WHITE PAPER

August 2013

Deploying Apache™ Hadoop® with Dell and Mellanox VPI Solutions

In collaboration with Dell

Background

Storing and analyzing rapidly growing amounts of data via traditional tools introduces new levels of challenges to businesses, government and academic research organizations.

Hadoop framework is a popular tool for analyzing large structured and unstructured data sets. Using Java based tools to process data, a data-scientist can infer users' churn pattern in retail banking, better recommend a new service to users of social media, optimize production lines based on sensor data and detect a security breach in computer networks. Hadoop is supported by the Apache Software Foundation.

Hadoop workloads vary based on target implementation and even within the same implementation. Designing networks to sustain the different variety of workloads introduces challenges to legacy network designs in terms of bandwidth and latency requirements. Moving a terabyte of information can take several minutes using a 1 Giga-bit network. Minutes long operations are not acceptable in an on-line user experience, fraud detection and risk management tools. A better solution is required.

Mellanox Solutions for Apache Hadoop

Building a Hadoop cluster requires taking into consideration many factors such as, disk capacity, CPU utilization, memory usage and networking capabilities.

Using legacy networks creates bottlenecks in the data flow. State-of-the-art CPUs can drive over 50 Giga-bits-per-second while disk controllers capable of driving 12 Giga-bits-per-second are entering the market, and the result is more data trying to flow out of the compute node.

Using 40Gb Ethernet and FDR InfiniBand satisfies the needed dataflow requirements for high speed SAS controllers and Solid State Drives (SSDs) 10Gb Ethernet is becoming the entry level requirement to handle dataflow requirements of common spindle disk drives.

Scaling and capacity planning should be another point of consideration. While businesses grow linearly, their data grows in an exponential form at the same time. Adding more servers and storage should not require a complete re-do of the network, using edge switches and easy to balance, flat, network is a

necessity. Mellanox's InfiniBand and Ethernet switches provide the best cost/performance ratio for scale-out systems, and creating a balanced network with 40Gb Ethernet and FDR InfiniBand is a straightforward procedure.

Usage of RDMA capable network interface cards delivers the needed CPU offload and low latency connectivity for the Hadoop framework. In the next section we review the RDMA acceleration features and benefits.

Ethernet Performance Acceleration RDMA over Converged Ethernet

ConnectX-3 utilizing IBTA RoCE technology provides efficient RDMA services, delivering low-latency and high-performance to bandwidth and latency sensitive applications. With link-level interoperability in existing Ethernet infrastructure, Network Administrators can leverage existing data center fabric management solutions.

Sockets Acceleration

Applications utilizing TCP/UDP/IP transport can achieve industry-leading throughput over 10/40/56GbE. The hardware-based stateless offload and flow steering engines in ConnectX-3 reduce the CPU overhead of IP packet transport, freeing more processor cycles to work on the application. Sockets acceleration software further increases performance for latency sensitive applications.

Mellanox Unstructured Data Accelerator (UDA)

Mellanox's Unstructured Data Accelerator (UDA) is a user transparent software plug-in solution to the Hadoop MapReduce framework. UDA accelerates the intermediate data transfer between Mappers and Reducers.

UDA is a novel data moving protocol which uses RDMA in combination with an efficient merge-sort algorithm, to accelerate Hadoop clusters based on Mellanox InfiniBand and 10/40Gb Ethernet RoCE (RDMA over Converged Ethernet) adapter cards, to efficiently move data between data nodes in the Hadoop framework.

UDA Performance

UDA is based on the network-levitated-merge[1] algorithm. In this algorithm, the new data movement overcomes a serialization process between shuffle and merge and reduce phases. RDMA (Remote Direct Memory Access) accelerates the data transfers between mappers and reducers, as well as reducing CPU overhead by offloading the burden of data transfer. Offering better CPU availability increases the number of processes available for analytics, increasing throughput capability.

UDA parallelizes the shuffle and merge processes with the reduce phase, The Map output Files (MoF) should be available and complete on time in order to enable this parallelism. The new processing scheme implemented in UDA adds a significant performance boost to the framework by better utilizing CPU cores and reducing the re-submission of jobs due to failed merge process.

