Hortonworks

Community Driven Apache Hadoop

Apache Hadoop
Cluster Configuration Guide

April 2013

Introduction

Sizing a Hadoop cluster is important, as the right resources will allow you to optimize the environment for your purpose. However, this is no simple task as optimizing a distributed environment and its related software can have its complexities.

We have packaged up years of experience building and optimizing Hadoop cluster into a single document and present to you this guide that should help guide you through this process. It is important to note that while we outline suggested sizing configurations, no two implementations are alike and mileage may vary. To this end, this guide provides you a basic framework to address sizing. We hope it will be helpful to you.
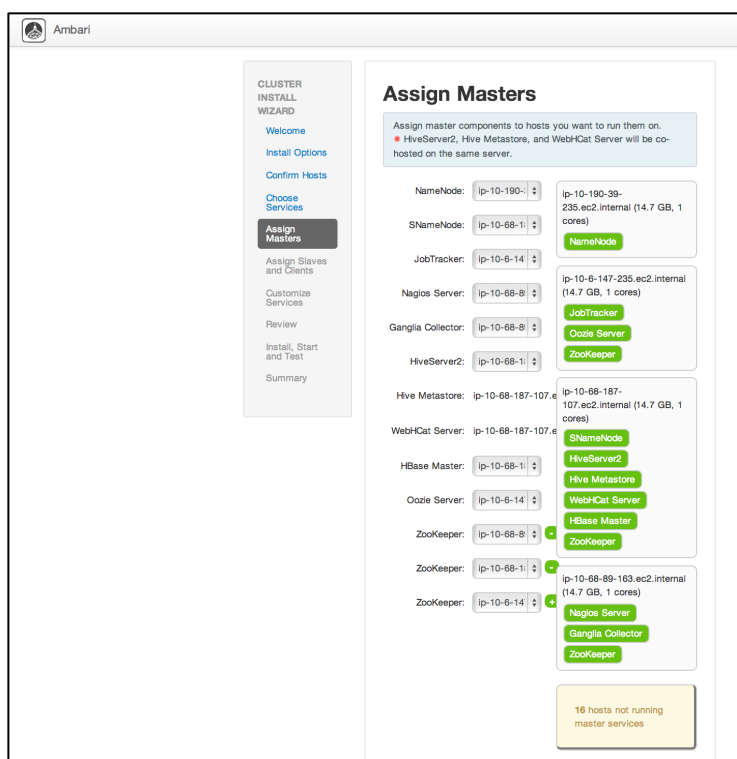
The Basics

Hadoop and HBase clusters have two types of machines:

- **masters** (the HDFS NameNode, the MapReduce JobTracker, and the HBase Master)
- and **slaves** (the HDFS DataNodes, the MapReduce TaskTrackers, and the HBase RegionServers).

The DataNodes, TaskTrackers, and HBase RegionServers are co-located or co-deployed for optimal data locality. In addition, HBase requires the use of a separate component (ZooKeeper) to manage the HBase cluster.

Hortonworks recommends separating master and slave nodes because of the following reasons:

- Task workloads on the slave nodes should be isolated from the masters.
- Slaves nodes are frequently decommissioned for maintenance.

Clusters of three or more machines typically use a dedicated NameNode/JobTracker and all the other nodes act as the slave nodes. For a medium (45-150 node) to large(300+) cluster, we recommend expanding this to 4 master nodes as follows:

1. Name Node
2. JobTracker Node
3. Secondary Name Node, HBase Master, Hive Server
    Note – in a half-rack cluster, this would be combined with the Job Tracker
4. Management Node (Ambari, Ganglia, Nagios)
    Note – in a half-rack cluster, this would be combined with the Name Node

For NameNode failover, we recommend setting up an NFS share to hold the block map that is accessible through the same mount point on all of the master nodes above.

