Hadoop – Different Modes of Operation

As we all know Hadoop is an open-source framework which is mainly used for storage purpose and maintaining and analyzing a large amount of data or datasets on the clusters of commodity hardware, which means it is actually a data management tool. Hadoop also posses a scale-out storage property, which means that we can scale up or scale down the number of nodes as per are a requirement in the future which is really a cool feature.

Hadoop Mainly works on 3 different Modes:

1. Standalone Mode
2. Pseudo-distributed Mode
3. Fully-Distributed Mode

1. Standalone Mode

In *Standalone Mode* none of the Daemon will run i.e. Namenode, Datanode, Secondary Name node, Job Tracker, and Task Tracker. We use job-tracker and task-tracker for processing purposes in Hadoop1. For Hadoop2 we use Resource Manager and Node Manager. Standalone Mode also means that we are installing Hadoop only in a single system. By default, Hadoop is made to run in this Standalone Mode or we can also call it as the *Local mode*. We mainly use Hadoop in this Mode for the Purpose of Learning, testing, and debugging.

Hadoop works very much Fastest in this mode among all of these 3 modes. As we all know HDFS (Hadoop distributed file system) is one of the major components for Hadoop which utilized for storage Permission is not utilized in this mode. You

can think of HDFS as similar to the file system's available for windows i.e. NTFS (New Technology File System) and FAT32(File Allocation Table which stores the data in the blocks of 32 bits). when your Hadoop works in this mode there is no need to configure the files – *hdfs-site.xml*, *mapred-site.xml*, *core-site.xml* for Hadoop environment. In this Mode, all of your Processes will run on a single JVM(Java Virtual Machine) and this mode can only be used for small development purposes.

2. Pseudo Distributed Mode (Single Node Cluster)

In Pseudo-distributed Mode we also use only a single node, but the main thing is that the cluster is simulated, which means that all the processes inside the cluster will run independently to each other. All the daemons that are Namenode, Datanode, Secondary Name node, Resource Manager, Node Manager, etc. will be running as a separate process on separate JVM(Java Virtual Machine) or we can say run on different java processes that is why it is called a Pseudo-distributed.

One thing we should remember that as we are using only the single node set up so all the Master and Slave processes are handled by the single system. Namenode and Resource Manager are used as Master and Datanode and Node Manager is used as a slave. A secondary name node is also used as a Master. The purpose of the Secondary Name node is to just keep the hourly based backup of the Name node. In this Mode,

- Hadoop is used for development and for debugging purposes both.
- Our HDFS(Hadoop Distributed File System) is utilized for managing the Input and Output processes.

- We need to change the configuration files *mapred-site.xml, core-site.xml, hdfs-site.xml* for setting up the environment.

3. Fully Distributed Mode (Multi-Node Cluster)

This is the most important one in which multiple nodes are used few of them run the Master Daemon's that are Namenode and Resource Manager and the rest of them run the Slave Daemon's that are DataNode and Node Manager. Here Hadoop will run on the clusters of Machine or nodes. Here the data that is used is distributed across different nodes. This is actually the *Production Mode* of Hadoop let's clarify or understand this Mode in a better way in Physical Terminology.

Once you download the Hadoop in a tar file format or zip file format then you install it in your system and you run all the processes in a single system but here in the fully distributed mode we are extracting this tar or zip file to each of the nodes in the Hadoop cluster and then we are using a particular node for a particular process. Once you distribute the process among the nodes then you'll define which nodes are working as a master or which one of them is working as a slave.

Configuring XML files:-

Configuration Files are the files which are located in the extracted tar.gz file in the etc/hadoop/ directory.
All Configuration Files in Hadoop are listed below,

1) HADOOP-ENV.sh->>It specifies the environment variables that affect the JDK used by Hadoop Daemon (bin/hadoop). We know that Hadoop framework is wriiten in Java and uses JRE so one of the environment variable in Hadoop Daemons is $Java_Home in Hadoop-env.sh.

2) CORE-SITE.XML->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop cluster. It informs Hadoop daemons where the NAMENODE runs in the cluster. It also informs the Name Node as to which IP and ports it should bind.

3) HDFS-SITE.XML->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop. It contains the configuration settings for NAMENODE, DATANODE, SECONDARYNODE. It is used to specify default block replication. The actual number of replications can also be specified when the file is created,

4) MAPRED-SITE.XML->>It is one of the important configuration files which is required for runtime environment settings of a Hadoop. It contains the configuration settings for MapReduce . In this file, we specify a framework name for MapReduce, by setting the MapReduce.framework.name.

5) Masters->>It is used to determine the master Nodes in Hadoop cluster. It will inform about the location of SECONDARY NAMENODE to Hadoop Daemon. The Mater File on Slave node is blank.

6) Slave->>It is used to determine the slave Nodes in Hadoop cluster. The Slave file at Master Node contains a list of hosts, one per line. The Slave file at Slave server contains IP address of Slave nodes.