Unstructured Data Accelerator can double data analytics throughput and reduce total job execution time by up to 50 percent. Larger data sets will benefit from:

- Higher wire throughput and lower latency
- More CPU slots enable better allocation of Mapper and Reducer jobs in the framework pipeline
- fewer hard disk accesses due to memory-to-memory transaction result in faster data movement and overall faster execution time

Hardware

To implement and test the technology, you will need:

- At least one Dell R720 or Dell R720dx Master Node (NameNode, Job Tracker)
- At least three Dell R720 or Dell R720dx Slave Nodes (DataNode, Task Tracker)
- Four or more Mellanox ConnectX®-3
- Four or more cables required for the ConnectX-3 card

There are many options in terms of adapters, cables and switches. Refer to Mellanox's website, where you can find more information about Virtual Protocol Interconnect® (VPI) adapters, http://www.mellanox.com/page/infiniband_cards_overview, and Mellanox switches, http://www.mellanox.com/page/switch_

systems_overview.

In this article we will review a 5 node cluster configuration. Scaling the deployment is easily done by adding more Slave Nodes to the deployment. When scaling the deployment, take into consideration the amount of RAM you have in the Master Node, as well as the disk space.

High availability features are discussed within the above Apache Hadoop framework link. We recommend deploying two Master Nodes in master and secondary name node configuration.

Recommended Server Configuration

Node Type	Hardware Part	Specification	Comments
Master Node (NameNode, Job Tracker)	System CPUs	Two, Quad core or more	
	RAM	32GB or Higher	
	Disk Drives	Two or More, 1TB each	RAID configuration
Slave Node (DataMpde, Job Tracker)	System CPUs	Two, Quad core or more	
	RAM	24GB or Higher	
	Disk Drives	Four or more, 1TB each	JBOD configuration

Table 1. *Hadoop Server Recommended Configuration*

Use the Dell server from the below list to build a Master Node:

Model	Memory	Disk Bays	Expansion Slots
Dell R720	24 DIMMs, DDR3	16 Hot-Swap 2.5"	1x PCIe x16, 6x PCIe x8

Table 2. *Dell Hadoop Master Node Server Configuration*

Use either of the Dell servers from the below list to build a Slave Node:

Model	Memory	Disk Bays	Expansion Slots
Dell R720dx	24 DIMMs, DDR3	26 Hot-Swap 2.5"	2x PCIe x16, 4x PCIe x8
Dell R720	24 DIMMs, DDR3	16 Hot-Swap 2.5"	1x PCIe x16, 6x PCIe x8

Table 3. *Dell Hadoop Slave Node Server Configuration*

It is highly recommended to have larger RAM size on the master node to handle the cluster's metadata, and to minimize the spill to the disks during this operation.

The above configuration is recommended for most use cases. There are several cases in which higher RAM and disk space is required. For such deployments, it is recommended that you contact us at bigdata@mellanox.com, where you can engage with one of our regional system engineers to help deploy your Hadoop cluster.

Five Node using 40 GbE Interconnect

Quantity	Part Number	Description	Link
5	MCX314A-BCBT	ConnectX-3 Ethernet Dual QSFP+ Port Adapter	http://www.mellanox.com/related-docs/user_manuals/ConnectX-3_Ethernet_Single_and_Dual_QSFP+_Port_Adapter_Card_User_Manual.pdf
1	MC2210130-002	QSFP to QSFP cable, 40Gb Ethernet, 2m	http://www.mellanox.com/related-docs/prod_cables/DS_40GbE_Passive%20Copper%20Cables.pdf
1	MSX1036	40Gb Ethernet Switch, 36 ports, QSFP connectors, managed	http://www.mellanox.com/related-docs/user_manuals/SX10XX_User_Manual.pdf

Table 4. *40GbE Hadoop Deployment Networking Bill of Materials*

Five Node using 10 GbE Interconnect

Quantity	Part Number	Description	Link
5	MCX312A-XCBT	ConnectX-3 Ethernet Dual SFP+ Port Adapter	http://www.mellanox.com/related-docs/user_manuals/ConnectX-3_Ethernet_Single_and_Dual_SFP+_Port_Adapter_Card_User_Manual.pdf
1	MC2210130-002	SFP+ to SFP+ cable, 10Gb Ethernet, 2m	http://www.mellanox.com/pdf/prod_cables/DS_Passive_Copper_SFP_10Gb.pdf
1	MSX1024	10Gb Ethernet Switch, up to 60 ports, managed	http://www.mellanox.com/related-docs/prod_eth_switches/SX1024_User_Manual.pdf