That leaves the remaining hosts for Slave nodes, each running a DataNode, TaskTracker, and HBase Region Server. As an option, clients may be installed on these nodes to facilitate data and processing integration between the Hadoop cluster and legacy systems within the environment. This can be seen in the Assign Slaves and Clients step of the Ambari Cluster Install Wizard.
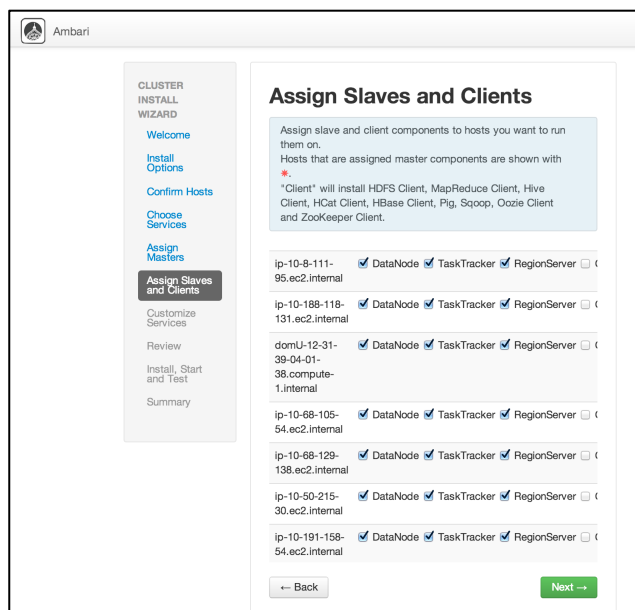
Balanced Value and Optimum Price/Performance

A Hadoop cluster is different from common enterprise IT server infrastructure. First, a workload or job within a Hadoop cluster will have little or no idle time during its life. Second, Hadoop requires a "balanced value" which guides you to select hardware configurations for master and slave nodes where consumption of CPU, memory and disk resources all peak together. The result of a balanced value design is optimum price/performance of your Hadoop cluster.

Hadoop workloads are different from common IT applications

The Hadoop workload is consumes data and creates new data sets and is typically limited by CPU, memory and disk bandwidth. If you compare this to a typical IT workload you find the Hadoop workload spends very little time in an idle state. Hence, A Hadoop workloads is typically a single execution task. Running multiple Hadoop workloads (in version 1.0) on a common set of hardware often results in saturation of the CPU, memory and/or disk I/O resources. The net is overall job load loses efficiency due to bottlenecks. We recommend running master and slave node on physical hardware for the best efficiency and price/performance optimization.

A "shared nothing" architecture promotes scalability

Another key design concept in Hadoop cluster design is "shared nothing". This concept says that a master or slave node should not share any hardware resource with any other node. This design concept highlights the performance issues that will result of using SANs for disk storage and unified I/O fabrics found in blade servers. Share resource infrastructure assumes that the average of the demand from the jobs will not exceed the total bandwidth of the resource. The performance issues come from the non-interactive job profile of Hadoop workloads. Since all Hadoop jobs are demanding max performance the shared resources will become saturated and result in a performance bottleneck.

The benefit of a "shared nothing" architecture comes when nodes are added to a cluster. Since each node is not sharing any resource with other nodes the addition of a new node results in added CPU, memory and disk I/O resources. When we couple this with the ability of a Hadoop cluster to break a larger job into multiple independent jobs the results is a near linear scalability of the cluster. If you need to double the capacity of the cluster you can easily double the number of nodes.

Balance: the key to high performance & affordable Hadoop Clusters

When building Hadoop Clusters make sure all components are balanced in capacity, cost and performance. With a balanced cluster we minimize performance bottlenecks and avoid excess unused capacity. Since there are many more slave nodes than master nodes most of our discussion will revolve around slave node design consideration.

Common Cluster Component Guidelines

If we look at a node design for master and slave nodes we see we have control over selection of CPU, memory and disk I/O. We have secondary considerations of power, cooling and physical space. We need to keep these in mind as the size of the cluster scales up it acts as a multiplier and these secondary considerations can very quickly make our design impractical to implement.

- **CPU and memory selection**

  Always start with CPU selection. Here we need to look at price/performance of our CPU choices. Picking the current top of the line CPUs will result in a very expensive cluster as you multiple out this cost by the number of nodes. Going the other direction picking underpowered CPUs results in you cluster not being able to meet its performance goals. Since there is strong innovation happening in the CPU market the sweet spot will move quarter to quarter. Right now we find the Intel Sandybridge quad core processors to be good value. We expect that the hex core processors will become more competitive and when you do your design you should check on the price/performance ratio.

- **Disk selection**

  The amount of disk on each slave node will define the amount of data your cluster can hold. The speed of the disks will be one of the bottlenecks that will constrain the slave node performance. So our disk selection is going to be a tradeoff between performance and capacity. The highest capacity disks us a SATA interface. SAS drives have a more sophisticated command set which will give you better performance and reliability but SAS drives cost more. We have found that near line SAS drives gives us a good combination of both capacity and performance. A near line SAS drive is the marriage of a SAS interface to a SATA disk mechanism. So we get the capacity of SATA with some of the performance of SAS. We do lose on reliability and life span but since Hadoop is designed to accept drive failures this is an acceptable trade off. This kind of design decision is typical of what you will do as you iterate through your cluster design.

