Table of Contents

Big Data & hadoop Ecosystem 2

*The Need for Hadoop…* 2

Hadoop 1.x Vs 2.x 3

*Hadoop 1x 3*

*Hadoop 2x* 3

The Hadoop Ecosystem 4

Hadoop Ecosystem components 6

 ***HDFS*** *6*

** ***MapReduce*** *6*

** ***HBase*** *6*

** ***Zookeeper*** *7*

** ***Oozie*** *7*

** ***Pig*** *7*

** ***Hive*** *7*

** ***Sqoop*** *7*

** ***Flume*** *7*

** ***Whirr*** *8*

** ***Mahout*** *8*

** ***BigTop*** *8*

** ***Ambari*** *8*

Hadoop Distributions 8

***Different Hadoop Vendors*** 9

Understanding Hadoop Clusters and the Network 12

OS – Prerequisites – Hadoop 12

Installation 17

Configuration 17

Trobleshooting 17

High Availability 18

Security 18

# Big Data & hadoop Ecosystem

Hadoop is a framework used to handle big data for storing, processing and analyzing data that was previously ignored due to the limitations of traditional data management technologies.

## The Need for Hadoop…

* Hadoop fills several important needs in your data storage and processing infrastructure:
* Store and use all types of data: Allows semi-structured, unstructured and structured data to be processed in a way to create new insights of significant business value.
* Process all the data: Instead of looking at samples of data or small sections of data, organizations can look at large volumes of data to get new perspective and make business decisions with higher degree of accuracy.
* Scalability: Reducing latency in business is critical for success. The massive scalability of Big Data systems allow organizations to process massive amounts of data in a fraction of the time required for traditional systems.
* Commodity hardware: Self-healing, extremely scalable, highly available environment with cost-effective commodity hardware.

# Hadoop 1.x Vs 2.x

## Hadoop 1x

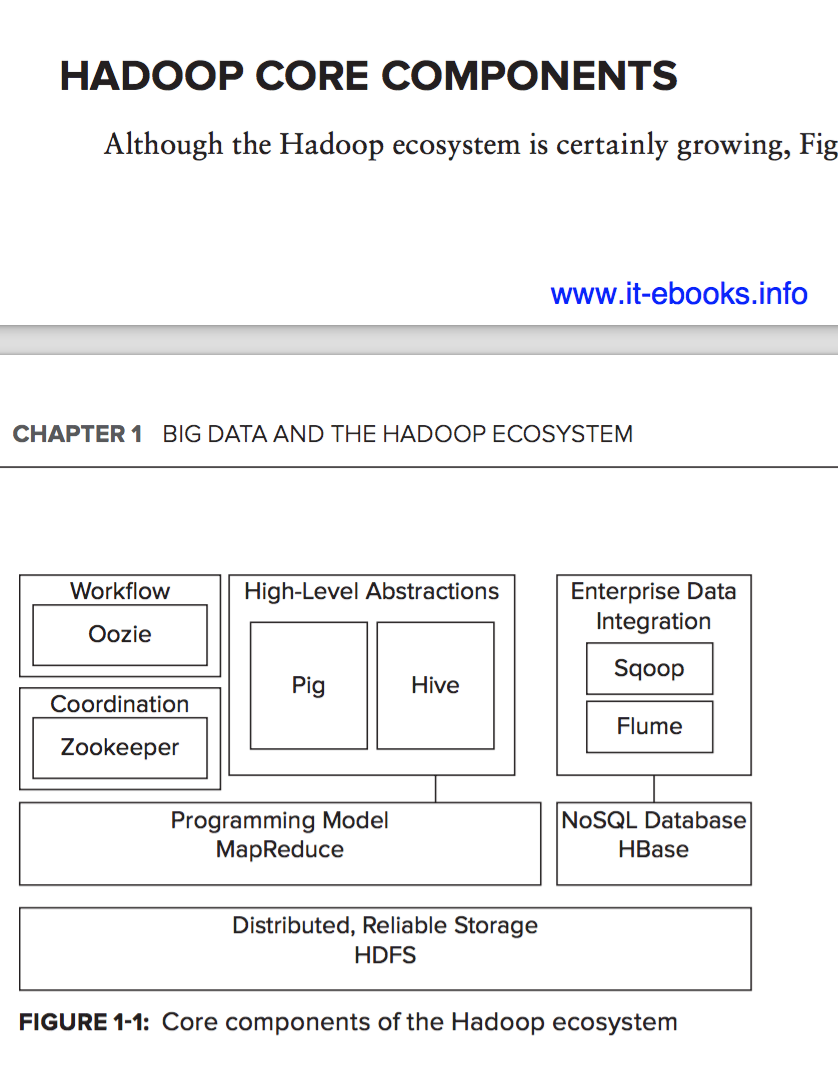
* Limited up to 4000 nodes per cluster.
* Job tracker bottleneck- resource management, job scheduling and monitoring.
* Only has one namespace for managing HDFS.
* Map and Reduce slots are static.
* Only job to run is Map Reduce.
* It’s having the concept of slots.
* Hadoop 1 allows us to write programs in only MR