Table 5. *10GbE Hadoop Deployment Networking Bill of Materials*

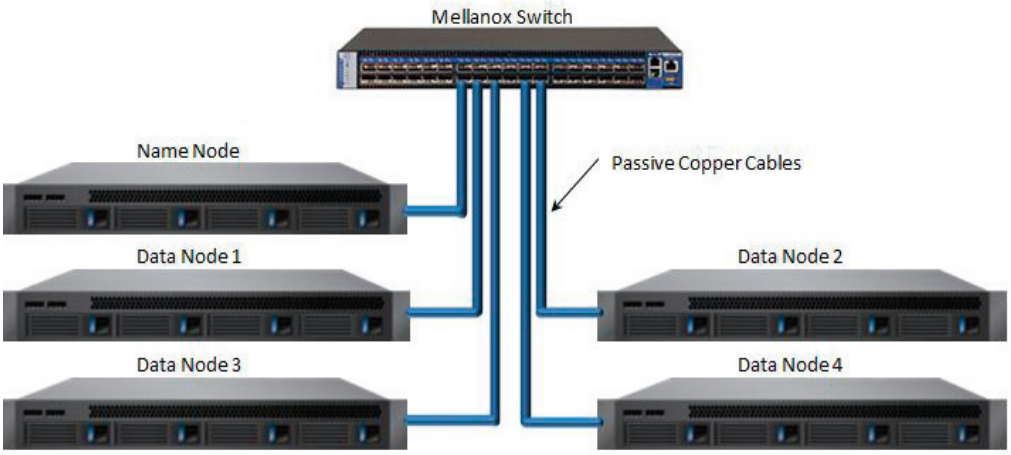

Figure 1: Hadoop, 5 Nodes Deployment

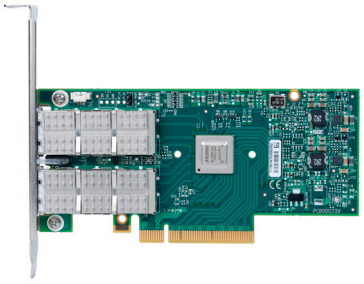

Figure 2: Mellanox FDR InfiniBand and/or 40Gb Ethernet Adapter

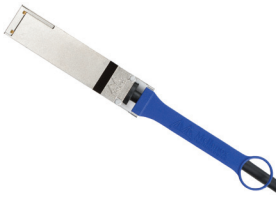

Figure 3: Mellanox QSFP Copper Cable

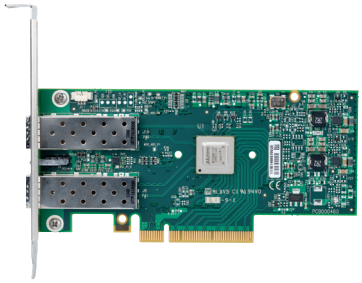

Figure 4: Mellanox 10Gb Ethernet Adapter

Figure 5: Mellanox SFP+ Copper Cable

In the above example, where nodes are connected with a FDR InfiniBand 56Gb/s fabric, the All-to-All available bandwidth will be 18.6Gb/s. Scaling to larger clusters is done in the same fashion. Connection ToR switches with enough bandwidth to satisfy nodes throughputs.

Software Requirements

1. Supported OS
    a. RHEL5.5, 5.6, 5.7, 5.8, 6.0, 6.1, 6.2, 6.3
        i. Corresponding CentOS distributions
        ii. SLES10 sp4, SLES11, SLES sp1, SLES sp2
2. Java Development Kit (JDK) version 1.6.0_25 or higher
3. Mellanox driver 1.5.3 or higher
4. The Hadoop distribution mentioned in section 1 above

Installation

The following section describes the installation of Hadoop on a Linux based machine(s). The supported Linux versions are described in the Software Requirements section.