- **Networking**

  Networking for Hadoop clusters is simple since most of the data traffic is taking place inside the slave nodes. The network is used for control, bringing in new data and moving out the results of jobs. For each rack of servers there will be a 1G rack level switch with a 10G uplink. that connects to the nodes in the rack. For small to medium clusters we

can just daisy chain these switches together. When you go over 5-7 racks you will then need a 10GB spine switches to interconnect the racks.

Our trade off decision in this case is 1 GB Ethernet vs 10 GB Ethernet. This decision will be made on cost and infrastructure considerations. While the incremental cost of 10 GB Ethernet interfaces may be reasonable the next question is can the interconnect infrastructure handle it?

Later we will talk about cluster isolation and if we have isolated the cluster this may be a good decision. From a balanced value point of view right now 1 GB Ethernet between nodes in a rack with a 10GB Ethernet backbone will meet most current designs.

- **Redundancy in cluster components**
In the design of an Apache Hadoop cluster we apply redundancy sparingly. From a cluster point of view the built in redundancy of Hadoop software allows a cluster to lose a drive, a slave node or if the scale is large enough a whole rack of servers. So as your cluster grows the loss of any component diminishes. This is part of the beauty of the Hadoop design.

The important place to apply hardware redundancy, i.e. extra power supplies, multiple network paths, etc. is with the master nodes. This goes back to the fact that the loss of a master node will affect the whole cluster. This is the place to employ enterprise grade server hardware to minimize the chance of losing a master node.

The exact opposite happens with the slave nodes. Since data is replicated through the cluster the loss of a disk drive or a slave node it managed by Apache Hadoop. In a large enough cluster replacement of failed hardware is a scheduled maintenance task and Apache Hadoop manages replacement of hardware so there is no cluster downtime. It is important to realize that slave hardware failure is not a priority one issue and it can be resolved the next time there is a hardware replacement maintenance event.

Network redundancy changes as you go from small to large clusters. In a small and medium cluster loss of a switch has a larger impact than in a large cluster. We do not recommend redundant switches but we do recommend redundant paths so you can minimize the impact of a switch failure. We mention this later on but a good design

pattern to use here is to physically split the cluster into two equal halves. So if one cluster has a failure the other cluster can still process jobs.

Master Node Design Guidelines

The master nodes are key components of your Hadoop cluster. Since there are only a few of them we can relax some of our cost constraints. We want to optimize these nodes for availability. We recommend using enterprise quality hardware since a master node failure will impact the cluster. Redundant power supplies and network connections are applicable here to help increase the availability of the node.

- **Memory**

   Memory demand for a master node is based on the NameNode data structures that grow with the storage capacity of your cluster. We found 1 GB per petabyte of storage is a good guideline for master node memory. You then need to add on your OS overhead, etc. We have found that with Intel Sandybridge processors 32GB is more than enough memory for a master node.

- **Local disk configuration**

   The local disks on a master node should be configured using RAID10 with hot spares. This configuration is picked to give good performance and fast recovery on loss of a disk. We recommend 6 x 1TB drives. Four are allocated to the Raid 10 with two hot spares**.**

- **NFS server for cluster meta-data**

   The critical element of the master nodes are the meta-data files that hold the information about the cluster. Loss of these files will render your cluster useless. A best practice is the place all these files on a small NFS server. This will allow you to quickly recover from a loss of a master node by simply starting a new copy of the master node and pointing it at the metadata files on the NFS share.

- **HA considerations**

   In order to protect the master nodes many clusters use an HA configuration to maintain the availability of the master nodes. However if we look at the rate of failure of master nodes, especially if they are built with redundant hardware this does not happen very

often. So you need to balance the complexity of HA against the possibility of failure of the master node. Also if you implement HA it needs to be tested regularly to make sure it works and you know how to transition through a failover. Overall you need to consider the trade off of complexity vs. the recovery benefits from HA.

An alternate configuration would be to split your servers into two independent clusters and replicate the data between the two clusters. In the beginning you keep the two clusters together until operations has stabilized. Now when a cluster fails you can just move jobs over to the other cluster. You are also set up very nicely to later move one cluster to a remote datacenter to achieve geographical dispersion. The penalty to this configuration is the data replication and you cannot use all your slave nodes on one big problem.