## Hadoop 2x

* Potentially up to 10,000 nodes per cluster.
* Supports multiple namespace for managing HDFS.
* Efficient cluster utilization (YARN).
* MRv1 backward and forward compatible.
* Any apps can integrate with hadoop.
* It’s having the concept of container.
* Hadoop 2 which has resource management framework YARN
* MRAdmin has been removed in MRv2 because MRadmin commands no longer exist. They have been replaced by the commands in rmadmin.

The most salient advantages of Hadoop 2 over Hadoop 1 is the improved reliability and speed improvements that come with the improvements to HDFS federation and the introduction of YARN, which separates processing management from resource management.

## The Hadoop Ecosystem



* The Hortonworks Data Platform, or HDP for short, is the only 100% open source data management platform for Apache Hadoop, and is the most stable and reliable Apache Hadoop distribution. It delivers the cost-effectiveness of Hadoop and the advanced services required for enterprise deployments. In this class, we will discuss all of them, but with some more depth for Pig, Hive, Sqoop, Oozie, MapReduce, as these are key frameworks.
* HDP can be installed and is supported on RedHat/CentOS Linux, Windows Server, Suse, and Oracle Linux.
* The key features of HDP include:
* High Availability: HA is now achieveable in HDP 2.1 without the use of an outside technology.
* Open Source Cluster Management: HDP includes Apache Ambari, the only open source operations tools that allows you to provision, manage and monitor a Hadoop cluster of any size.
* Metadata Services & HCatalog: HCatalog provides metadata services and a REST interface that provides an additional SQLlike interface to Hadoop.
* Data Integration Services: including Sqoop, Flume and WebHDFS.
* ODBC Done Right: Hive has a free high-performance ODBC driver that includes a SQL engine so you can interact with nearly every BI tool, including all SQL-92 interfaces.

## Hadoop Ecosystem components

* **HDFS** — A foundational component of the Hadoop ecosystem is the Hadoop Distributed File System (HDFS). HDFS is the mechanism by which a large amount of data can be distributed over a cluster of computers, and data is written once, but read many times for analytics. It provides the foundation for other tools, such as HBase.
* **MapReduce** — Hadoop’s main execution framework is MapReduce, a programming model for distributed, parallel data processing, breaking jobs into mapping phases and reduce phases (thus the name). Developers write MapReduce jobs for Hadoop, using data stored in HDFS for fast data access. Because of the nature of how MapReduce works, Hadoop brings the processing to the data in a parallel fashion, resulting in fast implementation.
* **HBase** — A column-oriented NoSQL database built on top of HDFS, HBase is used for fast read/write access to large amounts of data. HBase uses Zookeeper for its management to ensure that all of its components are up and running.
* **Zookeeper** — Zookeeper is Hadoop’s distributed coordination service. Designed to run over a cluster of machines, it is a highly available service used for the management of Hadoop operations, and many components of Hadoop depend on it.
* **Oozie** — A scalable work ow system, Oozie is integrated into the Hadoop stack, and is used to coordinate execution of multiple MapReduce jobs. It is capable of managing a signi cant amount of complexity, basing execution on external events that include timing and presence of required data.
* **Pig** — An abstraction over the complexity of MapReduce programming, the Pig platform includes an execution environment and a scripting language (Pig Latin) used to analyze Hadoop data sets. Its compiler translates Pig Latin into sequences of MapReduce programs.
* **Hive** — An SQL-like, high-level language used to run queries on data stored in Hadoop, Hive enables developers not familiar with MapReduce to write data queries that are translated into MapReduce jobs in Hadoop. Like Pig, Hive was developed as an abstraction layer, but geared more toward database analysts more familiar with SQL than Java programming.

