Virtualizing Hadoop® on VMware vSphere®

A DEPLOYMENT GUIDE

vmware®

Table of Contents

Introduction

This document provides a set of approaches that can be used by systems managers, architects, and developers who intend to deploy their Hadoop® systems in virtual machines (VMs) running on VMware vSphere®.

The document can be used as a starting point for a new installation of Hadoop on vSphere or for rearchitecting an existing environment, possibly one that was previously deployed on bare metal. The example designs given here can be adapted to suit the environment, based on the particular requirements of the business and the available resources.

This document addresses the following topics:

- An overview of the Hadoop architecture
- Considerations for deploying Hadoop on vSphere
- Architecture and configuration of Hadoop systems on vSphere, with three example deployments

Overview of the Hadoop Architecture

This section presents a primer on Hadoop to readers who might be new to the concepts, those who are already familiar with the vSphere core concepts and are either embarking on a first deployment of Hadoop on vSphere or considering the various approaches to do so.

The "References" section of this paper contains more advanced technical material for those who are already experienced in the Hadoop field. Throughout this document, we will frequently refer to that advanced material rather than repeating the details. The information given in this section on Hadoop provides the technical background required for the deployment discussions in later sections of the document. Readers who are very familiar with the Hadoop architecture may skip this section.

The Hadoop distributed application platform originated in work done by engineers at Google, and later at Yahoo, to solve problems that involve storing and processing data on a very large scale in a distributed manner. The engineers initially focused on Web indexing, and on search in particular. The source code for Hadoop was contributed to the Apache community as an open-source project, and development of the core platform continues there.

Apache™ Hadoop® is a programming and execution environment as well as a file system and data storage mechanism, which together provide a framework for reliable and scalable distributed computing on a large scale. Along with the Apache Hadoop distribution, there are several commercial companies—including Cloudera, Hortonworks, IBM, MapR, and Pivotal—that provide their own supported distributions based on Apache Hadoop. These are often integrated with other Hadoop ecosystem components, both open-source and proprietary. Examples of these ecosystem components are the management tools and the SQL engines for querying Hadoop data, which differ somewhat across the various vendors. In this document, we will refer to all such implementations, including the Spark™ technology, as "the Hadoop distributions" or simply as "Hadoop." We are primarily concerned here with the deployment of the core components of Hadoop.

All of the previously mentioned Hadoop distributions run very well in VMs in a vSphere environment. There is no favorite Hadoop distribution for vSphere. Several detailed technical reports on performance testing done internally at VMware labs on a Hadoop distribution deployed on vSphere are provided in references [1], [2], [3], and [4]. To enhance the user experience on vSphere, VMware works with the Hadoop distribution companies to provide validation and support for their respective Hadoop products on the vSphere platform.

Hadoop is typically used for processing large data sets across clusters of independent machines. Hadoop clusters can scale up to thousands of machines, each participating in computation as well as file and data storage. Hadoop is adopted by companies for a wide range of custom-built and packaged applications that are designed to help businesses make better decisions through analysis of their larger, diverse data sets. Hadoop has become one of the leading "big data" platforms for storing large quantities of unstructured data. It supports the following functions:

- A range of extract, transform, and load (ETL) tools
- MapReduce
- Spark
- Event streaming
- Machine learning
- Additional functions that support analytics applications

A very broad base of tools in the ecosystem surrounds the core of Hadoop.

Hadoop systems have two major components: the Hadoop Distributed File System (HDFS™), which is used to store the data; the computational framework, where programs run to process that data. Collectively, these two can be called the *Hadoop* core. In the early days of Hadoop, its computational framework was called Hadoop MapReduce, after the main program construction style at that time. In recent years, that particular style of programming is complemented, and in many cases has been replaced, by a new programming API and execution style called Spark. The core Hadoop scheduler has also had MapReduce specifics removed from it. Spark and MapReduce are said to be "frameworks" that execute on the Hadoop core.

Many other technologies in the ecosystem use the Hadoop core as a basis: Hive™, Pig™, Sqoop, Flume, Oozie, Impala, and others. These components are outside the scope of this document. More information on these tools can be found on the respective Hadoop distribution vendors' Web sites.

The Hadoop distributed application platform uses a "divide and conquer" approach to processing large amounts of data, so as to produce a set of results. Programs executing on the Hadoop platform are inherently parallel in nature, so they are well suited to a highly distributed environment. The most current form of job scheduling, resource management, and execution for Hadoop is referred to as Hadoop YARN (Yet Another Resource Negotiator). It represents the second iteration of the Hadoop architecture. The original scheduling mechanism took a MapReduce-centric approach to scheduling, which proved to be inflexible. The redesign of the core to Hadoop YARN reworked the job scheduling system to make it more general, so as to cater to many different types of workloads.

In the Hadoop YARN architecture, the main Hadoop roles or processes are the ResourceManager and NameNode master services and the NodeManager and DataNode worker services. These are implemented as long-lived Java programs that execute on various machines or "nodes" in the Hadoop system. There are several other process roles in the architecture, such as ZooKeeper™ and JournalNode, that perform various other management functions. For the sake of brevity, we will not delve into those here. Further information on these other roles can be found in the Apache Hadoop documentation and on the respective Hadoop distribution vendors' Web sites.

The ResourceManager process allocates CPU power and memory resources, on request, to the worker processes that require them. ResourceManager handles scheduling of all jobs in the Hadoop cluster as well as the individual parts, called tasks, that make up each job. A job is split into a set of tasks that are then associated with containers that execute on the worker nodes—that is, machines—in the Hadoop cluster. There are normally many worker nodes. Dozens, hundreds, or even thousands of nodes are deployed in Hadoop clusters today. Managing these workers is typically done by the master processes running on fewer than 10 nodes in a Hadoop cluster.

The NodeManager process, which runs on each worker node, is responsible for starting containers, which are Java Virtual Machine (JVM) processes, on that node to fulfill tasks and for reporting their progress back to the central scheduler, the ResourceManager. Many containers can run simultaneously on multiple nodes. They are under the control of their local NodeManager, which can start or stop a container's execution when required.

HDFS is a Linux-like file system with hierarchical directories and files that are distributed across nodes. It is controlled by one or more NameNode processes that manage the file system namespace and the placement onto the worker nodes of the individual data blocks that compose a file. There are Secondary NameNodes, for failure tolerance, and Federated NameNodes, for larger systems in the current Hadoop architecture.

The NameNode is the master process for HDFS. It works by keeping its metadata, such as block-to-file mapping, mainly in memory, so it can require a sizable Java heap space. The NameNode works in conjunction with a set of worker processes called DataNodes that each take charge of a set of blocks of HDFS data on different machines. The DataNode's job is to write and retrieve these data blocks when instructed. HDFS manages multiple replicas of each block of data, providing resiliency against failure of a worker node or DataNode in the cluster. By default, three replicas of an HDFS block are retained, but the administrator can change this "replication factor" number.

The NameNode manages the file system namespace by maintaining a mapping of all the filenames and their associated data blocks. The NameNode keeps a list of the DataNodes that are responsible for replicating any data block in the file system.

The entire Hadoop system, therefore, is broken down into the following three types of processes and machines:

- The machines that run the ResourceManager and NameNode processes, along with their supporting processes, are called the "master" nodes of the cluster.
- The "worker" nodes usually contain the DataNode and local NodeManager processes.
- The remaining nodes, which are responsible for interacting with the end user and for job submission into the cluster, are called the "client," "gateway," or "edge" nodes. These nodes contain processes such as the Pig Client or Hive Client as well as other processes that enable jobs to be started.

In the Hadoop YARN technology, there is a Secondary NameNode that keeps snapshots of the NameNode metadata. These snapshots are used for recovery in the event of a primary NameNode failure. The lifetime and health of the NameNode must be preserved because all operations on the HDFS data depend on them.

A small Hadoop cluster can include a single master machine containing the ResourceManager and NameNode, along with multiple worker nodes, as is shown in Figure 1. These processes, however, are held on separate, dedicated machines for larger Hadoop clusters.

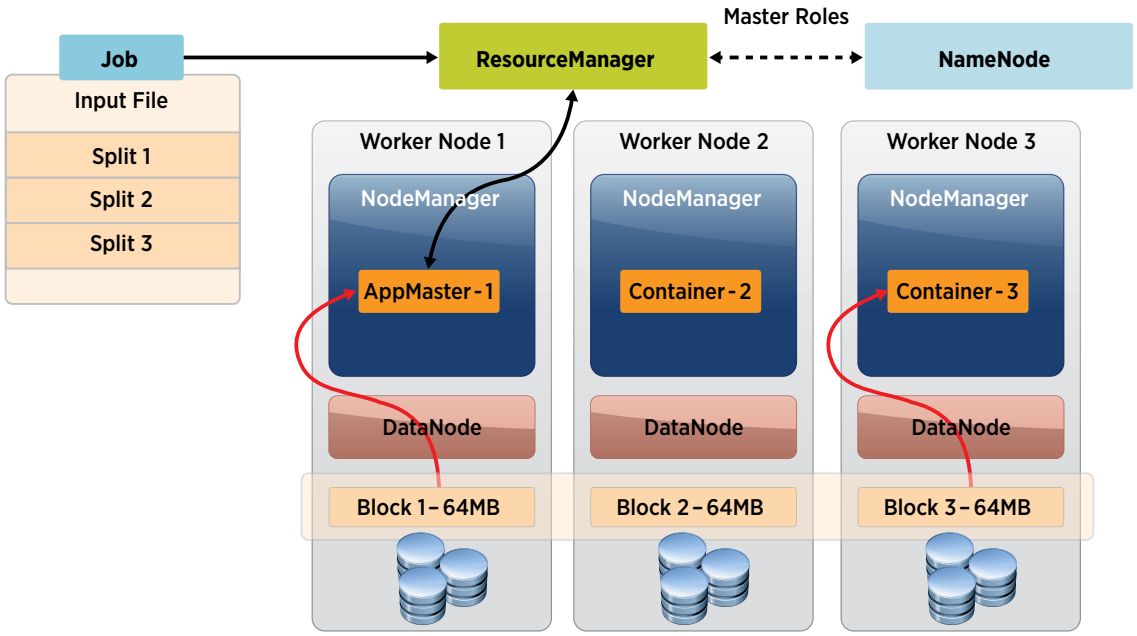

Figure 1. Components of the Hadoop Architecture

In Figure 1, the master machine or machines contain the ResourceManager and NameNode as the main Hadoop management processes or roles. Each worker machine typically contains the DataNode and NodeManager roles together. It is also possible to have various combinations of these, such as data-only worker nodes and compute-only worker nodes.

The Hadoop Architecture on vSphere

When Hadoop is virtualized, all of the components of Hadoop, including the NameNode, ResourceManager, DataNode, and NodeManager, are processes running independently within the guest operating system (OS) of a set of purpose-built VMs rather than on a native OS. These processes are sometimes referred to as the Hadoop services or daemons. The VMs contain exactly the same processes as do the physical machines and can be laid out as shown in Figure 1.