- **NameNode guideline**

  An important resource for NameNodes is memory to hold working data structures. A very rough guideline for NameNode memory is there needs to be 1GB of RAM for every petabyte of allocated storage space.

Slave node design guidelines

The design of slave nodes is dominated by the fact there are very many of them. Anything you do good or bad is multiplied by the number of nodes. As noted above you are looking to balance the CPU, memory and disk I/O configuration such that they all reach their peak limits around the same point.

- **Memory**

  If we follow the Intel Sandybridge processor selection we have found 32 GB of ram to be sufficient for a dual quad core server. Remember that memory is cheap but when you multiple the cost of the number of slave nodes the extra money spent on ram could allow you to buy a few more slave nodes.

- **Disk**

  Disk selection is limited by the I/O bandwidth of the processor chip set and the disk controller card. Based on our Intel Sandybridge recommendation we find that 6 x 1TB disks configured as JBOD will do well. A JBOD configuration keeps the disk controller out of the way and allow for full speed I/O with the processor.

Cluster Design Tradeoffs

We classify clusters as small (around 2-3 racks), medium(4-10 racks) and large(above 10 racks). What we have been covering so far are design guidelines and part of the design process is to understand how to bend the design guidelines to meet you goals. In the case of small, medium and large clusters things get progressively more stringent and sensitive when you bend the guidelines. For a small the smaller number of slave nodes allow you greater flexibility in your decisions. There are a few guidelines you don't want to violate like isolation.

When you get to a medium size cluster the number of nodes will increase your design sensitivity. You also now have enough hardware the physical plant issues of cooling and power become more important. Your interconnects also become more important. At the large scale things become really sensitive and you have to be careful because making a mistake here could result in a design that will fail. Our experience at Hortonworks has allowed us to develop expertise in this area and we strongly recommend you work with us if you want to build Internet scale clusters.

Growing Your Cluster

Once you get your initial cluster built your success will create a demand to increase your cluster size. This will be adding new slave nodes. We recommend adding to your cluster in increments of racks of nodes. This keeps things simpler since you just need to replicate racks. You will probably be expanding in your existing datacenter so try to keep the new racks near because you will need to directly connect the new racks to the cluster.

As you grow from a small to medium cluster you want to watch the bandwidth of the rack switches. At this point the racks are daisy chained together. When you exceed 5-7 racks you may need to introduce a spine switch that interconnects the rack switches. Growing the cluster one rack at a time is a good strategy because it will let you spot issues as they emerge vs have a large change in the cluster and trying to find issues. Otherwise as you grow to the 10-14 rack range the cluster should grow nicely.

When you go from medium to large clusters your physical infrastructure, space, power, cooling, etc. will be the driving functions. As server best practices dictate test as you build. Integrate and test each rack or row. Make sure you are adding a solid and stable resource to your cluster.

If you are planning a major expansion please engage your Hortonworks team as we have been through many cluster expansions. We have the knowledge and tools to make this a smooth transition.

Isolation

A Hadoop cluster is an integrated supercomputer cluster. As such issues like network interconnect latency is important. It is strongly recommended that your Hadoop cluster be implemented on an isolated network with a minimal amount of network complexity. We have installations where the Hadoop cluster servers were scattered through the datacenter and the result was network instability of the whole datacenter. You need to isolate and localize the Hadoop servers from the rest of the datacenter.

Heterogenous Slave Nodes

In your first build out of your cluster it will be homogenous and all the slave nodes will be the same or quite similar. Over time technology changes and you will by buying different node configurations.

When you have a mix of faster and slower nodes Apache Hadoop does make adjustments in the job stream. A feature of Apache Hadoop is called speculative execution. This happens when a node is behind in processing its share of a job. The job tracker will start another copy of that task on another node just in case the first one fails. It will then take the results of whichever node finishes first. So this works our fine at the job level.

At the cluster level this results in some inefficiency because you are running a task twice. Where problems might results is when there a large performance differences in nodes. Some things to watch for are things like SATA I/SATA II disks, 1GB/10GB Ethernet, difference in spindle count, ext3/xfs file systems, etc. These things make a difference in node performance and what can happen is that your cluster will develop a hot spot where a server or set of servers become a bottleneck in getting jobs done. In general you want to keep slave nodes balanced so that any node can run a task equally well.