Installing the Mellanox OpenFabrics Enterprise Distribution (OFED) driver

1. Download the Mellanox OFED driver iso from: http://www.mellanox.com/content/pages.php?pg=products_dyn&product_family=26&menu_section=34
2. Mount the iso (*mount -o loop MLNX_OFED_LINUX-1.5.3-3.1.0-rhel6.3-x86_64.iso /mnt*)
3. Install the missing packages
    a. For namenode (e.g. rhel/centos Software development workstation)
        i. *yum install tcl tk*
    a. For datanode (e.g. rhel/centos Basic Server)
        i. *yum install tcl tk gcc-gfortran*
4. cd /mnt
5. ./mlnxofedinstall
6. Reboot
7. Run connectx_port_config (Choose the right config required InfiniBand or 40GbE Ethernet)
8. Run service openibd restart
9. Verify with the ibstatus command that you have the links active (e.g. port 1 InfiniBand, port 2 Ethernet)

    Infiniband device 'mlx4_0' port 1 status:
    default gid: fe80:0000:0000:0000:0002:c903:00fa:ced1
    base lid: 0x39
    sm lid: 0x2c
    state: 4: ACTIVE
    phys state: 5: LinkUp
    rate: 56 Gb/s (4X FDR)
    link_layer: InfiniBand
10. If you have the LinkUp, you are all set.

Installing Hadoop

Using Mellanox interconnect provides two options of installation:

1. "Vanilla" – Installing Hadoop framework without taking advantage of the RDMA capabilities integrated within Mellanox's end-to-end interconnect. In this mode the data flow will use the TCP/IP stack inherent with the Linux operating system in conjunction with Mellanox drivers.
2. Unstructured Data Accelerator (UDA) Based – Installing Hadoop framework and Mellanox's UDA. In this mode the intermediate data flow will use the RDMA capabilities to accelerate the Map Reduce capabilities. Testing with large data sets (500GB and more) shows over 45% reduction in execution time. To learn more on Mellanox's UDA please visit: http://www.mellanox.com/content/pages.php?pg=products_dyn&product_family=144&menu_section=69

The "Vanilla" Option

Installing Apache Hadoop Distribution 1.0.4 using Dell and Mellanox high-end servers and networks capabilities.

1. Setup the required network (In the example below we add –ib for Infiniband). You will need to edit the portion of the $HADOOP_PREFIX/conf/hadoop-env.sh "NODENAME" to reflect the correct hostname used for the cluster. All hostnames should have DNS setup as well.
2. Download JDK 1.6.x and install (The install location will be your $JAVA_HOME) on all nodes.
3. Update the .bashrc with $JAVA_HOME and change the path to include this as the first choice
4. Add line "export HADOOP_PREFIX=$HOME/ hadoop-1.0.4
5. Copy .bashrc to all the nodes
6. Plan on the disk that will be used for hadoop and you can use preparedisks.sh in $HOME/ hadoop-scripts directory
    a. Use with caution you need edit the script for the disks you need to initialize or you may lose data on your existing disks
7. Create a simple hduser login on all nodes
8. Untar the hadoop-scripts on the home directory of hduser
9. Download http://download.nextag.com/apache/hadoop/common/hadoop-1.0.4/hadoop-1.0.4.tar.gz (You can use wget)
10. cd hadoop-scripts
11. run the crsshkeys.sh script to generate a passwordless ssh login on all nodes (ex: ./crsshkeys.sh hydra001 thru 5). This script creates authorized keys in the .ssh directory
12. chmod g-w ~/.ssh/authorized_keys
13. scp $HOME/.ssh/.authorized_keys hduser@hydra002 (run the same for all the nodes)
14. Test ssh works without password (ssh hydra002)
15. Modify the $HOME/hadoop-scripts/runcmdall.sh script to your cluster name and needs
16. Use the runcmdall.sh script to untar the hadoop-1.0.4.tar.gz on all nodes
17. Check if the $JAVA_HOME is set and java – version does report the JAVA version you have installed (java –version)
    a. [hduser@hydra001-ib ~]$ java -version
    b. java version "1.6.0_33"
    c. Java(TM) SE Runtime Environment (build 1.6.0_33-b04)
    d. Java HotSpot(TM) 64-Bit Server VM (build 20.8-b03, mixed mode)
18. Login from the namenode to all the other nodes to add the host id's or disable the key checking
19. mv $HOME/ hadoop-1.0.4/conf $HOME/ hadoop-1.0.4/conf.empty
20. Copy the conf files to $HOME/ hadoop-1.0.4/conf
21. Modify the files masters, slaves, core-site.xml, hdfs-site.xml, mapred-site.xml , hadoop-env.sh files to suit your environment
22. scp –r $HOME/ hadoop-1.0.4/conf hduser@<nothernodes>:/$HOME/ hadoop-1.0.4/conf
23. $HOME/hadoop-scripts/runcmdall.sh "mkdir -p /data01/hduser/dfs/nn /data02/hduser/dfs/nn"
24. $HOME/hadoop-scripts /runcmdall.sh "mkdir -p /data01/hduser/dfs/dn /data02/hduser/dfs/dn"
25. $HOME/hadoop-scripts /runcmdall.sh "mkdir -p /data01/hduser/mapred/local /data02/hduser/mapred/local"
26. $HOME/hadoop-scripts/runcmdall.sh "chmod go-w /data01/hduser/dfs/dn /data02/hduser/dfs/dn " – Verify the permissions on the datanode slices
    a. It should be: drwxr-xr-x 6 hduser hduser 4096 Feb 28 11:23 /data01/hduser/dfs/dn