We use the terms "node" and "virtual machine" interchangeably in this document to refer to a VM containing one or more Hadoop processes.

These Hadoop processes can be installed into the VMs by the Hadoop vendor's tools, such as Cloudera Manager, Ambari™, and others. The installation of the Hadoop software should appear similar to that of native machines. Where the installation procedure formerly referred to physical hosts, VMs now contain the same processes. With virtualization infrastructure, however, two or more worker VMs can be run on the same physical host server. With virtualization, it is not necessary to purchase new hardware to start up a new worker VM in a Hadoop cluster if the physical capacity for it exists. This can save considerable time when, for example, expanding a cluster.

Some of these VMs might execute together on the same vSphere server host. This means that the topology of a virtualized Hadoop system is treated in a somewhat different way. An important consideration is whether the replicas of a data block in HDFS might be placed on two VMs within the same server host. The separation of data block replicas to VMs on different hosts is achieved via a functionality called Hadoop Virtualization Extensions (HVE), which VMware contributed to the Apache Hadoop code. These extensions will be described later in this document.

In a native implementation of a large Hadoop cluster, HDFS can be managed by a NameNode on a dedicated server. The NameNode manages the HDFS file system metadata, including the mapping of HDFS data blocks to individual files and the ownership of those data blocks by particular DataNode processes. When virtualized, the NameNode process can run on one VM in isolation, or other processes can run alongside it in separate VMs on the same host server if there is capacity available.

A ResourceManager process, running natively on its own server, has also been used in some native installations to manage job scheduling, resource allocation, and other functions. In a vSphere environment, this is done by running the NameNode process in one VM and the ResourceManager process in a separate VM, most likely on separate host servers. In a smaller test or experimental cluster, they might be run on the same server host. For security purposes, the secondary, standby versions of these processes might also exist in separate VMs on separate host servers to provide a more resilient Hadoop cluster.

In some Hadoop cluster deployments, the HDFS-specific components—that is, the NameNode, Secondary NameNode, and DataNode processes—are replaced by vendor solution–specific components that provide the HDFS APIs in a compatible manner. This is the case with the Dell EMC® Isilon® network-attached storage (NAS) form of Hadoop support as well as with the MapR distribution.

In the Isilon type of deployment, the functionalities of the NameNode and DataNode processes are supplied by software functions that exist on the Isilon OS, which is called OneFS®. The consumers of the HDFS APIs or RPCs—that is, the NodeManagers and containers that require access to files and data—get their data requests fulfilled by these OneFS components in the same way as if they were the NameNode and DataNode themselves. There is no difference in the functionality. This separation between processing and storage yields significant benefits in data ingestion and in reducing overall data storage needs. Business data can be loaded via NFS, CIFS, SMB, HTTP, FTP, or other protocols and then made visible through HDFS, without the need for a further ingestion step.

When an Isilon NAS system handles the HDFS portion of Hadoop, the VMs on a vSphere system are described as "compute-only" nodes—that is, they are hosting the NodeManager and execution containers, along with a separate machine for the master role ResourceManager, but no HDFS storage handling is done here in the VMs. This approach enables more "compute-only" VMs to be placed on the host servers.

Why Virtualize Hadoop?

The following are among the many benefits of virtualizing Hadoop on vSphere:

- Better resource utilization: Dedicating a potentially large set of hardware servers to the exclusive use of one Hadoop cluster is wasteful of resources. This is particularly true if the Hadoop cluster is not fully occupying the hardware all of the time. There frequently are opportunities for sharing a set of hardware servers and their storage across more than one Hadoop cluster.

  Often, enterprises have at least a development Hadoop cluster, a preproduction staging Hadoop cluster, and a production Hadoop cluster. The traditional view has been that each of these is allocated a dedicated hardware set of servers. With virtualization, there is the flexibility to share the hardware for these Hadoop clusters.

  More than likely, the version of the Hadoop software for the development cluster is more current than that of the others, because developers like to work with the newer versions. This is a second argument for virtualizing Hadoop. Dedicating a set of hardware to one version of a Hadoop vendor's product does not make the best use of resources.

  Colocating Hadoop VMs on host servers with VMs supporting different workloads is also possible, particularly for situations that are not performance critical. Doing this can balance the use of the system. This often enables better overall utilization by consolidating applications that either use different kinds of hardware resources or use the hardware resources at different times of the day or night.

- Alternative storage options: Originally, Hadoop was developed with local, direct-attached storage (DAS) in mind. This DAS storage scheme can also be used with vSphere and has been used in production virtualized Hadoop clusters for some time. The shared storage that is very commonly used by customers as a basis for vSphere can also be leveraged under certain circumstances for Hadoop workloads.

  For smaller Hadoop clusters, with 10 or fewer host servers, the data and compute tiers that compose a Hadoop application can be held entirely on shared storage. As medium- and larger-sized clusters are built, a hybrid storage model enables better performance. The hybrid storage model is one in which parts of the Hadoop infrastructure use DAS and other parts use storage area network (SAN)–type or VMware vSAN™ storage. The NameNode and ResourceManager processes use a small amount of storage resources and, for purposes of reliability, can be placed on SAN or vSAN storage. When the design goals are to achieve higher utilization and higher Hadoop performance, one reasonable approach is to put the HDFS data and the temporary data on DAS with an option to place the HDFS data on NAS-type shared storage such as Isilon.

- Isolation: This includes running different versions of Hadoop itself on the same hardware cluster or running Hadoop alongside other applications, forming an elastic environment potentially to be used by various Hadoop tenants.
- Availability and fault tolerance: vSphere features such as VMware vSphere High Availability (vSphere HA) and VMware vSphere Fault Tolerance (vSphere FT) can protect the Hadoop components from server failure and improve availability. Resource management tools such as VMware vSphere vMotion® can provide availability during planned server downtime and maintenance windows.
- Efficiency: VMware enables easy and efficient deployment of Hadoop on an existing virtual infrastructure as well as consolidation of otherwise dedicated Hadoop cluster hardware into a data center or cloud environment.
- Rapid provisioning: vSphere enables rapid deployment, management, and scalability of Hadoop in virtual and cloud environments. Virtualization tools ranging from simple cloning to sophisticated end-user provisioning products such as VMware vRealize® Automation™ can speed up the deployment of Hadoop, which in most cases requires multiple nodes with different configurations in them. The cluster provisioning steps can be performed in parallel across several newly created VMs, reducing deployment time.

A virtualized infrastructure, with suitable automation added, enables users to achieve on-demand creation of Hadoop cluster instances.

Prerequisites

Before beginning deployment of a Hadoop cluster, check that the following requirements are fulfilled:

- Carefully calculate the required data storage space

  a) for data that is stored in HDFS

  b) for data stored outside of HDFS (temporary, buffer-spill, or shuffle data that is used between the phases of the Hadoop processing algorithms)

  HDFS stores both the input and output data for the application in one or more files. Calculate the minimum data space required by multiplying the input and output data sizes by their respective replication factor—the default is 3—and adding the temporary space needed. The number of data replicas can be different for separate applications. The size of the temporary data can be larger than the size of the input or output data. The temporary data most often is kept in a storage area separate from that containing the HDFS data, so they must be sized separately.
- Perform checks on all aspects of the networking on the system so that all hosts are identified correctly on the network and can communicate with each other. Make sure that forward and reverse lookup in DNS is available for all virtual and physical machines and that an effective mechanism for getting IP addresses, such as DHCP, and host names exists for all the VMs being created.

- Ensure that there is sufficient physical memory. No process swapping should occur at either the VMware ESXi™ level or guest OS level. To avoid swapping at the ESXi level, ensure that there is adequate physical memory for the demands of all VMs on the ESXi host server as well as of the hypervisor itself. For a full discussion of this topic, see reference [6].
- Ensure that there is enough virtual memory available to the VMs. To avoid swapping at the guest OS level within the VM, ensure that the VM is configured with enough memory for all its resident processes and OS needs. Pay particular attention to the Java heap space, the buffer cache space, and other requirements on memory of the processes that will be run. Consult with the Hadoop distribution vendor to get recommendations on the sizes of memory needed for the various processes that will be run.
- Finally, decide on the number of physical server hosts to use and the number of VMs to use on those servers. Using a small number of host servers—10 or fewer—is not uncommon in the early stages of testing Hadoop on vSphere. This is recommended by the Hadoop vendors as a good way to get started. The number of VMs can begin at two per host server and then increase later if appropriate. Some serious production deployments, discussed later in this document, have standardized at two VMs per host server, so this is a good starting point for testing.

After creating a Hadoop cluster, a good quality assurance step is to run a sample self-contained application to ensure the health of the cluster. Monitor the performance of the cluster by using a Hadoop management or monitoring tool, such as Cloudera Manager or Ambari. These sample test programs are supplied by the Hadoop vendor along with their Hadoop distributions. To test MapReduce-style applications, the TeraSort suite can be useful. To test Spark or other programming styles, choose other types of test programs to ensure that the cluster is well set up.

Reference Architecture

This section focuses on the technical considerations for virtualizing a Hadoop cluster and provides a set of choices for building various sizes of Hadoop systems. Because infrastructure constraints will vary from one enterprise to another, it is not the intent here to provide a prescriptive, single option for the system architecture.

In addition, the applications built and run on Hadoop platforms can differ widely in how they use various Hadoop ecosystem services and how they consume the computing resources provided. MapReduce applications are quite different from Spark-based ones in their consumption of resources. ETL and ingest-heavy applications place different pressures on the infrastructure than do machine learning algorithms, for example. These two application types also deal with different quantities of data.

The Hadoop cluster's infrastructure features as managed by vSphere must be adapted to the particular application job types hosted and the services provided.

This section consists of three parts. The first part outlines the requirements that must be met before proceeding into the architecture work.

The second part is a set of mappings of the Hadoop services described earlier—NameNode, DataNode, ResourceManager, and so on—to their respective VMs. Two types of storage mechanisms are discussed here—DAS and NAS—for the HDFS data and other types of data that are used in applications. There are trade-offs to consider when choosing a storage strategy.

The third part discusses system configuration—including sizing factors, availability, hardware choices, and network infrastructure—as a basis for forming a particular architecture. Options are presented, along with their respective trade-offs, because no single architecture addresses every situation.

Understanding Application Requirements

It is important to estimate the resource consumption of an existing or proposed Hadoop cluster and to do so early in the project's life cycle. This provides the information needed to size the system correctly, an essential part of the architecture process. Further technical details on the various deployment models will be discussed in later sections of this document.

The person responsible for deploying a Hadoop-based system might be required to move a previously native Hadoop cluster over to vSphere or to fit the new system into an existing architecture in the data center. As a result, there are constraints to work with, such as preexisting storage or network considerations.