Small Hadoop Cluster Design Example

A small Hadoop cluster could consist of a two-level architecture built with rack-mounted servers housed in one or two racks. With a high density of disks (6-12 per node) on the slave nodes, Hortonworks does recommend two racks to start in order to best support the load, power, & cooling specifications of most datacenters as well as the expansion that most customers undertake after their initial Hadoop implementations – so these constraints / plans should be reviewed before choosing a single rack. A single rack of servers should be interconnected using a 1 Gigabit Ethernet (GbE) switch. For purposes of this cluster planning document, we are assuming a single rack is 42U.

Recommended hardware specifications for a half-rack cluster are as follows:

Rack 1: Small Cluster	
Master Nodes **2 Servers**	*2 - Intel Quad core Sandybridge processors* *8 - 8GB DDR3 rDIMMs (Sparing, ECC)* *1 – Perc S100 (RHEL6/xfs – RAID10)* *6 – 1 TB, NL-SAS disks* *Built-in dual GbE* *Dual Power Supply*
Slave Node **8 x 2U servers**	*Intel Quad core Sandybridge processor* *8 - 4GB DDR3 DIMMs* *2 – Internal 2.5" SATA drives for root (RHEL6/xfs)* *6 – 2TB, SATA disks (SATA/ahci)* *Built-in dual GbE* *Dual Power Supply*
Rack-level Switch: 1U	*24 – ports GbE*

This half-rack cluster is sized for 96 TB of Raw Storage. To calculate how much usable disk space that provides we typically plan for 2 replicas for redundancy purposes with 20% headroom on each disk. This breaks down to 26.6 TB usable disk across this 8 node cluster. The calculation is as follows:

Usable Disk = Raw Storage / (3 * 1.2)
Usable Disk = 96 TB / 3.6
Usable Disk = 26.6 TB

Typical Hadoop Cluster – Multiple Racks

Hortonworks recommends starting with a multiple rack configuration for ease of expansion. Typically, a large Hadoop cluster consists of a three-level architecture built with rack-mounted servers. Each rack of servers is interconnected using a 1 Gigabit Ethernet (GbE) switch. Each rack-level switch is connected to a cluster-level switch (which is typically a larger port-density 10GbE switch). These cluster-level switches may also interconnect with other cluster-level switches or even uplink to another level of switching infrastructure.
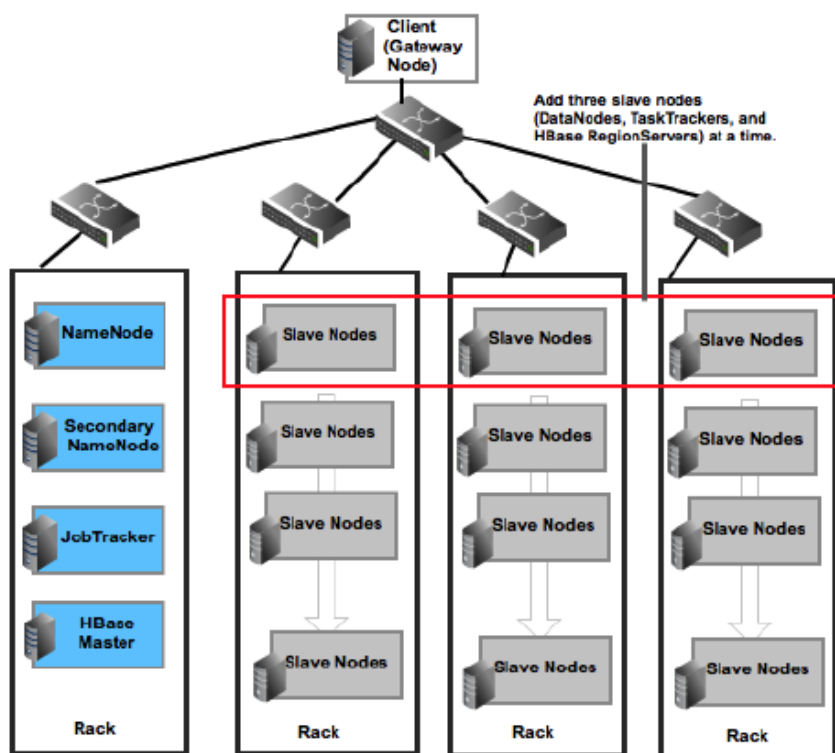
NOTE: DataNodes, TaskTrackers, and RegionServers are typically co-deployed.

