Log-streaming system

Documentation

Introduction
This application is designed for processing sdp transaction logs in near real-time, using Big data and Hadoop technologies. In here we have used cloudera cdh distribution and Knowage, for implementation purpose.

Features
Near real time data processing
Fault tolerance with checkpoints
Interactive Reports and dashboards

Requirements

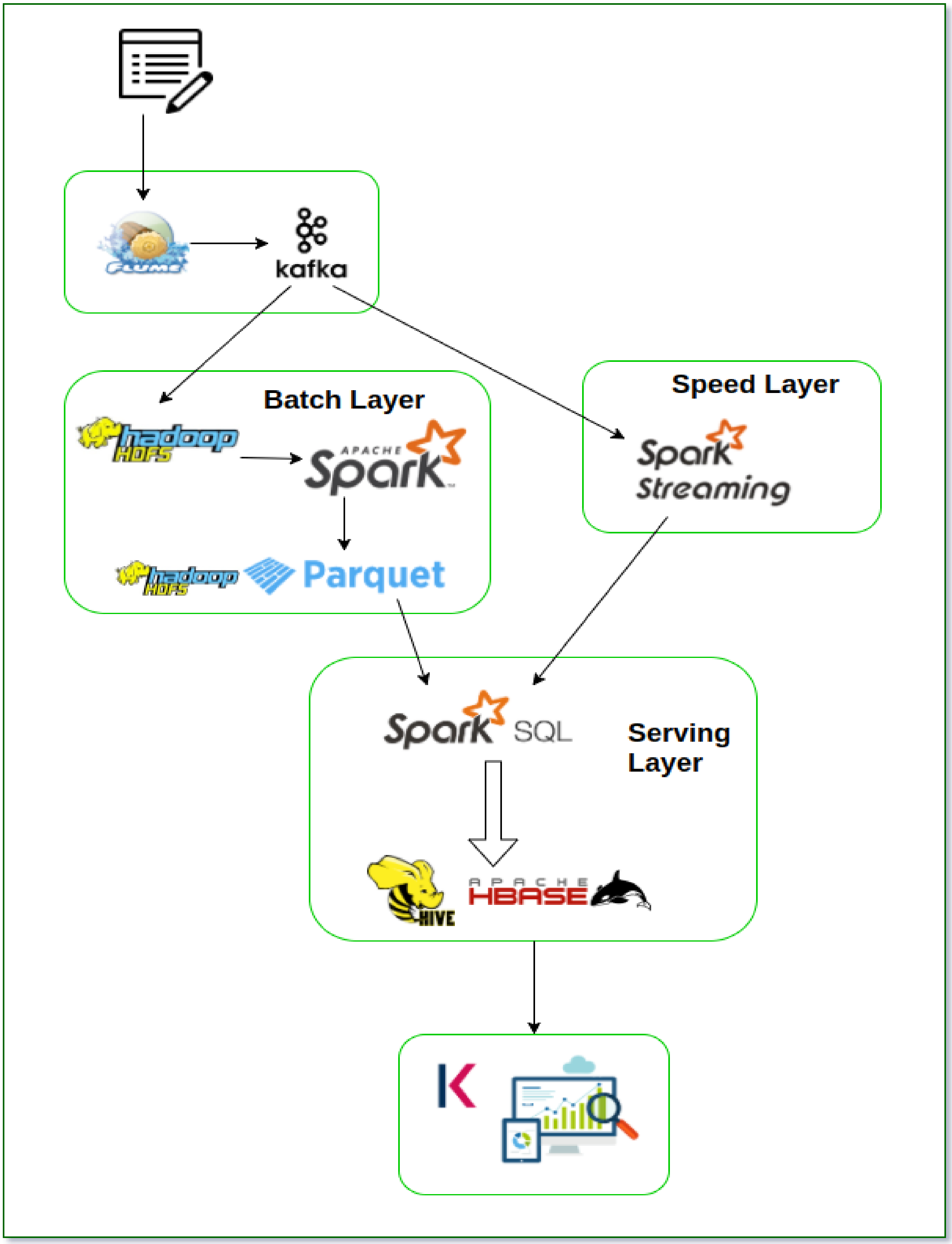
- Cloudera CDH version 5.12.x
  Or
  A Set-up with following components
  Hadoop with Yarn and Map-reduce
  Apache zookeeper
  Apache Spark
  Apache Hive
  Apache Impala
  Apache Flume
  Apache Parquet

- Apache Kafka

- Java 8
- Mysql server 5.5 or above

- Knowage Server (CE) 6.1.1
- Knowage Report Designer 6.1.0

Architecture of log-processing system



Cloudera quickstart CDH installation

1. Download Vmware player or Virtualbox and install it.
2. Download Cloudera Quickstart VM from https://www.cloudera.com/downloads/quickstart_vms/5-12.html
3. Create a VM using downloaded cloudera quickstart VM, in Virtualbox.