To begin the task of building a suitable virtualized Hadoop cluster, first accumulate the sizing figures regarding the data itself, along with the types of applications, queries, and traffic that will be using it. This information comes from conversations with users, application design and development teams, and the vendor of the Hadoop or application software. The following are among the types of data that are required as input to the process:

- The amount of input data that will be ingested into the Hadoop cluster as well as its frequency
- The expected growth rate of that data over time
- The amount of temporary data that will be required to be stored between the stages of the application's processing—that is, the shuffle and spill space
- The amount of data that will be written to HDFS
- The throughput and bandwidth required for read and write of all data: megabytes/second read and written
- The replication factor of the data: how many times the data blocks will be copied and stored
- How large the HDFS data blocks should be
  - Many sites favor 128MB, 256MB, or even larger-sized data blocks.
- How the business application's performance characteristics are expected to compare with known workloads
  - Comparisons can be drawn with CPU-intensive, I/O-intensive, or network-intensive applications such as the TeraGen, TeraSort, TeraValidate, or BigBench test applications.
- The mix of applications to be deployed in the Hadoop cluster, with the information gathered as previously listed for each application
- The priorities of the various applications that will be hosted in one or more Hadoop clusters
- The multitenancy factors and how resources must be allocated to each tenant

After this information has been collected and analyzed, there is a solid basis for making the key design decisions for the Hadoop clusters.

Deployment Models for Hadoop Clusters

This section presents a set of technical options for deployment of the main Hadoop components. It also provides a set of basic principles for choosing between them. All these models have been successfully deployed on vSphere systems.

Figure 2 outlines the basic topology choices. There are more-complex variants of these, but we explore the basics first. Here, the term *compute* refers to the NodeManager process and any containers or executors that it manages in the Hadoop architecture; the term *data* refers to the DataNode process.

We make the assumption for now that the NameNode and ResourceManager master processes along with the other master processes are held in separate VMs located elsewhere in the system. These processes can run on dedicated servers for holding the master roles. In some cases, the master VMs might share their host servers with other, lightweight consumer processes contained in VMs of their own. For now, we concentrate on the worker VMs, which will be in the majority.

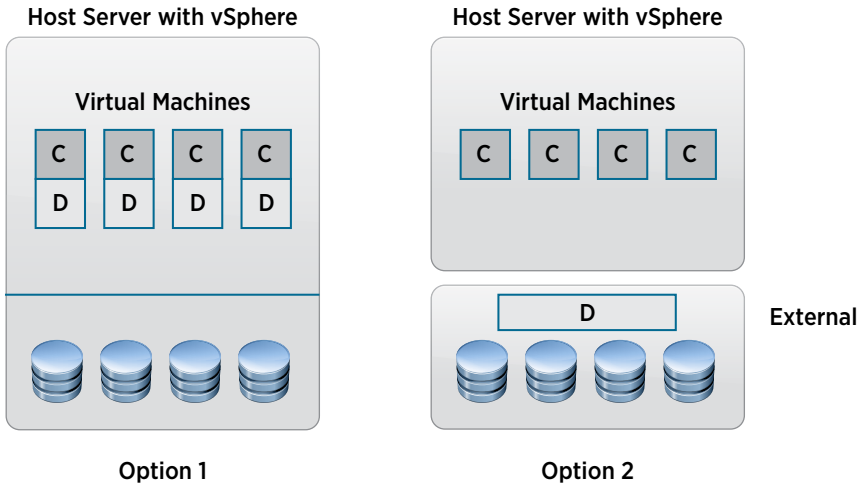

Figure 2. Choices for Deployment of the Hadoop Worker Processes on Virtual Machines

In option 1 of Figure 2, the compute (C) and data (D) roles are combined into one VM. In option 2, the data/storage functions (D) are removed from the VMs and held elsewhere, such as on a NAS-supported HDFS system or another "external" storage mechanism. These models have different benefits, which will be discussed in the following sections along with a review of their architectural variants. There are other useful models, such as those described in references [7] and [8], but those are less commonly used today, so we concentrate on the more popular deployment models here.

Architecture 1: The Combined Compute and Data Model

In this architecture, the NodeManager and DataNode Hadoop processes run together in a VM just as they would in a physical machine. This is the "standard" approach to deploying Hadoop, but now these processes are in a VM rather than on a physical machine. There can be multiple VMs of this type running together on any one vSphere host server. The number of VMs per server is limited only by the physical resources available and the minimum demands of the processes within the VMs.

This architecture is used extensively with varying VM-to-host-server ratios in the performance tests described in references [1], [2], [3], and [4]. This combined model is in common use today at enterprises that virtualize their Hadoop clusters. It is very familiar to the technical architects at the Hadoop vendors, and it presents distinct advantages in that respect. There are also advantages in costs, because the cost per gigabyte of DAS disks is low in comparison with other schemes. However, this is not the only model for deploying virtualized Hadoop that we consider.

For the purposes of illustrating this example architecture, Figure 3 shows a vSphere virtualization host server that has 12 local DAS disks configured in it. This number of disks is just by way of example. There can be twice that number of disks, or more, in contemporary rack-mounted servers. If no local disk storage is available, there are alternatives, such as SSD and flash storage, for deploying the model.

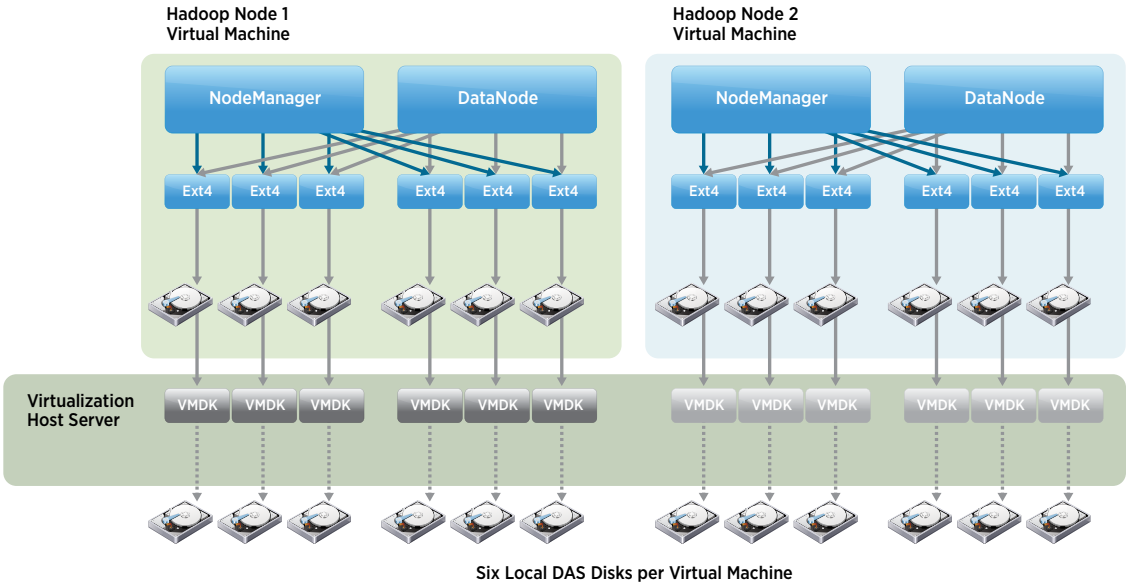

Figure 3. Multiple Virtual Machines Containing the Hadoop Worker Processes on a Virtualization (ESXi) Host Server

In this example, there are 12 DAS disk spindles directly attached to the vSphere host server. Each disk spindle is presented to vSphere as a datastore. There is then one virtual disk (VMDK) file contained in each vSphere datastore, and that VMDK file is associated with one VM. This means that the VM has exclusive access to that datastore and disk spindle, thereby avoiding access contention. If contention is not a concern, VMDK files from different VMs can be placed into the same datastore. The best practice is to constrain any one disk device or datastore to belong to one VM.

The NodeManager and DataNode processes access their own individual OS directories for reading and writing data. These directories map down to the physical disks through the VMDK and datastore mechanisms in vSphere. The guest OS and swap disks for the VMs can be placed on the extra DAS disks on the server—that is, those disks that are not in use for storing the Hadoop data. For the purpose of simplicity, this is omitted in Figure 3. The guest OS and swap disks can be held on separate shared types of storage if necessary. When no shared storage is available, the OS and swap disks can be placed on a vSphere datastore that is on local disks configured with suitable RAID protection, to ensure that the OS is protected against disk failure.

Deploying multiple VMs concurrently on a vSphere host server in this way enables better hardware resource utilization. Performance tests on this architecture, described in references [1], [2], [3], and [4] have shown that multiple Hadoop worker nodes per host, where each node is a VM, can bring significant benefits to overall system performance. The DAS disks are divided evenly between the VMs; they are exclusively accessed by their "owner" VM.

In development and test environments for Hadoop, where achieving the best performance is not a premium concern, that dedication of disks to individual VMs might not be needed. There, the VMs can mix together their access to the disks. But for more serious testing and production use, the separation shown in Figure 3 is recommended.

The separation of disk access shown in Figure 3 minimizes contention for the same disk between the two VMs on the same host server. A set number of disks can be allocated in this way exclusively to one VM. The recommendation is to allocate six or more physical disks to any one VM that is configured in this way.

Master Processes – Support on Direct-Attached Storage

The master Hadoop roles, primarily the NameNode and the ResourceManager, are normally placed into separate VMs from the worker processes. Several other master processes, such as ZooKeeper and JournalNode, run in VMs with them. This mixing of roles into VMs is done in the same way as on a server for a native implementation. The reference architecture described in reference [1] provides more technical details on this.

These master Hadoop processes can utilize VMs that benefit from the use of vSphere HA and vSphere FT by having their data contained on shared storage. More information is provided on this subject in reference [13]. If a shared storage mechanism such as SAN, vSAN, or NAS is not available, the use of vSphere HA and vSphere FT is not applicable.

An alternate safety strategy is to set aside a subset of the local DAS disks to be used for the guest OS and swap disks for the master VMs. These particular disks are organized so as to have higher levels of failure handling by configuring two local disks as RAID 1 or higher levels, such as RAID 10. Place a vSphere datastore on those disks. The VMDK files for the NameNode and ResourceManager VMs are then placed on this protected datastore. This technique for the master processes is explored further in references [1] and [4].

As an alternative approach to the combined NodeManager and DataNode on one VM model, the user can specify that the DataNode and NodeManager roles are to be run in separate VMs. This separation of the processes provides the benefit of making the lifetime of the NodeManager process independent of the lifetime of the DataNode process. Cloning a NodeManager-only VM does not instantiate a DataNode process with it in this case. This can provide advantages in avoiding data rebalancing when adding more NodeManager VMs. When using DAS, this is a much less commonly used approach today, although it is feasible. We do not go into that "separated model" architecture here; it is explored in more detail in references [7] and [8].

Storage for the Application-Specific Temporary Data

Each Hadoop worker node, now operating within a VM, has its own temporary or shuffle data space, the size of which should be planned ahead of time. This temporary data size is closely related to the design of the specific applications that are using Hadoop. This is not HDFS data, but it is used as supporting data to complete the job, such as holding spills from memory buffers or shuffling data between components of the distributed system. In native deployments, temporary data is generated by the Hadoop services onto OS directories on local disks. For example, the TeraSort Hadoop benchmark application requires a temporary data space size that is at least twice its input data set size. On the other hand, the TeraGen and TeraValidate applications from the same test suite do not require significant temporary data space. This part of the design requires careful measurement before the Hadoop cluster is constructed.