The Hadoop ecosystem also contains several frameworks for integration with the rest of the enterprise:

* **Sqoop** — is a connectivity tool for moving data between relational databases and data warehouses and Hadoop. Sqoop leverages database to describe the schema for the imported/ exported data and MapReduce for parallelization operation and fault tolerance.
* **Flume** — is a distributed, reliable, and highly available service for ef ciently collecting, aggregating, and moving large amounts of data from individual machines to HDFS. It is based on a simple and exible architecture, and provides a streaming of data ows. It leverages a simple extensible data model, allowing you to move data from multiple machines within an enterprise into Hadoop.
* **Whirr** — This is a set of libraries that allows users to easily spin-up Hadoop clusters on top of Amazon EC2, Rackspace, or any virtual infrastructure.
* **Mahout** — This is a machine-learning and data-mining library that provides MapReduce implementations for popular algorithms used for clustering, regression testing, and statistical modeling.
* **BigTop** — This is a formal process and framework for packaging and interoperability testing of Hadoop’s sub-projects and related components.
* **Ambari** — This is a project aimed at simplifying Hadoop management by providing support for provisioning, managing, and monitoring Hadoop clusters.

# Hadoop Distributions

Although Hadoop is a set of open source Apache (and now GitHub) projects, a large number of companies are currently emerging with the goal of helping people actually use Hadoop. Most of these companies started with packaging Apache Hadoop distributions, ensuring that all the software worked together, and providing support. And now they are developing additional tools to simplify Hadoop usage and extend its functionality. Some of these extensions are proprietary and serve as differentiation. Some became the foundation of new projects in the Apache Hadoop family. And some are open source GitHub projects with an Apache 2 license. Although all of these companies started from the Apache Hadoop distribution, they all have a slightly different vision of what Hadoop really is, which direction it should take, and how to accomplish it.

One of the biggest differences between these companies is the use of Apache code. With the exception of the MapR, everyone considers Hadoop to be de ned by the code produced by Apache projects. In contrast, MapR considers Apache code to be a reference implementation, and produces its own implementation based on the APIs provided by Apache. This approach has allowed MapR to introduce many innovations, especially around HDFS and HBase, making these two fundamental Hadoop storage mechanisms much more reliable and high-performing. Its distribution additionally introduced high-speed Network File System (NFS) access to HDFS that signi cantly simpli es integration of Hadoop with other enterprise applications.

Two interesting Hadoop distributions were released by Amazon and Microsoft. Both provide a prepackaged version of Hadoop running in the corresponding cloud (Amazon or Azure) as Platform as a Service (PaaS). Both provide extensions that allow developers to utilize not only Hadoop’s native HDFS, but also the mapping of HDFS to their own data storage mechanisms (S3 in the case of Amazon, and Windows Azure storage in the case of Azure). Amazon also provides the capability to save and restore HBase content to and from S3.