Cloudera installation

Note : requires Java 1.7 or above

Install using apt-get

1. First add the repository
    - Run the following command

      ```
      sudo vi /etc/apt/sources.list.d/cloudera.list
      ```

    - Add the below lines to the file

```
deb [arch=amd64] http://archive.cloudera.com/cdh5/ubuntu/xenial/amd64/cdh xenial-cdh5 contrib
deb-src http://archive.cloudera.com/cdh5/ubuntu/xenial/amd64/cdh xenial-cdh5 contrib
```

    - Run the below command

```
sudo vi /etc/apt/preferences.d/cloudera.pref
```

    - Add the following lines to the opened file

```
Package: *
Pin: release o=Cloudera, l=Cloudera
Pin-Priority: 501
```

2. Install hadoop-yarn-resourcemanager, hadoop-hdfs-namenode, hadoop-mapreduce-historyserver, hadoop-yarn-proxyserver and hadoop-client

```
sudo apt-get install hadoop-yarn-resourcemanager
sudo apt-get install hadoop-hdfs-namenode
```

```
sudo apt-get install hadoop-mapreduce-historyserver
hadoop-yarn-proxyserver
sudo apt-get install hadoop-client
```

3. Install the following to build the data node in the same node which has installed the name node.

```
sudo apt-get install hadoop-hdfs-secondarynamenode
sudo apt-get install hadoop-yarn-nodemanager hadoop-hdfs-datanode
hadoop-mapreduce
```

4. Create a separate directory to add the configurations for hadoop

```
sudo cp -r /etc/hadoop/conf.empty /etc/hadoop/<directory-name>
```

5. Make the new directory as the currently using configurations

```
sudo update-alternatives --install /etc/hadoop/conf hadoop-conf
/etc/hadoop/<directory-name> 50
sudo update-alternatives --set hadoop-conf /etc/hadoop/<directory-name>
```

6. Change the core-site.xml file in the /etc/hadoop/<directory-name>

```
<property>
    <name>fs.defaultFS</name>
    <value>hdfs://<ip-address>:8020</value>  <!-- localhost or ip address
-->
  </property>
  <property>
     <name>io.compression.codecs</name>
<value>org.apache.hadoop.io.compress.DefaultCodec,org.apache.hadoo
p.io.compress.GzipCodec,org.apache.hadoop.io.compress.BZip2Codec,or
g.apache.hadoop.io.compress.SnappyCodec</value>
  </property>
  <property>
    <name>hadoop.proxyuser.mapred.groups</name>
    <value>*</value>
  </property>
  <property>
```

```
      <name>hadoop.proxyuser.mapred.hosts</name>
      <value>*</value>
   </property>
```

7. Change the hdfs-site.xml file

```
<property>
     <name>dfs.permissions.superusergroup</name>
     <value>hadoop</value>
</property>
<property>
     <name>dfs.namenode.name.dir</name>
     <value>file:///data/1/dfs/nn</value>
</property>
<property>
      <name>dfs.datanode.data.dir</name>
     <value>file:///data/1/dfs/dn</value>
</property>
<property>
      <name>dfs.namenode.http-address</name>
      <value><ip-address>:50070</value>   <!-- localhost or ip address -->
      <description>
            The address and the base port on which the dfs NameNode Web
UI will listen.
       </description>
</property>
<property>
      <name>dfs.webhdfs.enabled</name>
      <value>true</value>
</property>
```

8. Add new configurations to mapred-site.xml

```
<property>
     <name>mapreduce.framework.name</name>
     <value>yarn</value>
    </property>
    <property>
     <name>mapreduce.jobhistory.address</name>
```

```
 <value><ip-address>:10020</value>
</property>
<property>
 <name>mapreduce.jobhistory.webapp.address</name>
 <value><ip-address>:19888</value>
</property>
<property>
<name>yarn.app.mapreduce.am.staging-dir</name>
<value>/user</value>
</property>
```