Studies on the use of the storage bandwidth by various Hadoop-based applications have shown that a significant portion of the overall I/O bandwidth—more than 50 percent in the case of certain older MapReduce applications—is consumed in reading and writing to the temporary data. This is data that is used by applications during the sort and shuffle phases as well as for buffer spills. In the future, the storage of temporary data might be in HDFS, but that work will not be discussed in this guide.

Speed of access, to both the HDFS data and the temporary data, is important for the efficiency of a Hadoop application. This means that higher-bandwidth disk access for the HDFS and the temporary data is needed. The Hadoop vendors typically recommend that 100–150 megabytes per second of I/O bandwidth be made available to each core that is used. High-performance applications with greater levels of bandwidth requirements might require the use of strategies other than SAN shared-storage technology for these types of data. The best storage strategy can be decided on by dividing the total I/O bandwidth available for the storage device by the number of consuming servers and cores and then comparing that figure to the application's needs. This is based on the 100–150 megabytes per second per virtual core recommendation cited previously. This I/O bandwidth measurement applies to all storage models.

A storage device's I/O bandwidth characteristics are vendor and model specific, so they will not be discussed in detail here. For those clusters where I/O performance demands are not high, and who do not have a service-level agreement (SLA) statement on performance, traditional SAN-type storage can provide an adequate storage mechanism in smaller Hadoop configurations. When there is a higher-specification read/write bandwidth required, and when SLAs are in force, those Hadoop clusters are often best supported most economically by the DAS, All-Flash or SSD technologies.

Architecture 2: The NAS Architecture Model

This example deployment model uses NAS storage for HDFS data management. This type of storage comes with HDFS intelligence, such as that implemented by the Isilon NAS storage mechanism. The Isilon storage is composed of an expandable set of file servers that collectively export the HDFS client interfaces in software through their own OS. The Isilon OneFS OS has a configurable feature set that implements the RPC interfaces of the Hadoop NameNode and DataNode processes. The HDFS interfaces appear to a consumer process as if it were the original HDFS, while internally that data is handled differently in the Isilon OneFS file system. In the architecture shown in Figure 4, all of the HDFS data is stored on the Isilon NAS storage. Any server within the collection of servers composing the Isilon NAS unit can respond to requests for NameNode or DataNode functionality. Those requests are load balanced across the networks leading into the Isilon NAS bank of servers.

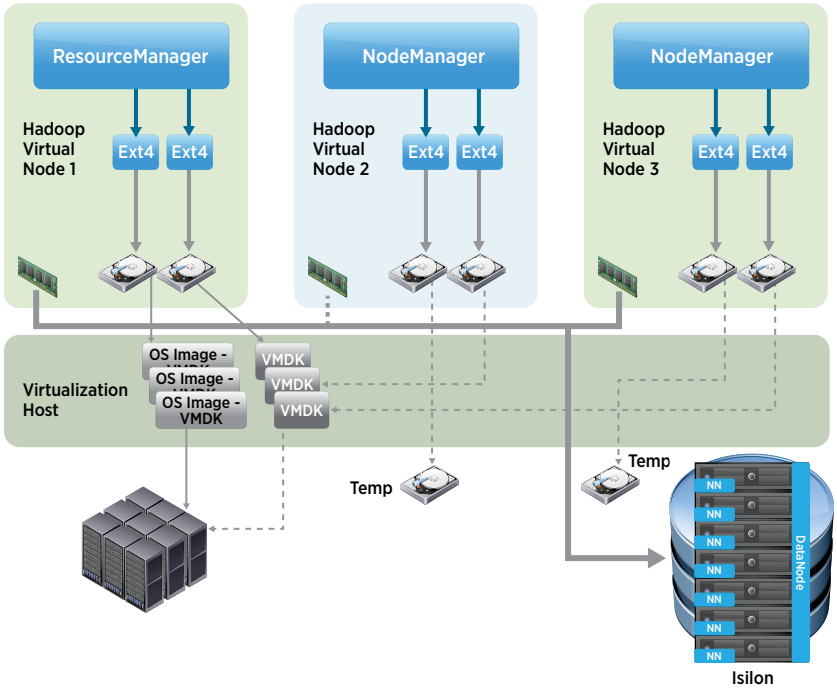

Figure 4. The NAS Model for Data and Compute Separation in Virtualizing Hadoop

Data that is outside of HDFS, such as the system disks for the VMs' guest OSs, are held in VMDK files that can be stored on the SAN shared-storage mechanisms. This is shown on the bottom left side of Figure 4. Because all the data other than the references to the Isilon HDFS storage are on the SAN, these VMs can use vSphere facilities such as vSphere vMotion if needed. The NAS-based type of architecture is described in much greater technical detail in references [10] and [20] and therefore will not be discussed in detail here.

The following are among the notable advantages that accrue from this architectural approach:

- There is lower storage consumption. This is due to employing an alternative approach to the three–replica block strategy that HDFS uses as a default on DAS.
- If the data can be placed onto the storage using NFS, FTP, HTTP, or other protocols, it need not be reingested into HDFS before it is used in Hadoop.
- Storage of the HDFS data is not spread across multiple servers dispersed around the data center. It is centralized, enabling more cost-effective management of that data.
- If a storage device were to fail, the controlling OS would provide facilities for detection and reconfiguration.
- Access control to the business data is centralized through dedicated networking and to one storage device.

The NAS architecture separates the compute from the data storage handling. Because higher-capacity networking—10GbE and greater—has become more commonplace in enterprise deployments, the design center in Hadoop that required the compute layer to be directly on the servers where the data is stored has become less important.

With a separate compute layer, the Hadoop administrator can now alter the population of worker nodes—that is, the NodeManager processes in the VMs—without affecting the data storage layer. This enables dynamic expansion and contraction of the compute power of the Hadoop cluster. The data layer benefits in the same way from this separation.

If we were to clone a new instance of the VMs described in the combined model in Figure 3, we would get a new DataNode as well as NodeManager process. Adding a set of new DataNodes to a Hadoop cluster means that a data rebalance process is required. We avoid that expensive rebalancing in this second architecture—that is, data and compute separation—by not having to include the DataNode process in the worker node VMs. These are called "compute-only workers."

Combining NAS with Other Types of Storage

In the early stages of implementing Hadoop clusters, architects want to leverage their existing virtual infrastructure, which is very commonly deployed with a SAN or NFS shared storage for the VM data. The primary goal is to minimize change. Keeping the existing host servers and using NAS storage for the HDFS data is an interesting option for this scenario. How to best handle the temporary data storage needs becomes the question.

An important concern regarding the VMs for systems that do not have DAS storage available is the storage for the "compute" Hadoop processes that create and use that temporary data: NodeManager, ApplicationMaster, and containers.

If the I/O bandwidth needs for those processes are measured and found to be less intensive, such as in prototyping in smaller clusters—that is, 10 host servers or fewer—or testing clusters, the data can be placed on SAN-based or NFS-based shared storage. Care should be taken here to monitor the consumption of the I/O bandwidth of the storage by the Hadoop cluster and applications within it. In the past, bottlenecks have been seen in this area, mainly in the connections to the SAN. The nature of the I/O in Hadoop is long, sequential reads and writes of blocks with hundreds of megabytes each. This does not match the IOPS-intensive I/O style that SANs are tuned for, when the number of servers connected to the SAN is scaled up.

Where Hadoop application performance is a primary concern for the architecture team and the I/O bandwidth demands are significant, ideally the temporary data for that cluster is best placed on DAS or on flash storage. But if no DAS or flash storage option is available, the following alternatives might be feasible:

1. If only SAN-based or NFS-based shared storage is available, it might be necessary to place the temporary data on that shared storage, potentially limiting the scalability of the system.
2. There can be a capability available to keep the temporary data on a separate zone on the NAS device, with suitable networking power added to it.

All other DataNode and NameNode application data is securely stored within HDFS on the NAS device. Traffic to and from the NAS is carried over dedicated switches and network interfaces in the VMs that map to physical switches. The switches provide a front end to the Isilon device.

A distinct advantage of this design is that it centrally stores the important business data while still providing the normal HDFS interfaces to it. This approach reduces the amount of time spent on the management of individual DAS disks on the host servers.

More-detailed instructions on constructing a Hadoop cluster with NAS storage can be found in the Hadoop Starter Kits for Isilon, which are found in references [11] and [21].

Deployment Model Summary

In this section, we have discussed two different models for deployment of Hadoop on virtual infrastructure. There are guidelines for choosing between them and for integrating ideas from different models into an individual user's design. There is no one right answer here, particularly where the applications being deployed have different storage requirements.

System Availability

After choosing a design suited to the enterprise and to the Hadoop application needs, the user then creates a plan for the computing resources that the server hosts and their VMs will use. This plan includes systems availability support. Availability is discussed in this section. The topics of CPU, memory and storage sizing, networking, and general hardware layout configurations will be treated in subsequent sections.

The architecture of Hadoop itself ensures that a Hadoop cluster is resilient in the face of many common failures, such as server or storage failure. This subject is treated separately here for the master and worker nodes, because they require different levels of protection. The Hadoop design premise is that failing worker nodes, while degrading the operation of a cluster, are not critical. Losing a master node is a much more significant event.

Availability of the Master Nodes

A Hadoop cluster functions correctly when the availability of its master nodes is ensured. Among them, the NameNode, ResourceManager, and ZooKeeper are the most critical, though they are not the only master processes. Early Hadoop releases did not provide a high-availability solution for its master nodes, but that has been rectified in the more recent Hadoop YARN releases. There are now additional, secondary processes for these master processes, running on separate VMs in the cluster. Various algorithms such as active–passive, with a warm standby for the NameNode, ensure that they are highly available from a process point of view.

The master nodes provide key services that are necessary for the correct functioning of the Hadoop distributed system. Without the NameNode process, the data stored in HDFS cannot be accessed or modified. Without the ResourceManager, resource allocation cannot be controlled and new jobs cannot be executed on the cluster. As a result, there is a need for more-extensive high-availability solutions. The additional features in vSphere can provide these solutions.

Virtualization makes availability easier for important applications, such that a VM containing one of these master processes can continue without interruption through the use of key vSphere features that handle fault tolerance or a restart on a hardware failure.

vSphere High Availability

The vSphere HA capability reduces unplanned downtime by leveraging multiple ESXi hosts configured as a vSphere HA cluster. This offers rapid recovery from server or application outages and provides a cost-effective high-availability solution for applications running in VMs.

vSphere HA protects Hadoop application availability in the following ways:

- It protects against hardware failure and network disruptions by restarting VMs on active host servers within the vSphere cluster when an original host server fails or the network connection to it fails.
- It protects against guest OS failure by continuously monitoring a VM and restarting it as required.
- It provides a mechanism to react to application failures.
- It provides the infrastructure to protect all workloads within the cluster.