### **Different Hadoop Vendors**

|  |  |
| --- | --- |
| **Vendor** | **Hadoop Characteristics** |
| **Cloudera CDH, Manager, and Enterprise** | Based on Hadoop 2, CDH (version 4.1.2 as of this writing) includes HDFS, YARN, HBase, MapReduce, Hive, Pig, Zookeeper, Oozie, Mahout, Hue, and other open source tools (including the real-time query engine — Impala). Cloudera Manager Free Edition includes all of CDH, plus a basic Manager supporting up to 50 cluster nodes. Cloudera Enterprise combines CDH with a more sophisticated Manager supporting an unlimited number of cluster nodes, proactive monitoring, and additional data analysis tools. |
| **Hortonworks Data Platform** | Based on Hadoop 2, this distribution (Version 2.0 Alpha as of this writing) includes HDFS, YARN, HBase, MapReduce, Hive, Pig, HCatalog, Zookeeper, Oozie, Mahout, Hue, Ambari, Tez, and a real- time version of Hive (Stinger) and other open source tools. Provides Hortonworks high-availability support, a high-performance Hive ODBC driver, and Talend Open Studio for Big Data. |
| **MapR** | Based on Hadoop 1, this distribution (Version M7 as of this writing) includes HDFS, HBase, MapReduce, Hive, Mahout, Oozie, Pig, ZooKeeper, Hue, and other open source tools. It also includes direct NFS access, snapshots, and mirroring for “high availability,” a proprietary HBase implementation that is fully compatible with Apache APIs, and a MapR management console. |
| **IBM InfoSphere BigInsights** | As of this writing, this is based on Hadoop 1 and available in two editions. The Basic Edition includes HDFS, Hbase, MapReduce, Hive, Mahout, Oozie, Pig, ZooKeeper, Hue, and several other open source tools, as well as a basic version of the IBM installer and data access tools. The Enterprise Edition adds sophisticated job management tools, a data access layer that integrates with major data sources, and BigSheets (a spreadsheet-like interface for manipulating data in the cluster). |
| **GreenPlum’s Pivotal HD** | As of this writing, this is based on Hadoop 2, and includes HDFS, MapReduce, Hive, Pig, HBase, Zookeeper, Sqoop, Flume, and other open source tools. The proprietary advanced Database Services (ADS) powered by HAWQ extends Pivotal HD Enterprise, adding rich, proven, parallel SQL processing facilities. |
| **Amazon Elastic MapReduce (EMR)** | As of this writing, this is based on Hadoop 1. Amazon EMR is a web service that enables users to easily and cost-e ectively process vast amounts of data. It utilizes a hosted Hadoop framework running on the web-scale infrastructure of Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3). It includes HDFS (with S3 support), HBase (proprietary backup recovery), MapReduce, Hive (added support for Dynamo), Pig, and Zookeeper. |
| **Windows Azure HDInsight** | Based on the Hortonworks Data Platform (Hadoop 1), this runs in the Azure cloud. It is integrated with the Microsoft management console for easy deployment and integration with System Center. It can be integrated with Excel through a Hive Excel plug-in. It can be integrated with Microsoft SQL Server Analysis Services (SSAS), PowerPivot, and Power View through the Hive Open Database Connectivity (ODBC) driver. The Azure Marketplace empowers customers to connect to data, smart mining algorithms, and people outside of the users’ firewalls. Windows Azure Marketplace o ers hundreds of data sets from trusted third-party providers. |

# Understanding Hadoop Clusters and the Network

[**http://bradhedlund.com/2011/09/10/understanding-hadoop-clusters-and-the-network/**](http://bradhedlund.com/2011/09/10/understanding-hadoop-clusters-and-the-network/)

# OS – Prerequisites – Hadoop

* **CentOS v6.x (64 bit)**
* **Software requirements:**

CentOS v6.x: Python 2.6. \*

* **JDK**:

|  |  |
| --- | --- |
| Stack | JDKs |
| * HDP 2.3 or 2.4 | * JDK 1.7 or JDK 1.8 |
| * HDP 2.2 | * JDK 1.7 |

* **Memory Requirements**:

|  |  |  |
| --- | --- | --- |
| No. of hosts | Memory Available | Disk Space |
| * 1 | * 1024 MB | * 10 GB |
| * 10 | * 1024 MB | * 20 GB |

* **Check the Maximum Open File Descriptors:**

The recommended maximum number of open file descriptors is 10000, or more. To check the current value set for the maximum number of open file descriptors, execute the following shell commands on each host:

ulimit -Sn

ulimit -Hn

If the output is not greater than 10000, run the following command to set it to a suitable default:

ulimit -n 10000

* **FQDN (Fully Qualified Domain Name):**

The fully qualified domain name (FQDN) of each host in your system. The Ambari install wizard supports using IP addresses. You can use hostname -f to check or verify the FQDN of a host.

* **Set Up Password-less SSH:**

To have Ambari Server automatically install Ambari Agents on all your cluster hosts, you must set up password-less SSH connections between the Ambari Server host and all other hosts in the cluster. The Ambari Server host uses SSH public key authentication to remotely access and install the Ambari Agent.

* **Enable NTP on the Cluster and on the Browser Host:**

The clocks of all the nodes in your cluster and the machine that runs the browser through which you access the Ambari Web interface must be able to synchronize with each other.

To check that the NTP service will be automatically started upon boot, run the following command on each host:

RHEL/CentOS/Oracle 6

chkconfig --list ntpd

To set the NTP service to auto-start on boot, run the following command on each host:

RHEL/CentOS/Oracle 6

chkconfig ntpd on

To start the NTP service, run the following command on each host:

RHEL/CentOS/Oracle 6

service ntpd start

* **Check DNS and NSCD:**

All hosts in your system must be configured for both forward and and reverse DNS.

