid.selectors.coordinator.serviceName=druid/prod/coordinator |

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/coordinator/runtime.properties |
| # Default host, port, service name.  [druid.service=druid/prod/coordinator](https://tde.sktelecom.com/wiki/display/METATRON/druid.service=druid/prod/coordinator)  [druid.port=](https://tde.sktelecom.com/wiki/display/METATRON/druid.port=)8081 |

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/overlord/runtime.properties |
| # Default host, port, service name.  druid.service=druid/prod/overlord  druid.port=8090 |

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/broker/runtime.properties |
| druid.service=druid/prod/broker  druid.port=8082  # HTTP server threads  druid.broker.http.numConnections=20  #druid.server.http.numThreads=8  druid.server.http.numThreads=20  # Processing threads and buffers, 512MB  druid.processing.buffer.sizeBytes=536870912  druid.processing.numThreads=24  druid.server.http.maxIdleTime=PT10m  druid.broker.http.readTimeout=PT30M  # Query cache (we use a small local cache)  druid.broker.cache.useCache=true  druid.broker.cache.populateCache=true  druid.broker.cache.unCacheable=["select", "groupBy"]  # Query Result Count  druid.query.groupBy.maxResults=100000000  druid.query.groupBy.maxIntermediateRows=100000000 |

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/historical/runtime.properties |
| druid.service=druid/prod/historical  druid.port=8083  # HTTP server threads  druid.server.http.numThreads=15  # Processing threads and buffers, 512MB  druid.processing.buffer.sizeBytes=536870912  druid.processing.numThreads=24  # ex. 50G per disk  druid.segmentCache.locations=[{"path":"/data1/druid/segment-cache","maxSize":53687091200},{"path":"/data2/druid/segment-cache","maxSize":53687091200},{"path":"/data3/druid/segment-cache","maxSize":53687091200},{"path":"/data4/druid/segment-cache","maxSize":53687091200},{"path":"/data5/druid/segment-cache","maxSize":53687091200},{"path":"/data6/druid/segment-cache","maxSize":53687091200},{"path":"/data7/druid/segment-cache","maxSize":53687091200},{"path":"/data8/druid/segment-cache","maxSize":53687091200},{"path":"/data9/druid/segment-cache","maxSize":53687091200},{"path":"/data10/druid/segment-cache","maxSize":53687091200}]  druid.server.maxSize=536870912000  druid.historical.cache.useCache=true  druid.historical.cache.populateCache=true  druid.historical.cache.unCacheable=["select", "groupby"]  druid.query.groupBy.maxResults=100000000  druid.query.groupBy.maxIntermediateRows=100000000 |

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/conf/druid/middleManager/runtime.properties |
| # default port, service name.  druid.service=druid/prod/middlemanager  druid.port=8091  # HTTP server threads  druid.server.http.numThreads=12  # Processing threads and buffers  druid.processing.buffer.sizeBytes=256000000  druid.processing.numThreads=2  # Hadoop indexing  druid.indexer.task.hadoopWorkingPath=/druid/indexing-tmp  druid.indexer.task.defaultHadoopCoordinates=["org.apache.hadoop:hadoop-client:2.7.6"]  # Task launch parameters  druid.indexer.runner.javaOpts=-server -Xmx3g -XX:+UseG1GC -XX:MaxGCPauseMillis=100 -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -Duser.timezone=UTC -Dfile.encoding=UTF-8 -Djava.util.logging.manager=org.apache.logging.log4j.jul.LogManager  druid.indexer.task.baseTaskDir=/home/metatron/data1/druid/var/task  # Peon properties  druid.indexer.fork.property.druid.monitoring.monitors=["com.metamx.metrics.JvmMonitor"]  druid.indexer.fork.property.druid.processing.buffer.sizeBytes=536870912  druid.indexer.fork.property.druid.processing.numThreads=2  druid.indexer.fork.property.druid.segmentCache.locations=[{"path": "/home/metatron/data1/druid/var/segment-cache", "maxSize": 0}]  druid.indexer.fork.property.druid.server.http.numThreads=50  druid.worker.capacity=20  druid.worker.ip=localhost  druid.worker.version=0 |

### Modify deploy envirnonment script

|  |
| --- |
| ./druid\_dist/druid\_dist\_v2/druid\_bootstrap/scripts/druid-env.sh |
| #!/bin/sh  DRUID\_HADOOP\_CONF\_DIR="/usr/lib/hadoop/etc/hadoop"  DRUID\_LOG\_DIR=/home/metatron/data1/druid/var |

### Deploy to each cluster server

|  |
| --- |
| #Move modified conf and script file to binary directory  ./druid\_dist/druid\_dist\_v2/init.sh {binary directory name}  #Deploy to each cluster server  ./druid\_dist/druid\_dist\_v2/dist.sh {binary directory name} |

### Start up Druid services

|  |
| --- |
| #Start up  ./druid\_dist/druid\_dist\_v2/run\_druid.sh  #Status monitoring  ./druid\_dist/druid\_dist\_v2/status\_druid.sh  #Stop druid  ./druid\_dist/druid\_dist\_v2/kill\_druid.sh |

## Additional Info.

### Console URI

|  |
| --- |
| druid overlord - [http://OVERLORD-HOST:8090/console.html](http://dtg2-j2-01:8092/console.html) |
| druid coordinator - [http://COORDINATOR-HOST:8081](http://dtg2-j2-01:8081/) |
| druid broker - <http://BROKER-HOST:8082> |

### Connecting metatron Discovery

After the cluster setup, you need to add or update the configuration as shown below to connect with discovery.

- polaris.engine: The setting used for connecting each druid node.

- polaris.ingestion: The setting used for copying files to the middle manager on ingestion stage.

- polaris.query: The setting used when the file download & loads the data source.

|  |
| --- |
| polaris:  engine:  hostname:  broker: http://{BrokerHostName}:8082  overlord: http:// {Overlord Master HostName}:8090  coordinator: http:// {Coordinator Master HostName}:8081  ingestion:  loader:  remoteType: SSH  localBaseDir: ${java.io.tmpdir:-/tmp}  remoteDir: ${java.io.tmpdir:-/tmp}  hosts:  middlemanager\_hostname01:  port: 22  username: metatron  password: password  middlemanager\_hostname02:  port: 22  username: metatron  password: pem:/tmp/metatron.pem  query:  loader:  remoteType: SSH  localBaseDir: ${java.io.tmpdir:-/tmp}  remoteDir: ${java.io.tmpdir:-/tmp}  hosts:  broker\_hostname01:  port: 22  username: metatron  password: password |

※ ‘middlenamanager\_hostname’ must match the hostname in the worker list on overload console.