After vSphere HA has been configured for a vSphere cluster of servers, no actions are required to protect new VMs. If all of the data storage of the VMs is contained on shared storage, those VMs that reside on hosts within the vSphere HA cluster are automatically protected. vSphere HA can be combined with VMware vSphere Distributed Resource Scheduler™ (vSphere DRS) to protect against failures and to provide load balancing across the hosts within a cluster.

The NameNode and ResourceManager roles can be points of temporary failure in Hadoop clusters. If the hardware, OS, or master software for these hosts fails, the entire Hadoop cluster might become degraded. This issue can be addressed by deploying these roles in a vSphere HA cluster.

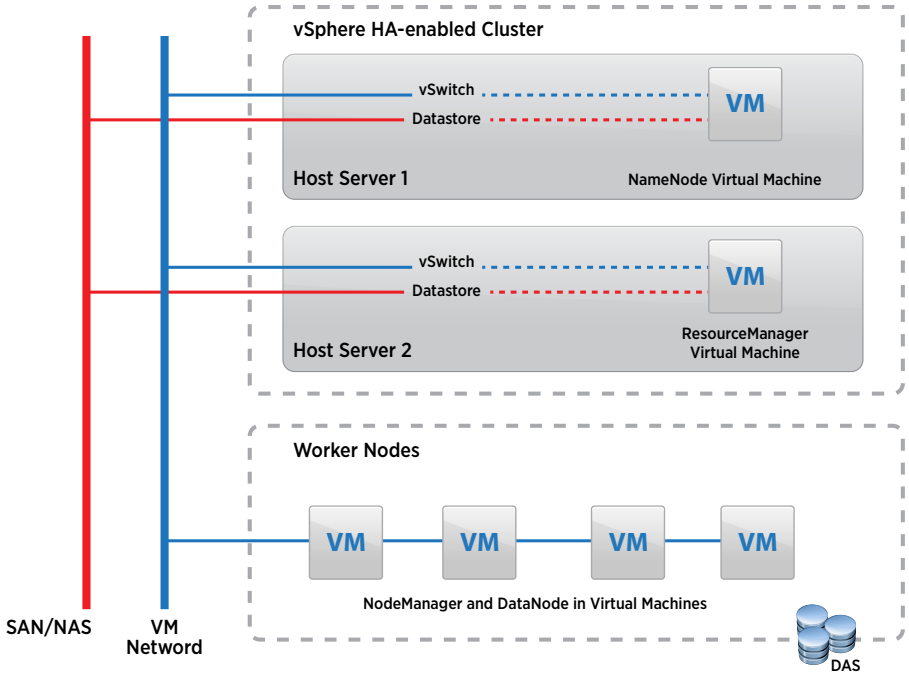

Figure 5. Virtualized Hadoop Architecture Example

Figure 5 shows a very simple example vSphere HA-enabled cluster that consists of a number of ESXi server hosts with the NameNode and ResourceManager VMs running on them. These VMs can also contain other master processes such as ZooKeeper.

If a vSphere host server that hosts a NameNode or ResourceManager process in VMs fails unexpectedly, another suitable host server within the vSphere cluster can bring up the appropriate VM to remedy the situation. This mechanism can operate in tandem with the Secondary NameNode process that is implemented in Hadoop YARN–based systems today.

To store the data for these protected VMs, a shared-storage mechanism such as SAN or vSAN is required. The virtualization administrator uses VMware vCenter® to initially configure vSphere HA functionality on the vSphere cluster. More detail on this solution can be found in reference [19].

vSphere Fault Tolerance

vSphere FT is a capability that applies on an individual VM basis. It provides an even higher level of failover than vSphere HA for those components of the system that need it. When a VM is configured in vCenter to be fault tolerant, a secondary VM operates in lockstep with it on a separate host server. If the primary vSphere FT–enabled VM fails, the secondary VM takes over from it immediately, without interruption to the execution of the processes within the guest OS. The new primary VM is then backed up by a new secondary VM, which maintains the same level of fault tolerance. vSphere automatically executes all of these functions when vSphere FT is enabled on a particular VM.

This vSphere FT functionality supports mission-critical VMs and processes, such as those found in production systems, that must not stop for even the shortest period of time. vSphere FT is also dependent on having shared storage for the VMs. The subject of fault tolerance for the protection of critical processes in Hadoop systems is explored in greater detail in reference [13].

Availability of the Worker Nodes

The majority of VMs executing on the host servers in a Hadoop cluster are the worker nodes. These often outnumber the master VMs by a large factor. We have seen deployments with five master VMs managing hundreds of worker VMs in one Hadoop cluster. The Hadoop platform software has built-in failure-recovery algorithms to detect and repair a cluster when a worker node fails. If one or more worker nodes fail, the Hadoop cluster continues to operate with a diminished capacity from a compute power perspective until that worker node is fixed and rejoined to the Hadoop cluster.

Many architects encounter the question of what happens to the data being managed by the DataNode process in a worker node when that node fails. In native HDFS implementations, the data blocks managed by a DataNode process are replicated by default to two other nodes. This is also done when the worker node is virtualized.

If the worker VM that is managing any data block becomes unavailable for any reason, the replica for that block can be found on one or more other VMs running on separate host servers in the cluster. VMware contributed source code–level additions to the Apache implementation for Hadoop to make the algorithms suitable for deploying on vSphere, along with other features. These additions are called the Hadoop Virtualization Extensions (HVE). The HVE implementation for HDFS, a subset of the full HVE, ensures that these block replication algorithms work well in a VM environment. The code contributed by VMware implements key additions to the standard Hadoop view of the physical deployment topology. This HVE technology for HDFS is available in the mainstream distributions of Hadoop that are based on the Apache original code. This is fully described in reference [12].

As mentioned earlier, Hadoop has its own built-in failure-recovery algorithms to detect and repair Hadoop cluster components if a worker node fails. Although the worker nodes do not always require vSphere HA support, there are advantages to using a uniform approach in which vSphere manages all the participating hosts, including those that have worker VMs on them. Virtualizing the entire Hadoop cluster provides operational benefits in automation, data center consolidation, higher resource utilization, systems management, and uniform monitoring and management framework for all nodes. The vSAN storage mechanism can operate across all the host servers, which enables this type of vSphere HA support for the workers as well as the master VMs.

Hardware Considerations

The hardware vendors and the Hadoop software distributors provide specific recommendations for configuring the hardware for a Hadoop cluster. There are some important points to consider when choosing hardware for virtualized Hadoop. Because big data quantities are used, and the nature of HDFS I/O is more storage bandwidth intensive, the storage mechanism chosen is a prime concern. CPU sizing and networking also play an important part.

The design and sizing options for the VMs are discussed in various architecture examples that can be found in references [1], [2], [3], and [4]. Table 1 provides some specifications of certain CPU, RAM, and disk setups that form a baseline for a server to be used in an initial proof-of-concept Hadoop cluster.

CATEGORY NAME	CONFIGURATION
CPU	Servers should have two CPUs (sockets) with at least eight cores each. Hyperthreading should be enabled. More-modern, powerful CPUs are preferred, to prevent bottlenecks in computing resources. Exact commitment is possible with virtual CPUs mapped 1:1 to logical cores. This means that the total number of virtual CPUs can equal the number of logical cores or hyperthreads on a server.
RAM	A recommendation from VMware is that 6 percent of the available physical memory is set aside for the vSphere virtualization layer at the host level beyond the total memory required by the VMs on a server. Provide at least 8GB of memory per core. Physical memory sizes of 256Gb or 512Gb are common in Hadoop deployments today. Overcommitment of memory, where the sum of configured VM memory equals or is greater than the available physical memory, is to be avoided.
Storage	The performance testing described in references [1], [2], [3], and [4] shows that JBOD, 7,200rpm SATA, and NL-SAS disks provide acceptable performance for benchmark programs such as TeraGen, TeraSort, and TeraValidate when executed across multiple VMs. The Hadoop software distributors usually recommend 1 to 1.5 physical disks per core on native systems. The same principle applies to matching numbers of disks to virtual CPUs at the vSphere layer. Generally speaking, the more disks available on the host servers, the better the performance of applications. As all-flash systems gain popularity, they will become an important type of storage of choice for Hadoop data. The guidelines for using this all-flash storage with virtualized Hadoop are provided in the vendor documents, including an Intel and VMware reference architecture.

Table 1. CPU, RAM, and Disk Configurations for Worker Nodes

The systems administrator should closely follow the guidelines provided by the Hadoop software distributor company when choosing the hardware specification for their particular setup, applying prudent judgment at the same time.

Avoid overcommitting hardware resources such as physical memory. Avoid contention for CPU power by exactly committing CPU logical cores—for example, the total number of virtual CPUs (vCPUs) in all VMs on a host server matches the number of logical cores.

For the ResourceManager and NameNode roles, consider deploying additional RAM and secondary power supplies to ensure the highest performance and reliability of these critical servers in the cluster. Given the Hadoop data distribution model, however, it is not typically necessary to deploy power redundancy on the ESXi servers that host the worker VMs.

Example Deployments of Hadoop on vSphere

This section gives three example deployment architectures for different virtualized Hadoop systems, representing small, medium, and large cluster configurations.

These examples serve as patterns that can be considered for deployment, depending on the application requirements, its size, and the organization's maturity level in Hadoop adoption.

Some organizations prefer to have a small number of larger-sized clusters; others choose a greater number of smaller clusters. This decision depends on the business requirements of the organization. There are certainly other viable architectures that are not covered here. A number of these are described in references [16], [17], and [18].

1. A Small Hadoop Configuration

This Hadoop configuration is used in the VMware IT department for analysis of customer reports of problems, buying behavior, and other patterns that are important to the business. It is an example of a starting point for those involved in proof-of-concept testing with Hadoop on vSphere or in early production deployments. The architecture, shown in Figure 6, uses four vSphere server host servers, that are connected to a 10GbE switch within the same rack. This shows that an organization can start on a small scale and work upward from there. This configuration was not tuned for high-performance results, and the jobs executed on it performed within user expectations.

The "Hadoop Master 1" VM shown in this design hosts the active ResourceManager process along with a Hive Server and Hive Metadata process. The "Hadoop Master 2" VM contains the standby ResourceManager, along with the Application History Server and Application Timeline Server. All five "Hadoop Worker" VMs contain the NodeManager process, along with other server processes that support the Pivotal SQL query engine.