27. $HADOOP_PREFIX/bin /hadoop namenode –format –Answer "Y"
28. Start HDFS service
    a. $HADOOP_PREFIX/bin/start-dfs.sh
29. Verify using the jps command if the namenode,secondarynamenode and datanodes in other nodes working.
    a. Namenode should show
    b. [hduser@hydra001-ib hadoop-1.0.4]$ jps
    c. 4731 Jps
    d. 3607 NameNode
    e. 3993 SecondaryNameNode
    f. [hduser@hydra001-ib hadoop-1.0.4]$
    g. Datanode will show "DataNode"
30. Create required tmp HDFS directories
    a. $HADOOP_PREFIX/bin/hadoop fs -mkdir /tmp
    b. $HADOOP_PREFIX/bin/hadoop fs -chmod -R 1777 /tmp
31. Verify all nodes are up and storage is being shown correctly
    a. $HADOOP_PREFIX/bin/hadoop dfsadmin –report
32. Start mapreduce services
    a. $HADOOP_PREFIX/bin/start-mapred.sh
33. Verify using the jps command if the namenode,secondarynamenode and datanodes in other nodes working.
    a. Namenode should show
    b. [hduser@hydra001-ib hadoop-1.0.4]$ jps
    c. 4731 Jps
    d. 3607 NameNode
    e. 3993 SecondaryNameNode
    f. 4125 JobTracker
    g. [hduser@hydra001-ib hadoop-1.0.4]$
    h. Datanodes (all other nodes) should show "DataNode" & "TaskTracker"
34. Run the terasort to verify the cluster is working fine
    a. $HOME/hadoop-scripts/runterasort.sh
    b. Check the namenode ip ex: http://hydra001:50030 – You should see the Job Tracker page with the jobs running
35. If you see the Terasort job completed on the JT page, You are all set!!

Adding the UDA Package on top of Vanilla.

Make sure the Mellanox ConnectX®-3 cards are properly installed on your Name Node and Data Nodes before starting the UDA installation.

To install UDA, you should first follow the Hadoop installation in the *"Vanilla Option"* section.

After successfully installing the "vanilla" Hadoop version, follow these next steps:

Set the ulimit to unlimited:

ulimit –l unlimited

Increase the maximum number of memory translation table segments per HCA

Check for the following settings in: /etc/modprobe.d/ib_ipoib.conf

"options mlx4core log_numm_mtt=XX"

If present, check the value of mtt and based on your memory footprint, this value needs to be adjusted (Ex: 64Gb of memory, you can set this to 24). More information on this can be obtained here: http://www.open-mpi.org/faq/?category=openfabrics#ib-low-reg-mem).

If not present, create a mofed.conf with the setting:

echo "options mlx4_core log_num_mtt=24" > /etc/modprobe.d/mofed.conf

Reboot the server for the settings to take effect.