If you are unable to configure DNS in this way, you should edit the /etc/hosts file on every host in your cluster to contain the IP address and Fully Qualified Domain Name of each of your hosts. The following instructions are provided as an overview and cover a basic network setup for generic Linux hosts. Different versions and flavors of Linux might require slightly different commands and procedures. Please refer to the documentation for the operating system(s) deployed in your environment.

Hadoop relies heavily on DNS, and as such performs many DNS lookups during normal operation. To reduce the load on your DNS infrastructure, it's highly recommended to use the Name Service Caching Daemon (NSCD) on cluster nodes running Linux. This daemon will cache host, user, and group lookups and provide better resolution performance, and reduced load on DNS infrastructure.

* **Configuring iptables:**

For Ambari to communicate during setup with the hosts it deploys to and manages, certain ports must be open and available. The easiest way to do this is to temporarily disable iptables, as follows:

RHEL/CentOS/Oracle Linux 6

chkconfig iptables off

/etc/init.d/iptables stop

* **Disable SELinux and Package Kit and check the umask Value:**

You must disable SELinux for the Ambari setup to function. On each host in your cluster,  setenforce 0

|  |  |
| --- | --- |
|  | **Note:** |
| To permanently disable SELinux set SELINUX=disabled in  /etc/selinux/config This ensures that  SELinux does not turn itself on after you reboot the machine . |

On an installation host running RHEL/CentOS with PackageKit installed, open /etc/yum/pluginconf.d/refresh-packagekit.conf using a text editor. Make the following change: enabled=0

|  |  |
| --- | --- |
|  | **Note:** |
| PackageKit is not enabled by default on Debian, SLES, or Ubuntu  systems. Unless you have specifically  enabled PackageKit, you may skip this step for a Debian, SLES, or  Ubuntu installation host. |

* **Umask:**

UMASK (User Mask or User file creation MASK) sets the default permissions or base permissions granted when a new file or folder is created on a Linux machine. Most Linux distros set 022 as the default umask value. A umask value of 022 grants read, write, execute permissions of 755 for new files or folders. A umask value of 027 grants read, write, execute permissions of 750 for new files or folders. Ambari & HDP support umask values of 022 (0022 is functionally equivalent), 027 (0027 is functionally equivalent). These values must be set on all hosts. UMASK Examples: Setting the umask for your current login session: umask 0022 Checking your current umask: umask 0022 Permanently changing the umask for all interactive users: echo umask 0022 >> /etc/profile

* **Setting Up a Local Repository:**

Setting up the local repository to download ambari and HDP repos

* **Disable Transparent Huge Pages (THP):**

When installing Ambari on CentOS6.x using the Cluster Installer Wizard at the Host Checks step, one or more host checks may fail if you have not disabled Transparent Huge Pages on all hosts. Host Checks warns you when a failure occurs.

**To disable THP:**

Add the following command to your /etc/rc.local file:

RHEL6 if test -f /sys/kernel/mm/redhat\_transparent\_hugepage/defrag; then echo never > /sys/kernel/mm/redhat\_transparent\_hugepage/defrag fi

SLES if test -f /sys/kernel/mm/transparent\_hugepage/defrag; then echo never > /sys/kernel/mm/transparent\_hugepage/defrag fi

To confirm, reboot the host. Then, run the following command: $ cat /sys/kernel/mm/transparent\_hugepage/enabled always madvise [never]

# Installation

* Configure a local HDP repository
* Install ambari-server and ambari-agent
* Install HDP using the Ambari install wizard
* Add a new node to an existing cluster
* Decommission a node
* Add an HDP service to a cluster using Ambari

# Configuration

* Define and deploy a rack topology script
* Change the configuration of a service using Ambari
* Configure the Capacity Scheduler
* Create a home directory for a user and configure permissions
* Configure the include and exclude DataNode files

# Trobleshooting

* Restart an HDP service
* View an application’s log file
* Configure and manage alerts

# High Availability

* Configure NameNode HA
* Configure ResourceManager HA
* Create a snapshot of an HDFS directory
* Recover a snapshot

# Security

* Install and configure Knox
* Install and configure Ranger
* Configure HDFS ACLs