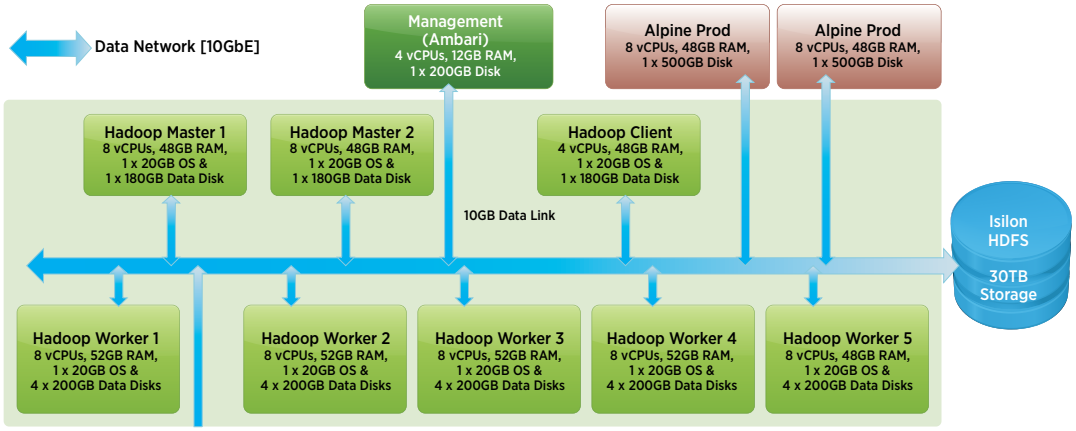

Figure 6. Example of a Virtualized Hadoop Topology for a Small Deployment

Figure 6 also shows the third type of VM. The "Hadoop Client" VM holds the Hive, HDFS, Pig, and Hadoop YARN client processes as well as a ZooKeeper process. The last is a master process, but it is held here inside a client VM because there was capacity for it. All the HDFS data that is used by these Hadoop components is stored on the Isilon HDFS device shown on the right side of the diagram. For the temporary or spill data, the design used a set of LUNs that were mapped from a separate shared storage device, a NetApp storage product. Because this cluster was run on a small number of servers, this shared storage was deemed to be adequate for the application needs.

There is a mixture of different application workloads running on this system. An example of one such application is given in the "Alpine" VMs on the top right of the diagram. Hosting different workloads on one cluster is very common in virtualized Hadoop environments. Other tools are also being placed on this cluster, and the set of use cases for it is growing over time. The architecture for both hardware and software will be expanded to meet those greater needs.

The test programs that are normally shipped with the Hadoop distributions, such as TeraSort and others, were used in earlier stages of deployment to establish confidence in the general operation of the system. The architecture in Figure 6 is used for internal IT analysis on clickstream data, support logs data, and country access data. See reference [14] for a more detailed description of this small reference architecture.

In the following section, we discuss an example of a medium-scale reference configuration with more resources, optimized for high performance.

For early, lower-scale work on Hadoop, where there are fewer than 25 nodes (VMs) in the Hadoop cluster, spread across 6 to 12 vSphere host servers, all vSphere host servers in the cluster frequently are contained within one physical rack in the data center. This means that all network traffic is determined by the networking configuration within that one rack.

In contrast, Hadoop clusters in larger systems use an architecture composed of several racks of host servers. Each server is connected to the top-of-rack (ToR) switch, and the rack of servers is connected to other racks via a spine, aggregation, or core switch.

The data-transfer speed at which these switches operate can be a deciding factor in overall system performance. When there is sizable cross-node traffic for replication of data, interrack traffic can become a limiting factor for system performance. The ToR switches can be replicated for redundancy to provide adequate bandwidth for data rebalancing when a ToR switch fails.

2. A Medium-Sized Hadoop Configuration

While carrying out a significant, long-lived performance testing exercise on various configurations and numbers of VMs containing Hadoop on vSphere, VMware engineers used a single rack loaded with 32 host servers. The architecture used two 10GbE network switches for connections between the servers, as shown in Figure 7. This setup was used to minimize or eliminate any network-based bottlenecks when looking at system performance. An architecture outline is provided here as a useful reference implementation for performance testing of an end-user's Hadoop-based application.

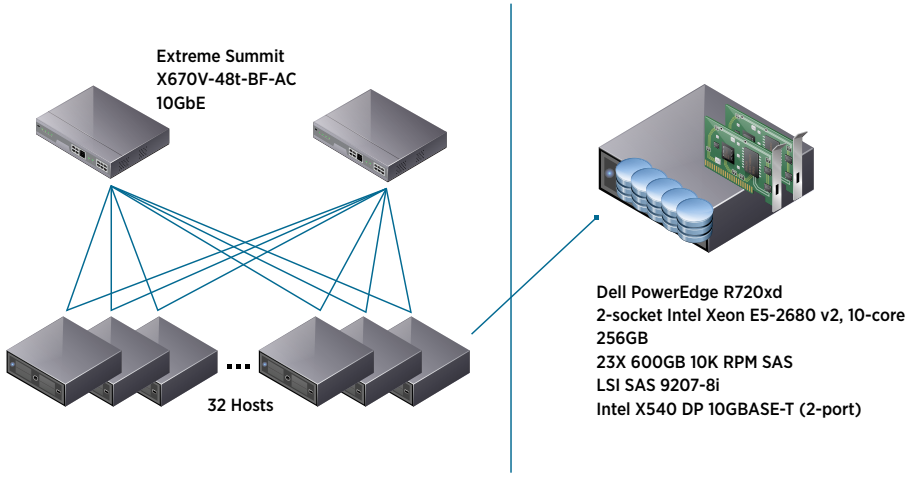

Figure 7. Example of a Medium-Sized Hadoop Cluster Hardware Setup

Each host server had two 10GbE ports on its network interfaces, and those ports were connected to separate 10GbE switches. This hardware setup, shown in Figure 7, along with the benchmark programs used and the full test results, is described in technical detail in reference [1]. This testing used the first—that is, the original Hadoop—deployment model described earlier, combining compute and data storage handling onto the same worker VMs. With two, four, or more suitably sized VMs per host server, this layout can be very effective for achieving high performance, comparable to that of native Hadoop implementations on the same hardware. Organizations that are constructing a medium-sized Hadoop cluster of up to approximately 128 VMs—that is, 128 nodes in the Hadoop cluster—can gain much insight from this work. Currently, most enterprises fit into this category. They often use fewer nodes and host servers in their Hadoop clusters for their data analysis work.

Next, we turn to a larger architecture composed of multiple racks containing many more host servers and VMs.

3. A Large, Multirack Hadoop Configuration

The larger architecture shown in Figure 8 grew within a few months from a medium-sized cluster deployed initially to quickly occupy more than 320 VMs running on more than 160 host servers. It was first deployed on eight data center racks. During its lifetime, the architecture expanded to even more racks and covered more than 220 host servers, with twice that number of VMs, in two or more Hadoop clusters.

This virtualized Hadoop cluster was originally provisioned across eight racks, each with 20 host servers. It was decided early in the process to run two VMs per host server in all cases. This principle applied whether the host servers were dedicated to hosting VMs with master processes or worker processes. The idea behind it was that if a host server were to fail, then not many VMs would fail with it. The host servers for the master VMs are shown in the orange-colored physical host servers in Figure 8. The host servers for the pairs of worker VMs are shown in green. The architecture team decided to construct separate vSphere clusters, each with 32 host servers, as horizontal stripes across the eight racks. These are indicated by the blue titles in Figure 8. This might also have been designed so that the vSphere clusters spanned one vertical rack each. The maximum number of host servers in a vSphere cluster has since that time also increased to 64. The vSphere cluster is a convenient mechanism for adding groups of host servers to an existing Hadoop system. The Hadoop cluster size is independent of that of the vSphere cluster.

The networking is organized in a leaf-and-spine approach in which each leaf switch is capable of 10GbE speed and the spine switches are capable of 40GbE speed. Each host server has 16 DAS disks that are split evenly between the two VMs on the host server. The VMs are configured to have 120GB of memory and 16 vCPUs each. These sizes enable the vSphere hypervisor to maintain each VM within a physical socket and within its nonuniform memory access (NUMA) memory node boundary, providing optimal fit and performance.

Aside from hosting important business applications, an objective of the early testing done here was to gauge the effects of multiple concurrent workloads executing in parallel on different sets of VMs on the same host servers. Applications that depend on both core Hadoop and its ecosystem components were used in this work. The enterprise tests proved, among other things, that multiple Hadoop clusters can satisfactorily coexist on the same set of hardware, without interfering with each other. This technical architecture is documented in greater detail in reference [5].

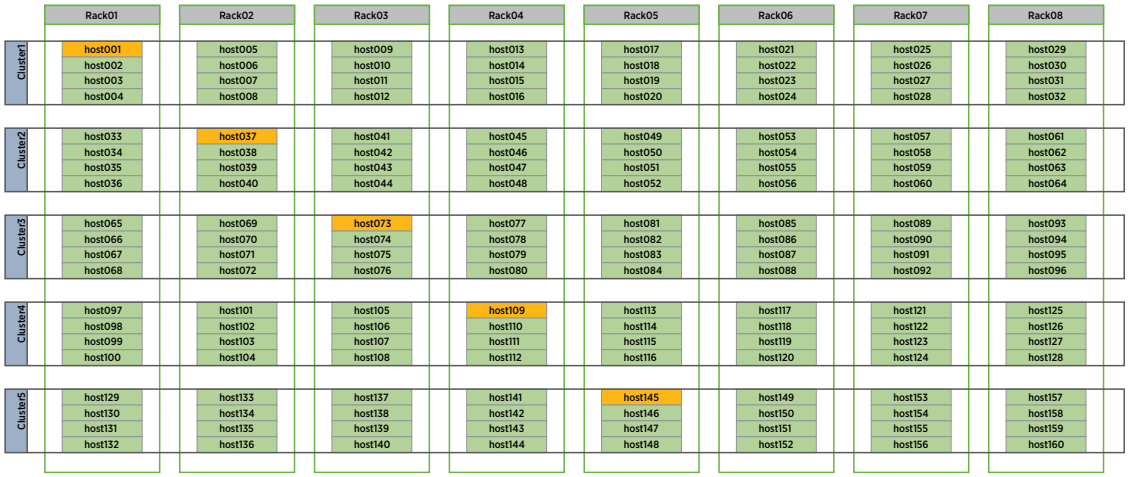

Figure 8. System Layout for a Large Hadoop Cluster for Hosting Multiple Concurrent Applications

This example large architecture demonstrates that higher-scale Hadoop clusters can be built on VMs on vSphere and can operate at runtime under load with equal or better performance than their native equivalent. It also proved that a Hadoop cluster can share the hardware with another, adjacent Hadoop cluster from a different vendor and conforming to different version and service requirements. Each cluster was dynamically scaled in and out over time by varying its number of worker VMs. This would have been difficult to achieve with the same speed if these clusters had been deployed on native systems.

Best Practices

This section introduces a number of best practices for setting up each of the main compute resources—disk storage, I/O, CPUs, and memory—when preparing them to run virtualized Hadoop-based workloads. This section should be read as an introduction to these best-practice areas. Further details on them are provided in references [1] and [4].

As a general recommendation, place a virtualized Hadoop cluster on the most up-to-date hardware that is available. This offers optimal specifications for CPU clock speed, core count, hyperthreading, physical memory, and cache sizes.