UDA Integration (To be executed for all nodes)

Patch the plugin (describe blew is the CDH3u4 and Hadoop 0.20.2 patch)

cd ../<hadoop dir> (ex: cd ../$HADOOP_HOME ,)
ls –ld hadoop-0.20.2-cdh3u4
drwxr-xr-x. 17 root root 4096 Sep 4 04:58 hadoop-0.20.2-cdh3u4
patch -p0 < cdh3u4.patch
cd <hadoop dir> (ex: cd /usr/lib/hadoop-0.20.2-cdh3u4)

Run ant

Copy the jar files from the build directory again to $HADOOP_HOME

Install the UDA RPM

rpm –ivh libuda-3.0.1-4453.el6.x86_64.rpm

Verify the rpm install:

rpm -ql libuda
/usr/lib64/uda/libuda.so
/usr/lib64/uda/set_hadoop_slave_property.sh
/usr/lib64/uda/uda-CDH3u4.jar
/usr/lib64/uda/uda-hadoop-0.20.2.jar
/usr/lib64/uda/uda-hadoop-1.x.jar
/usr/share/doc/libuda-3.0.1/LICENSE.txt
/usr/share/doc/libuda-3.0.1/README

Add UDA jar to classpath of hadoop-env.sh:

export HADOOP_CLASSPATH="$HADOOP_CLASSPATH":/usr/lib64/uda/uda-CDH3u4.jar
The Jar file would be different if you using a different distribution

UDA Configuration

Add the following properties in the files mentioned. For more information on these properties, please refer to the "Mellanox Unstructured Data Accelerator Quick start guide".

1. File hdfs-site.xml

```
<property>
    <name>dfs.datanode.dns.interface</name>
    <value>ib0</value>
</property>
```

2. File mapred-site.xml

```
<property>
    <name>mapred.rdma.setting</name>
    <value>1</value>
</property>
<property>
    <name>mapred.rdma.buf.size</name>
    <value>1024</value>
</property>
<property>
    <name>mapred.map.tasks.speculative.execution</name>
    <value>false</value>
</property>
<property>
    <name>mapred.reduce.tasks.speculative.execution</name>
```

```
        <value>false</value>
    </property>
    <property>
        <name>mapred.rdma.cma.port</name>
        <value>1</value>
    </property>
    <property>
        <name>mapred.rdma.cma.port</name>
        <value>9011</value>
    </property>
    <property>
        <name>mapred.reduce.slowstart.completed.maps</name>
        <value>0.95</value>
    </property>
    <property>
        <name>mapred.rdma.wqe.per.conn</name>
        <value>1024</value>
    </property>
    <property>
        <name>mapred.tasktracker.shuffle.provider.plugin</name>
        <value>com.mellanox.hadoop.mapred.UdaShuffleProviderPlugin</value>
    </property>
    <property>
        <name>mapred.reducetask.shuffle.consumer.plugin</name>
        <value>com.mellanox.hadoop.mapred.UdaShuffleConsumerPlugin</value>
    </property>
```

Testing UDA functionality

Execute a Terasort test.

For example: Execute a 300GB Tergen and Terasort job.

hadoop jar $HADOOP_HOME/hadoop-examples-.jar teragen 3000000000 /users/hadoop/terasort-input*

hhadoop jar /usr/lib/hadoop-0.20/hadoop-examples-.jar terasort /users/hadoop/terasort-input /users/hadoop/terasort-output*

UDA troubleshooting

1. Verify the plugin ability patch inside the hadoop jar
   jar -tf /usr/lib/hadoop/hadoop-core-.jar | grep ShuffleConsumerPlugin.class*
   jar -tf /usr/lib/hadoop/hadoop-core-.jar | grep ShuffleProviderPlugin.class*
2. Verify the UDA rpm installation exist
   rpm -qa | grep -i uda
3. Verify the UDA configuration parameters are set
   grep -i uda <hadoop configuration directory>
4. Examine tasktracker log files for any memory errors
   Ex : "MSG=Cannot allocate memory (errno=12)" – This error shows that the mtt value + number of reducers are not able to allocate memory. Reduce the number of reducers or decrease the mtt value based on the guideline provided. More information is provided in the tuning section of the quick start guide.