9. Change the yarn-site.xml file accordingly

```
<property>
          <name>yarn.resourcemanager.hostname</name>
          <value><ip-address></value>
  </property>
  <property>
          <name>yarn.resourcemanager.resource-tracker.address</name>
          <value><ip-address>:8031</value>
  </property>
  <property>
          <name>yarn.resourcemanager.address</name>
          <value><ip-address>:8032</value>
  </property>
  <property>
          <name>yarn.resourcemanager.scheduler.address</name>
          <value><ip-address>:8030</value>
  </property>
  <property>
          <name>yarn.resourcemanager.admin.address</name>
          <value><ip-address>:8033</value>
  </property>
  <property>
          <name>yarn.resourcemanager.webapp.address</name>
          <value><ip-address>:8088</value>
  </property>
  <property>
```

```
          <description>Classpath for typical applications.</description>
          <name>yarn.application.classpath</name>
          <value>
          $HADOOP_CONF_DIR,

$HADOOP_COMMON_HOME/*,$HADOOP_COMMON_HOME/lib/*,
          $HADOOP_HDFS_HOME/*,$HADOOP_HDFS_HOME/lib/*,
          $HADOOP_MAPRED_HOME/*,$HADOOP_MAPRED_HOME/lib/*,
          $HADOOP_YARN_HOME/*,$HADOOP_YARN_HOME/lib/*
          </value>
  </property>
  <property>
          <name>yarn.nodemanager.aux-services</name>
          <value>mapreduce_shuffle</value>
  </property>
  <property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
          <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
          <name>yarn.nodemanager.local-dirs</name>
          <value>file:///data/1/yarn/local</value>
  </property>
  <property>
          <name>yarn.nodemanager.log-dirs</name>
          <value>file:///data/1/yarn/logs</value>
  </property>
  <property>
          <name>yarn.log.aggregation.enable</name>
          <value>true</value>
  </property>
  <property>
          <description>Where to aggregate logs</description>
          <name>yarn.nodemanager.remote-app-log-dir</name>
          <value>hdfs://var/log/hadoop-yarn/apps</value>
  </property>
```

10. Add the ip of the node where namenode has been installed in the masters file

```
sudo vi /etc/hadoop/<directory-name>/masters
```

11. Create the data directory and give relevant permissions

```
sudo mkdir -p /data/1/dfs/nn
sudo chown -R hdfs:hdfs /data/1/dfs/nn
sudo chmod 700 /data/1/dfs/nn
```

12. Format the namenode

```
sudo -u hdfs hdfs namenode -format
```

13. Add the ip address to the slaves file

```
 sudo vi /etc/hadoop/<directory-name>
```

14. Start the HDFS and other necessary services by running the following command

```
for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do sudo service $x start ; done
```

15. Create /tmp dir on hdfs

```
sudo -u hdfs hadoop fs -mkdir /tmp
sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
```

16. Create user directories on hdfs

```
sudo -u hdfs hadoop fs -mkdir /user
sudo -u hdfs hadoop fs -mkdir /user/<user>
sudo -u hdfs hadoop fs -chown <user>:ubuntu /user/<user>
```

17. Directory for Job History on hdfs

```
sudo -u hdfs hadoop fs -mkdir -p /user/history
sudo -u hdfs hadoop fs -chmod -R 1777 /user/history
sudo -u hdfs hadoop fs -chown mapred:hadoop /user/history
```

18. Directory for YARN log files on hdfs

```
sudo -u hdfs hadoop fs -mkdir -p /var/log/hadoop-yarn
sudo -u hdfs hadoop fs -chown yarn:mapred /var/log/hadoop-yarn
```

19. Create local directories and give relevant permissions

```
sudo mkdir -p /data/1/yarn/local
sudo mkdir -p /data/1/yarn/logs
sudo chown -R yarn:yarn /data/1/yarn/local
sudo chown -R yarn:yarn /data/1/yarn/logs
```

20. Install zookeeper

Base package

```
sudo apt-get install zookeeper
```

Zookeeper server

```
sudo apt-get install zookeeper-server
```

Note : Change the permissions of the data directory

For more details :
https://www.cloudera.com/documentation/enterprise/5-5-x/topics/cdh_ig_zookeeper_package_install.html

21. Install spark (cloudera version)

```
sudo apt-get install spark-core spark-master spark-worker spark-history-server spark-python
```

For more details :
https://www.cloudera.com/documentation/enterprise/5-14-x/topics/cdh_ig_spark_install.html#spark_install_upgrade

22. Install hive

```
sudo apt-get install <pkg1> <pkg2>
```

Packages:

- hive – base package that provides the complete language and runtime

- hive-metastore – provides scripts for running the metastore as a standalone service (optional)
- hive-server2 – provides scripts for running HiveServer2
- hive-hbase - optional; install this package if you want to use Hive with HBase

For more details :
https://www.cloudera.com/documentation/enterprise/latest/topics/cdh_ig_hive_install.html#topic_18_3

23. Configure hive metastore to MySQL, PostgreSQL, and Oracle
    For more details refer the following link :

https://www.cloudera.com/documentation/enterprise/5-14-x/topics/cdh_ig_hive_metastore_configure.html