The Hadoop ecosystem—composed of application infrastructure, programming frameworks, and tools—is continually changing and improving. As this happens, more of these best practices will be revealed via our experiences as the newer features gain mainstream acceptance. For example, Spark is rapidly overtaking MapReduce for programming with Hadoop. Detailed best practices that should be consulted and followed are also presented in references [1], [2] and [4] and in other sources available from the Hadoop distribution vendors.

Disk I/O

Although various types of storage are usable with Hadoop, this section applies mainly to DAS-type storage, which is most commonly recommended by the Hadoop distribution vendors. For the other types of storage mechanisms, consult vendor documentation for more details.

1. Number of disk devices
   To set up the system for optimal performance, the more DAS disks available on the servers that support the Hadoop worker nodes, the better. Because the NodeManager and DataNode processes depend heavily on disk I/O bandwidth—100MB to 150MB per second—having more disks available decreases the likelihood that requests will execute in parallel on a single disk device and thereby cause an I/O bottleneck.

   There are examples of vSphere servers with eight local disks, but it is common to see servers in use now with two or three times that number of local disks. The Hadoop distribution vendor might recommend using 32 local DAS disks for larger servers. This is the “power user” end of the spectrum. Using DAS with large numbers of commodity disk devices has some disadvantages. In particular, the disk management task for IT staff in the case of failures is greater with higher numbers of DAS disks. There are more opportunities for a disk to fail and to require recovery processes to be applied.

2. Disks per core
   Use 1 to 1.5 disks per core or vCPU when possible. This recommendation is consistent with that of the Hadoop distribution vendors. For the architect and systems manager, there is a trade-off here between the previously described performance benefits and the cost of managing the independent disks.

3. Bandwidth
   Provide at least 100MB per second of disk bandwidth per core or per vCPU. For SATA disks, that equates to roughly one disk per vCPU.

4. Mechanisms
   For power-user applications, SAS or SATA-type disks at 7,200rpm have performed well in various testing scenarios.

5. Striping and RAID
   RAID is not required or recommended on disks that contain the HDFS data, because the HDFS system replicates its data blocks a number of times regardless of any underlying protection mechanisms at the disk level. If RAID controllers are present in the storage subsystem being used, the disks should be configured to use pass-through mode if available. If pass-through functionality is not available for the RAID controller, a single LUN or virtual disk should be created in RAID 0 format on each physical disk.

   NOTE: The term virtual disk here is a vendor-specific concept and does not refer to a VMDK.

   In keeping with the recommendations from the Hadoop software distributors, provided there is flexibility of choice, the LVM, RAID, and IDE technologies should not be used in the I/O-intensive areas of virtualized Hadoop systems.

6. Disk controller
   Ensure that the local disk controller has enough bandwidth to support all the local disks.

Sizing the Data Space

A Hadoop cluster deployment often starts small and grows as the load on it increases with more data and more deployed applications. Virtualization provides support for upscaling a deployed cluster through cloning new VMs from existing templates or downscaling a cluster through powering down a subset of the VMs.

The storage capacity to deploy is an important initial determination to make. This requires the user to predict data growth over time and to consider temporary data and replicated data for the Hadoop application's algorithms. The required data storage space can be three or more times the input data size when it is first loaded into the cluster's file system, depending on the replication factor used in HDFS particularly.

A storage-sizing exercise example follows:

1. Each HDFS data block has three replicas—the original plus two copies—by default. This must be taken into account when sizing the data space for DataNodes in particular.
2. This HDFS data block replication factor can be configured differently for different Hadoop applications. The deployment engineers for Hadoop clusters must consult with the application architects to understand the replication plan they have chosen.

The following is a simplified example of a disk-sizing exercise:

1. Assume that the data volume grows by approximately 3TB per week.
2. By default, HDFS is set up to contain three replicas of each data block.
3. Therefore, 9TB of extra storage space is required per week.
4. Add a value of 100 percent of the input data size for the temporary space. This results in 12TB per week.
5. Assuming server machines with 12 x 3TB disk drives, a new server machine is required every 3 weeks.

Creating Aligned Partitions on the Disk Devices

To store VMDK files for VMs, vSphere uses datastores, which are logical containers that hide the specifics of physical storage and provide a uniform model for storage. Datastores that represent storage devices use the VMware® Virtual Machine File System (VMFS) format, a high-performance file system format that is optimized for storing VMs.

Disk–datastore alignment is very important in achieving I/O operational efficiency in the virtualization environment. Alignment is required at two levels: at the vSphere host server level and within the guest OS of each VM.

- Alignment at the vSphere server host level
  - Use the vSphere Web user interface (UI) to achieve this alignment when creating a datastore on a disk. After datastores have been created in this way, vSphere automatically aligns partitions on the disks that it controls.
  - There might be situations where alignment must be done over many disks and where using the vSphere Web UI would involve much repetitive and error-prone work. For these situations, custom scripts are developed and used by operations personnel to execute the alignment on a larger scale.
  - See the arrangement of server datastores by using the following menu combination within the vSphere Web UI: *Hosts and Clusters > Configuration > Storage > Datastores*
- Guest OS disk alignment
  - On the Linux guest OS variants, use the `fdisk –lu` command to verify that partitions are aligned.

Provisioning Disks for a Hadoop Cluster

The vSphere VMDKs as stored on datastores can be created in one of several forms. We leave aside the raw device mapping (RDM) form for this discussion and will concentrate on creating VMFS disks in one of the three following main forms:

Zeroed Thick

All the space is allocated on the datastore at the time of virtual disk creation. It is not pre-zeroed, so for this reason it is somewhat quicker to create. As the OS in a VM writes to the disk, the space is zeroed as the I/O is committed. Zeroing the disk ensures that no old data from the underlying storage is found on the new disk.

Eager-Zeroed Thick

Using this method, as before, all the space is preallocated to the virtual disk on the datastore when the disk is first created. However, with eager-zeroed thick disks, the entire space is zeroed out at this time. These disks can take considerable time to create. But when they're ready, they exhibit a marked performance improvement over new zeroed thick disks. The performance of zeroed thick disks eventually catches up to that of eager-zeroed thick disks, typically within a few hours.

Thin Provisioned

Similar to thin provisioning on a storage array, VMDK thin disks are allocated space only as they grow from disk I/O activity. The disk starts small and expands as the space is zeroed, ready for disk I/O. It will not grow beyond its allowed size. Despite speculation to the contrary, thin provisioning does not impact performance, which is extremely close to that of zeroed thick disks. The main advantage of thin disks is the space saved by not allocating everything in advance. However, some guest disk operations, such as defragmentation, cause thin disks to inflate.

Choosing the Disk Format to Use

To avoid incurring the longer provisioning and creation times associated with the eager-zeroed thick form of disk, some systems administrators might use the zeroed thick option in vSphere as the standard method of laying out all the data disks when creating them.

When compared with zeroed thick, the eager-zeroed thick option takes a significantly longer time to provision disks because of the preallocation work that vSphere must do. However, this preallocation work gives better I/O write performance at runtime. Therefore, we recommend that users either a) provision using the eager-zeroed thick approach or b) zero-fill any disks that were created as (lazy) zeroed-thick disks, thereby changing them into eager-zeroed thick disks.

This is done so that all write operations can benefit from the disk preallocation work, thereby improving performance of the Hadoop cluster. This zero-filling process is called "warming up the disks." Based on the results of internal VMware experiments done on a test cluster, it is estimated that this warming-up process results in a performance gain.

There are a variety of techniques for warming up the disks, from executing custom scripts that zero-fill the disk to running a disk-writing I/O-intensive application, such as the TeraGen sample program, on them. The system administrator can choose which method to use for this.

All-Flash Storage

At the time of this writing, a transition is in progress in the industry from the use of magnetic disks for storage to *all-flash* technology. The flash technology is undergoing extensive testing in VMware R&D labs, in cooperation with vendors such as Intel, to prove its usefulness as a datastore for Hadoop environments. These tests use vSAN as the controlling software for the sharing of storage devices across the servers. The results of these tests show that the combination of all-flash storage devices with vSAN supports Hadoop clusters running on vSphere very well. Where an enterprise wants to use vSAN as an underlying technology with Hadoop, the recommendation from VMware is to utilize all-flash storage for that deployment. A document providing the learnings from this testing will be made available for those implementing this solution.

Virtual CPUs

1. At least two vCPUs are recommended for any VM that is executing a significant Java process, such as the main Hadoop processes. See reference [23] for more details.
2. The physical CPUs on the vSphere host should not be overcommitted. One viable approach here is that the total number of vCPUs configured across all VMs on a host server is equal to the physical core count on that server. This more conservative approach ensures that no vCPU is waiting for a physical CPU to be available before it can execute. If that type of waiting were to occur, the administrator would see a sustained increase in %Ready time as measured by the vSphere performance tools.

   When hyperthreading is enabled at the BIOS level, as is recommended, the total number of vCPUs in all VMs on a host server can be set up to be equal to twice the number of physical cores—that is, equal to the number of logical cores on the server. This "exactly committed" approach is used in demanding situations where the best performance is a requirement. Both the conservative method and the match-to-logical-core method are viable approaches, with the latter being seen as the more aggressive of the two in achieving performance results.
3. VMs whose vCPU count fits within the number of cores in a CPU socket, and that exclusively use the associated NUMA memory for that socket, have been shown to perform better than larger VMs that span multiple sockets. The recommendation is to limit the vCPUs in any VM to a number that is less than or equal to the number of cores in a CPU socket on the target hardware. This prevents the VM from being spread across multiple CPU sockets and can help it perform more efficiently. See the related discussion of NUMA in the "Memory" section that follows.

Memory

1. The sum of all the memory size configured in the VMs on a server should not exceed the size of physical memory on the host server.
2. Avoid exhausting the memory of the guest OS within the VM itself. Each VM has a configured memory size that limits its addressable memory space. When a set of memory-hungry processes, such as the Hadoop processes, is executing in the guest OS, ensure that there is enough guest OS memory space configured to enable those processes to execute without incurring swapping at the guest OS level.

3. To increase speed and efficiency, NUMA divides the host server's memory into parts that are closely associated with individual processors. Each of these parts is a NUMA node and has a particular size on various architectures. By designing a VM's memory size to fit within the boundary of a NUMA node, the resident Hadoop workload's performance should be optimal. This VM layout avoids cross–NUMA node migrations or accesses. VMs with a memory space setup that is large enough to span more than one NUMA node can incur performance impacts.
4. In the physical memory of a vSphere host server, allow for the memory requirements of the vSphere hypervisor. A general guideline is to set aside 6 percent of physical memory for the hypervisor's own use.
5. Use a memory-to-core ratio of at least 4GB of memory to one core or vCPU.

Networking

1. Use dedicated network switches for the Hadoop cluster if possible and ensure that all physical servers are connected to a ToR switch.
2. Use a network that has bandwidth of at least 10Gb per second to connect servers running virtualized Hadoop workloads.
3. Provide between 200Mb per second and 600Mb per second of aggregate network bandwidth per core, depending on the network requirements of the system.