The Apache Ambari project is aimed at making Hadoop management simpler by developing software for provisioning, managing, and monitoring Apache Hadoop clusters. Ambari provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.

Apache Ambari is an open-source administration tool deployed on top of Hadoop clusters, and it is responsible for keeping track of the running applications and their status. Apache Ambari can be referred to as a web-based management tool that manages, monitors, and provisions the health of Hadoop clusters.

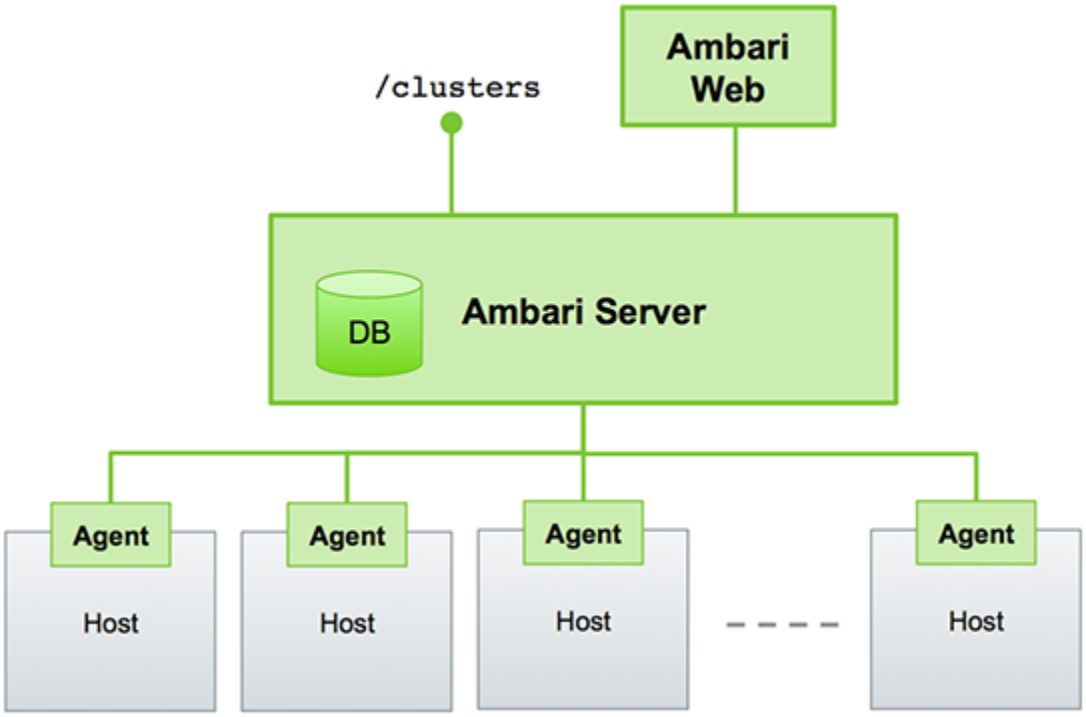
Apache Ambari collects a wide range of information from cluster nodes and services and displays that information in an easy-to-use, centralized interface known as Ambari Web.

- Instantaneous insight into the health of the Hadoop cluster using preconfigured operational metrics
- User-friendly configuration providing an easy step-by-step guide for installation
- Installation of Apache Ambari is possible through Hortonworks Data Platform (HDP)
- Monitoring dependencies and performances by visualizing and analyzing jobs and tasks
- Authentication, authorization, and auditing by installing Kerberos-based Hadoop clusters
- Flexible and adaptive technology fitting perfectly in the enterprise environment

Hadoop is a large-scale, distributed data storage and processing infrastructure using clusters of commodity hosts networked together. Monitoring and managing such complex distributed systems is not simple. To help you manage the complexity, Ambari server, agent and infrastructure components provide you operating control of hosts in the cluster as well as administrative control of cluster access.

Ambari Web displays information such as service-specific summaries, graphs, and alerts. You create and manage your cluster using Ambari Web to perform basic

operational tasks, such as starting and stopping services, adding hosts to your cluster, and updating service configurations. You also can use Ambari Web to perform administrative tasks for your cluster, such as enabling Kerberos security and performing Stack upgrades. Any user can view Ambari Web features. Users with administrator-level roles can access more options than those users with operator-level or view-only roles. For example, an Ambari administrator can manage cluster security, an operator user can monitor the cluster, but a view-only user can only access features to which an administrator grants required permissions.



The Ambari Server collects data from across your cluster. Each host has a copy of the Ambari Agent, which allows the Ambari Server to control each host.

Ambari Web is a client-side, JavaScript application that calls the Ambari REST API (accessible from the Ambari Server) to access cluster information and perform cluster operations. After authenticating to Ambari Web, the application authenticates to the

Ambari Server. Communication between the browser and server occurs asynchronously using the REST API.