24. Install Flume

```
sudo apt-get install <pkg>
```

Packages

- flume-ng — Everything you need to run Flume
- flume-ng-agent — Handles starting and stopping the Flume agent as a service
- flume-ng-doc — Flume documentation

For more details :
https://www.cloudera.com/documentation/enterprise/5-14-x/topics/cdh_ig_flume_package_install.html

25. Install apache kafka
    Download the apache kafka using the following link
    http://kafka.apache.org/downloads.html

    Extract the package and move to the /usr/lib directory

    Change the server.properties file in the config/ directory

26. Install impala

```
sudo apt-get install impala
sudo apt-get install impala-server
sudo apt-get install impala-state-store
sudo apt-get install impala-catalog
```

For more details :
https://www.cloudera.com/documentation/enterprise/5-14-x/topics/impala_noncm_installation.html

27. Installing hue

```
sudo apt-get install hue
```

For more details :
https://www.cloudera.com/documentation/enterprise/5-9-x/topics/cdh_ig_hue_install.html

28. Configure cdh components to connect to hue

Follow the below link
https://www.cloudera.com/documentation/enterprise/5-14-x/topics/cdh_ig_cdh_hue_configure.html

Installation in other environments

On RHEL-compatible Systems

- Use the above installed packages using the **yum** package manager

On SLES systems

- Use the **zypper** instead of apt-get from the above installation

Cluster installation

- Install java 1.7 or above in each node
- Install the namenode and other services required in a one node
- Install the required services for a datanode in separate nodes
- Add the datanodes ip addresses in the slaves file which is in the namenode configuration directory (/etc/hadoop/<directory-name>)

- Configure a secondary namenode in a one node
    - sudo apt-get install hadoop-hdfs-secondarynamenode
- Configure the directories needed in each node
- Start the services in each node

Note :

- Log file directory /var/log/
- Consider the permissions of data directories and log directories
- If the datanode does not work on single node installation refer the log files in /var/log/hadoop-hdfs
- If hive did not worked in hue add the following to the hue.ini file
    - In the database section add
        - option= ‘{timeout=30}’

Apache Kafka installation for cloudera quickstart

1 . Install Kafka using following commands.

```
$ sudo yum clean all
$ sudo yum install kafka
$ sudo yum install kafka-server
```

2. Start the Kafka server with the following command:

```
$ sudo service kafka-server start
```

3. Verify all nodes are correctly registered to the same ZooKeeper, connect to ZooKeeper using zookeeper-client.

```
$ zookeeper-client
$ ls /brokers/ids
```

4. Start Zoo-keeper server

$ sudo service zookeeper-server start   (most of time it is already starts)

5. Create a topic named 'test'

```
 $ kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1
--topic test
```

More information :
https://kafka.apache.org/quickstart

Installation

1. Copy system into cloudera quickstart file system.
2. Create a kafka topic as above.
3. Create a flume agent with spooldir source and kafka sink.
4. Build the project using *$ mvn package.*
5. Create Knowage birt report template and export it to Knowage server.

Configuration changes

If you need to do configurations, all the configurations are available in TypesafeConf.conf file. You can change them as per your requirement.

Note : After every change, build the project using *$ mvn package*.

Execution

1. First run flume agent

```
$ flume-ng agent -c pathToAgent -f agentConfigurationFile -n agentName
```

Example:

```
$ flume-ng agent -c /opt/examples/flume/conf -f
/opt/examples/flume/conf/flumeWithSpool.conf -n agent
```

2 . Check mysql and hive server is executing

```
$ sudo service mysqld status
$ sudo service hive-server2 status
$ sudo service hive-metastore status
```

If one of them is not running, run that service.

```
$ sudo service mysqld start
$ sudo service hive-server2 start
$ sudo service hive-metastore start
```

When log files have been created copy them to HDFS continuously.
Note : Here we had to copy log files into HDFS manually because, when we transport log files to HDFS using flume HDFS sink, program did not read log records correctly.

3. Batch Process Execution

```
$ cd Documents/tap_system/batch-layer/target/appassembler/bin
$ ./batch-layer
```

4. Speed Layer Execution

```
$ cd Documents/tap_system/batch-layer/target/appassembler/bin
$ ./speed-layer
```

Then add log files continuously to the directory which is pointed by flume.

5. Serving Layer Execution

```
$ cd Documents/tap_system/batch-layer/target/appassembler/bin
$ ./serving-layer
```

Start Knowage server and access localhost:8080/knowage in browser. Login as admin using user name - biadmin and password - biadmin.