The Master Nodes and Slave Nodes would typically deployed in two racks, inter-connected through a cluster-level switch (which is typically a larger port-density 10GbE switch). For purposes of this cluster planning document, we are assuming a single rack is 42U. Recommended hardware specifications for a large cluster are as follows:

Rack 1: Cluster	
Master Nodes **4 x 2U Servers**	*2 – Intel Sandybridge quad core processors* *8 – 4 GB DDR3 rDIMMs (Sparing, ECC)* *1 – Perc S100 (RHEL6/xfs – RAID10)* *6 – 1 TB, NL-SAS disks* *Built-in dual GbE* *Dual Power Supply*
Slave Node **8 x 2U servers**	*Intel Quad core Sandybridge processor* *8 - 4GB DDR3 DIMMs* *2 – Internal 2.5" SATA drives for root (RHEL6/xfs)* *6 – 2TB, SATA disks (SATA/ahci)* *Built-in dual GbE* *Dual Power Supply*
Rack-level Switch: 1U	*24 – ports GbE* *1 – 10GbE SFP (optional)*
Cluster-level Switch: 1U	*24 – ports GbE*
Rack 2-n: Cluster Data Nodes	
Data Nodes **20 x 2U Servers**	*Intel Sandybridge quad core* *12 - 4GB DDR3 DIMMs (2/channel, clocks to 1067, 3 channels/socket)* *2 – Internal 2.5" SATA drives for root (RHEL6/xfs)* *1 – 12x3.5" Drive Bay Option* *6 – 2TB, SATA disks (SATA/ahci)* *Built-in dual GbE* *Dual Power Supply*
Rack-level Switch: 1U	*24 – ports GbE per rack*

This multi-rack cluster is sized initially for 240 TB of Raw Storage. To calculate how much usable disk space that provides we typically plan for 2 replicas for redundancy purposes with 20% headroom on each disk. This breaks down to 66.7 TB usable disk across a 20-node cluster. The calculation is as follows:

Usable Disk = Raw Storage / (3 * 1.2)

Usable Disk = 240 TB / 3.6

Usable Disk = 66.7 TB

Additional capacity can be added to this multi-rack cluster by half-rack (10) or full-rack (20) increments of slave nodes. A half-rack of slave nodes would add 120 TB of additional Raw Storage (33.4 TB of additional Usable Disk). A full-rack of slave nodes would add 240 TB of additional Raw Storage (66.7 TB of additional Usable Disk).

Conclusion

The greatest trait of Apache Hadoop cluster design is the larger it gets the more resilient it gets. When you have a large numbers of nodes the loss of a disk, server or even rack diminishes. When you make your component selections we have found more nodes are better than more powerful nodes. Also the less you pay for each node the more nodes you can buy. This directly translates into how much work you can process with your cluster.

Key Takeway: A recipe for configuring a clean stable cluster

We leave you with a set of guidelines and suggestions on how to configure a clean stable infrastructure on which to install your Apache Hadoop cluster. These guidelines were developed by engineers with years of Hadoop experience and present their preferred default configuration. Following these guidelines will help you avoid 80% of the throughput bottlenecks in a cluster. If this is your first cluster you should start here, get the cluster stabilized and then start modifying it for what you are doing. This will get you a known stable baseline system.

- RedHat Enterprise Linux 6.3 – there are improvements in this release for memory page cache BDI flusher performance Hortonworks Data Platform 1.2 or higher – picks up fadvise/drop-behind and the 4MB readahead for file system performance
- Use XFS file system on all HDFS LUNs – as of 3/10/13 there is a write bug in ext4, ext4 read performance is fine
- For 2TB disks – create a 250GB temp and a 1.75 HDFS partition, this will isolate HDFS temp directory churn, on a 12 drive node this gives you 3TB of temp which is sufficient
- Use XFS on the temp directory partitions also

- Do not create /hdfs or HDFS temp directory on the root volume
- Slots: For Intel processors a dual processor hex core system will show up as a 24 CPU system. However half of the "CPUs" are due to the Intel Hyper Threading feature. To account for this, set the Apache Hadoop map slots to 2/3 of the number of cores.

As noted above these suggestions are all implemented before you install Apache Hadoop. The next step after this is to implement our Hortonworks Data Platform pre-installation checklist.

Apache Hadoop Cluster design is a serious platform-engineering project and design decisions need to be well understood. What you can get away with in a small cluster may cause issues as the cluster grows. This guide has given you the basics of Hadoop Cluster design and you can easily design a small cluster. We do strongly recommend you engage Hortonworks if you need to build a large Internet scale cluster. Our proven experience, learned from Yahoo, allows us to help you avoid common design issues and deliver a stable working Hadoop cluster.