Scaling the Cluster Size

Adding nodes or building a cluster with more nodes than a single rack can contain, is a common practice. The installation of servers and the network should adhere to the target application performance

requirements. Additional nodes provides additional storage space and compute power.

Scaling beyond the single switch requires the installer to take into consideration the needed throughput of the single server and the rack.

In an "All-to-All" setting, we've found that at least 10Gb of true bandwidth is required in order to scale effectively.

High Availability

When considering High Availability (HA) features, one should take advantage of the framework capabilities. For the interconnect consideration, there are several options to consider:

The first option would be doubling the number of switches and cables by using a dual rail configuration. Dual rail configuration is enabled by using a second port on the server's adapter card connected to a second switch. In this configuration, the node is connected to two fabrics in parallel, eliminating any single point of failure, in terms of connectivity from the server to its adjacent nodes.

The second option would be adding a secondary networking card to the servers and using it as the failover point, in the event the primary card fails or "hangs off". In such a configuration, the number of switch ports required is doubled.

The last option would be combining the first two options and doubling both the adapter cards installed and the number of switches in the configuration.

Appendix: Setup Scripts

File: checkconfig.sh

```
echo "Check Hadoop Home"
echo $HADOOP_HOME
echo "Hadoop Config Dir"
echo $HADOOP_CONF_DIR
echo "Current Active config"
ls -ld /etc/hadoop/conf
echo "Current active binary config"
ls -ld /usr/lib/hadoop*
echo "Checking the conf directory on the HADOOP_HOME"
ls -ld /usr/lib/hadoop-0.20/conf
```

File: checkdns.sh

```
nslookup `hostname`
ping -c 1 `hostname`
```

File: cleanlogs.sh

```
rm -rf /var/log/hadoop/*.out* /var/log/hadoop/*.log* /var/log/hadoop/metrics/*.log
/var/log/hadoop/SecurityAuth.audit /var/log/hadoop/job*.xml /var/log/hadoop/userlogs/*
touch /var/log/hadoop/metrics/dfsmetrics.log
touch /var/log/hadoop/metrics/jvmmetrics.log
touch /var/log/hadoop/metrics/mrmetrics.log
touch /var/log/hadoop/SecurityAuth.audit

chown hdfs:hdfs /var/log/hadoop/metrics/dfsmetrics.log
chown hdfs:hadoop /var/log/hadoop/metrics/jvmmetrics.log
chown mapred:mapred /var/log/hadoop/metrics/mrmetrics.log
chown hdfs:hadoop /var/log/hadoop/SecurityAuth.audit
chown hdfs:hadoop /var/log/hadoop/metrics
chmod g+rw /var/log/hadoop/metrics/dfsmetrics.log
chmod g+rw /var/log/hadoop/metrics/jvmmetrics.log
chmod g+rw /var/log/hadoop/metrics/mrmetrics.log
chmod g+rw /var/log/hadoop/SecurityAuth.audit
```

```
chmod g+rw /var/log/hadoop
chmod g+rw /var/log/hadoop/metrics
```

File: create-hadoop-sysusers.sh

```
groupadd -r hdfs
groupadd -r mapred
groupadd hadoop
useradd -r -g hdfs -G hadoop -c 'Hadoop HDFS' -d /usr/lib/hadoop-0.20 hdfs
useradd -r -g mapred -G hadoop,hdfs -c 'Hadoop MapReduce' -d /usr/lib/hadoop-0.20 mapred
useradd -g hadoop -G hdfs -m -c 'Hadoop User' -d /home/hadoop hadoop
```

File: cdhdfsdirs.sh

```
# This script creates all required HDFS directories for the
# cluster including the user of the cluster hadoop

cd $HADOOP_HOME
sudo -u hdfs bin/hadoop fs -chown -R hdfs:hadoop /
sudo -u hdfs bin/hadoop fs -chmod go+rx /
sudo -u hdfs bin/hadoop fs -chmod go-w /
sudo -u hdfs bin/hadoop fs -mkdir /tmp
sudo -u hdfs bin/hadoop fs -chmod -R 1777 /tmp
sudo -u hdfs bin/hadoop fs -mkdir /mapred/system
sudo -u hdfs bin/hadoop fs -chown mapred:hadoop /mapred/system
sudo -u hdfs bin/hadoop fs -mkdir /user/hadoop
sudo -u hdfs bin/hadoop fs -chown -R hadoop:hadoop /user/hadoop
sudo -u hdfs bin/hadoop fs -chmod go-rwx /mapred/system
sudo -u hdfs bin/hadoop fs -ls /
sudo -u hdfs bin/hadoop fs -ls /mapred/system
```