Example: 100-node Hadoop cluster

• 100 x 16 cores = 1,600 cores

• 1,600 cores x 50Mb per second = 80Gb per second

• 1,600 x 200Mb = 320Gb of network traffic

• 1,600 x 600Mb per second = 960Gb per second of network traffic

Contemporary host servers have two or more network ports. In many cases, two ports are not enough for ESXi hosts supporting Hadoop workloads. In the example performance configuration given in reference [1], each of the 32 host servers has two network ports, with each port connected to a separate switch for optimizing network usage.

When configuring ESXi host networking, consider the traffic and loading requirements of the following consumers:

• The management network

• VM port groups

• IP storage (NFS, iSCSI, FCoE)

• vSphere vMotion

• Fault tolerance

The VMs within the Hadoop cluster communicate frequently with each other. They exchange copies of replicated data blocks in HDFS, for example. They also issue heartbeat messages to their master nodes. The DataNodes send heartbeat messages regularly to NameNode processes in remote VMs.

These VMs can also swap large quantities of data between the Hadoop processes running in different VMs. When designing this portion of the infrastructure, the network supporting the traffic between VMs should be separated from those in the preceding bullet points. In performance tests described in reference [1], two network adapters were bonded on the servers to give better network throughput.

There should be no dependency on a single network adapter or on a connection to a single physical switch. Redundant pathways for VM-to-VM traffic should be planned into the design.

Scalability is another consideration. As the environment grows larger, it becomes more complex and dynamic. In addition, managing the network configuration and keeping it consistent across all the hosts in a cluster become more difficult. Here, the VMware vSphere Distributed Switch™ (VDS) and the VMware NSX® infrastructure prove useful for managing larger Hadoop environments.

The vSphere Virtual Switch

A vSphere virtual switch is a software component in the vSphere virtualization platform that, among other functions, logically connects two VMs. It might or might not be associated with a physical switch. When two VMs are located on the same vSphere host server and are connected to the same virtual switch, network traffic between them traverses the virtual switch in memory. Such traffic does not need to be carried on a physical switch or to be handled by the physical network adapter on the host. This feature enhances the performance of network traffic between those VMs. There are two types of virtual switch in vSphere: the standard switch (vSwitch) that resides on one host server and the VDS that provides a unified management interface that spans host servers.

The vSphere Distributed Switch

The VDS technology provides ease of management for switch configuration by treating the network as an aggregated resource. Individual, host-level virtual switches are abstracted into one large VDS that spans multiple hosts at the data center level. Port groups become distributed virtual port groups (dvport groups) that span multiple hosts and ensure necessary configuration consistency for VMs and virtual ports. These can be used for such functions as vSphere vMotion migration and network storage. A VDS is suitable for all types of networking that have been discussed in this guide. More details on this aspect of vSphere are provided in the vSphere documentation.

Using VMware NSX for Virtualized Hadoop

Deploying VMware NSX products for virtual networking has a number of distinct advantages for the Hadoop architecture on vSphere. This section does not discuss all of these benefits; it focuses instead on one straightforward use case that applies in many situations in big data. This use case concerns the isolation of different Hadoop clusters and their users through segmentation at the virtual network level.

Enterprises that host applications on different Hadoop clusters for use by different groups of users must separate those clusters from each other as far as access, security, and data visibility are concerned. One example is the requirement for different regulatory compliance levels across different data sets, such as HIPAA, PCI, SOX, and others. Another example is the separation of developer and testing users from preproduction staging users in an enterprise where their underlying infrastructure is shared.

These requirements demand separate use and visibility by different departments or business units within an enterprise. The same principle applies to application and cloud providers that serve multiple, external tenant users. These architectures can be described as multitenant in nature, in the sense that each tenant community has one or more unique Hadoop clusters with data of their own that are exclusively utilized by its users. Access to the data is disallowed across those Hadoop cluster boundaries.

The VMware NSX Distributed Firewall technology enables the separation of one tenant community and its Hadoop cluster from another when they are in a shared environment. The Distributed Firewall component of VMware NSX enables policies to be implemented that prevent users of one set of VMs from access to any other set of VMs and data outside of their firewall grouping.

This approach enables users to continue using their existing VLANs and VXLANs. The isolation is achieved at the VM-to-VM level, not at the hardware level. Each isolated unit is referred to as a *segment*; the overall architecture is called microsegmentation. This isolation is available even if the separate segments have VMs on common host servers. It is also independent of the logical switch that a set of VMs might be connected to.

A separate and more advanced use of VMware NSX is to provide a flat IP address space across hundreds of physical servers and their resident VMs. This is important for larger Hadoop clusters, where management of the IP space is required to be unified though the cluster spans different physical racks, switches, and even VMware vCenter Server® boundaries.

Conclusion

This deployment guide discusses the various architecture options for deploying Hadoop workloads on VMware vSphere. Virtualization reduces the time and effort required by Hadoop architects and administrators to configure and manage one or more clusters of Hadoop nodes. Customers want to create Hadoop clusters at will and to securely isolate one group of users onto their own clusters.

The vSphere platform provides increased flexibility through abstraction of the Hadoop layer from the particular hardware servers that it runs on. It also enables increased density of processing on any one server, through combinations of VMs.

This guide examines the various architectures and establishes guidelines for choosing between them in deploying a Hadoop cluster on vSphere. Several options apply here, ranging from separation of the compute and data functions within Hadoop to storing the various data items on direct-attached storage (DAS) or other mechanisms.

This document conveys how vSphere is a viable platform on which to successfully run development, testing, and production Hadoop workloads and that combining the Hadoop and vSphere technologies into one deployment provides distinct management and performance advantages.

About the Author

Justin Murray is a senior technical marketing architect at VMware. He has worked at the company since 2007 in various roles, with a main focus on helping customers and partners use VMware products for deploying their applications on Hadoop and other platforms. To this end, he creates technical material for consumption by architects in the virtualization and application spaces and gives talks regularly on these subjects.

References

Further technical information about virtualizing Hadoop on VMware vSphere can be found in the following documents:

[1] *Virtualized Hadoop Performance with VMware vSphere 6 on High-Performance Servers*
http://www.vmware.com/resources/techresources/10452

[2] *A Benchmarking Case Study of Virtualized Hadoop Performance on VMware vSphere 5*
http://www.vmware.com/files/pdf/VMW-Hadoop-Performance-vSphere5.pdf

[3] *Virtualized Hadoop Performance with VMware vSphere 5.1*
http://www.vmware.com/resources/techresources/10360

[4] *Big Data Performance on vSphere 6 – Best Practices for Optimizing Virtualized Big Data Applications*
http://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/bigdata-perf-vsphere6.pdf

[5] *Deploying Hortonworks Data Platform (HDP) on VMware vSphere – A Technical Reference Architecture*
http://hortonworks.com/wp-content/uploads/2014/02/1514.Deploying-Hortonworks-Data-Platform-VMware-vSphere-0402161.pdf

[6] *vSphere Resource Management*
http://pubs.vmware.com/vsphere-55/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-551-resource-management-guide.pdf

[7] *Scaling the Deployment of Multiple Hadoop Workloads on a Virtualized Infrastructure*
http://www.intel.com.tr/content/dam/www/public/us/en/documents/articles/intel-dell-vmware-scaling-the-deployment-of-multiple-hadoop-workloads-on-a-virtualized-infrastructure.pdf

[8] *Toward an Elastic Elephant – Enabling Hadoop for the Cloud*
http://labs.vmware.com/vmtj/toward-an-elastic-elephant-enabling-hadoop-for-the-cloud

[9] *Cloudera Reference Architecture for VMware vSphere with Locally Attached Storage*
http://www.cloudera.com/content/cloudera/en/documentation/reference-architecture/latest/PDF/cloudera_ref_arch_vmware_local_storage.pdf

[10] *Cloudera Enterprise Reference Architecture for VMware Deployments with Isilon-based Storage*
http://www.cloudera.com/content/cloudera/en/documentation/reference-architecture/latest/PDF/cloudera_ref_arch_vmware_isilon.pdf

[11] *EMC Isilon Hadoop Starter Kit for Cloudera*
http://hsk-cdh.readthedocs.io/en/latest/

[12] *Hadoop Virtualization Extensions on VMware vSphere 5*
http://www.vmware.com/files/pdf/Hadoop-Virtualization-Extensions-on-VMware-vSphere-5.pdf

[13] *Protecting Hadoop with VMware vSphere 5 Fault Tolerance*
http://www.vmware.com/files/pdf/techpaper/VMware-vSphere-Hadoop-FT.pdf

[14] *Virtualizing Big Data at VMware IT – Starting Out at Small Scale*
http://blogs.vmware.com/vsphere/2015/11/virtualizing-big-data-at-vmware-it-starting-out-at-small-scale.html

[15] *VMware vSphere VMFS: Technical Overview and Best Practices*
http://www.vmware.com/techpapers/2010/vmware-vstorage-virtual-machine-file-system-tech-10110.html

[16] *Adobe Deploys Hadoop-as-a-Service on VMware vSphere*
http://www.vmware.com/files/pdf/products/vsphere/VMware-vSphere-Adobe-Deploys-HAAS-CS.pdf

[17] *Virtualizing Hadoop in Large-Scale Infrastructures* – EMC Technical White Paper
https://community.emc.com/docs/DOC-41473

[18] *Skyscape Cloud Services Deploys Hadoop in the Cloud on VMware vSphere*
http://www.vmware.com/files/pdf/products/vsphere/VMware-vSphere-Skyscape-Cloud-Services-Deploys-Hadoop-Cloud.pdf

[19] *Hadoop 1.0 High Availability Solution on VMware vSphere*
http://docplayer.net/5884381-Apache-hadoop-1-0-high-availability-solution-on-vmware-vsphere-tm.html

[20] *Big Data with Cisco UCS and EMC Isilon: Building a 60 Node Hadoop Cluster* (using Cloudera)
http://www.cisco.com/c/dam/en/us/td/docs/unified_computing/ucs/UCS_CVDs/Cisco_UCS_and_EMC_Isilon-with-Cloudera_CDH5.pdf

[21] *EMC Isilon Hadoop Starter Kit for Hortonworks HDP*
http://hsk-hwx.readthedocs.io/en/latest/

[22] Trujillo, G., et al. *Virtualizing Hadoop.* VMware Press, 2015.
http://www.pearsonitcertification.com/store/virtualizing-hadoop-how-to-install-deploy-and-optimize-9780133811025

[23] *Enterprise Java Applications on VMware – Best Practices Guide*
http://www.vmware.com/files/pdf/techpaper/Enterprise-Java_Applications-on-VMware-Best-Practices-Guide.pdf

[24] Apache Hadoop Web Site
http://hadoop.apache.org/

[25] Sammer, E. *Hadoop Operations.* O'Reilly Media, 2012.

[26] *Big Data in the Enterprise – Network Design Considerations*
http://www.cisco.com/en/US/prod/collateral/switches/ps9441/ps9670/white_paper_c11-690561.html

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com