File: crsshkeys.sh

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
ssh root@hydra002-ib "ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa"
ssh root@hydra002-ib cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
ssh root@hydra003-ib "ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa"
ssh root@hydra003-ib cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
scp ~/.ssh/authorized_keys  root@hydra002-ib:/root
scp ~/.ssh/authorized_keys  root@hydra003-ib:/root
```

File: initialize-cluster.sh

```
rm -rf /data1/dfs/nn/* /data1/dfs/dn/* /data1/mapred/local/*
rm -rf /data2/dfs/nn/* /data2/dfs/dn/* /data2/mapred/local/*
rm -rf /data3/dfs/nn/* /data3/dfs/dn/* /data3/mapred/local/*
rm -rf /data4/dfs/nn/* /data4/dfs/dn/* /data4/mapred/local/*
rm -rf /data5/dfs/nn/* /data5/dfs/dn/* /data5/mapred/local/*
```

File: newslice-fixperm.sh

```
# Create the /data?? directories and initialize with the
# directories for namenode, datanode & mapred

mkdir -p /data01/dfs/nn
```

```
mkdir -p /data01/dfs/dn
mkdir -p /data01/mapred/local

chown -R hdfs:hadoop /data01

chown -R hdfs:hadoop /data01/dfs
chmod -R 700 /data01/dfs
chown -R mapred:hadoop /data01/mapred
chmod -R 755 /data01/mapred

…. For all data nodes

#Create the metrics and log directories

mkdir -p /var/log/hadoop/metrics
mkdir -p /var/log/hadoop/userlogs

chown -R hdfs:hadoop /var/log/hadoop
chown -R mapred:mapred /var/log/hadoop/userlogs

#Create the directory for hadoop pid's

mkdir -p /var/hadoop
chown hdfs:hadoop /var/hadoop
chmod g+rwx /var/Hadoop
```

File: prepareddisks.sh

```
# ***Use this script with caution*** It can wipe the entire disk
# clean ** this script shows an example of 3 disks
# sdb,sdbc & sdd.

parted /dev/sdb mkpart primary ext4 0% 100%
mkfs.ext4 /dev/sdb1

parted /dev/sdc mkpart primary ext4 0% 100%
mkfs.ext4 /dev/sdc1

parted /dev/sdd mkpart primary ext4 0% 100%
mkfs.ext4 /dev/sdd1
```

File: runcmdall.sh

```
# Use this script to run commands on all clusters or scripts from # the same directory
# ex: ./runcmdall "ls -l /etc/hadoop/conf" shows all files in the # conf direcotry

echo "Running on Hydra-2"
ssh root@hydra002 $1
echo "Running on Hydra-3"
ssh root@hydra003 $1
echo "Running on Hydra-4"
ssh root@hydra004 $1
echo "Running on Hydra-5"
ssh root@hydra005 $1
echo "Running on Hydra-1"
ssh root@hydra001 $1
```

File: testdfsio.sh

```
cd $HADOOP_HOME
sudo -u hdfs bin/hadoop jar $HADOOP_HOME/hadoop-test-*.jar TestDFSIO -write -nrFiles 10
-fileSize 1000
sudo -u hdfs bin/hadoop jar $HADOOP_HOME/hadoop-test-*.jar TestDFSIO -read -nrFiles 10
-fileSize 1000
sudo -u hdfs bin/hadoop jar $HADOOP_HOME/hadoop-test-*.jar TestDFSIO -clea
```

References

[1] "Hadoop acceleration through network levitated merge", Yandong Wang; Xinyu Que; Weikuan Yu; Goldenberg, D.; Sehgal, D., International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2011

350 Oakmead Parkway, Suite 100, Sunnyvale, CA 94085
Tel: 408-970-3400 • Fax: 408-970-3403
www.mellanox.com