Guidelines for Integrating a Hadoop Big Data System with Petascale Supercomputer

Content

1. Introduction

The Hadoop Big Data System located at the UNWE Sofia (DGS) is the foundation of Apache Hadoop under the management of Cloudera management. Its architecture consists of 3 NameNode, 2 ManagementNode, EdgeNode and multiple DataNode. Each of the DataNode has 128GB of RAM and 120TB of disk memory, with the total disk space of the Big Data System being about 4.5 PB and the total RAM about 4.6TV, and the total number of cores in the Big Data System being 576. What is specific about the architecture of the Big Data System is that it is cancer-oriented – the DataNodes are located in separate racks, with the network connection (local network) between the individual racks at a speed of 40Gbps, and the network connection between the individual DataNode at a speed of 10Gbps. Each DataNode is built of processors with an Intel Xeon Bronze architecture. The big data system is protected by a UPS system secured by a diesel generator.

The supercomputer in Tech Park Sofia has a size combining 12 computational Direct Liquid Cooling BullSequana server crabs. The Supercomputer's platform is built on water-cooled AMD EPYC processors building 376 computing nodes. The number of processor cores is 144,384 with a total RAM of 300TB. The optimal functionality is guaranteed by 2 PB high-speed disk storage space. The supercomputer is protected from power outages by an uninterruptible power supply of 1 MW.

The Hadoop big data system is connected to 10Gbps MAN connections with major cities in Bulgaria. There are also 2 separate connections of 10Gbps to the Supercomputer in the Tech Park and the upcoming for commissioning another supercomputer in the IICT of BAS - Fig. 1

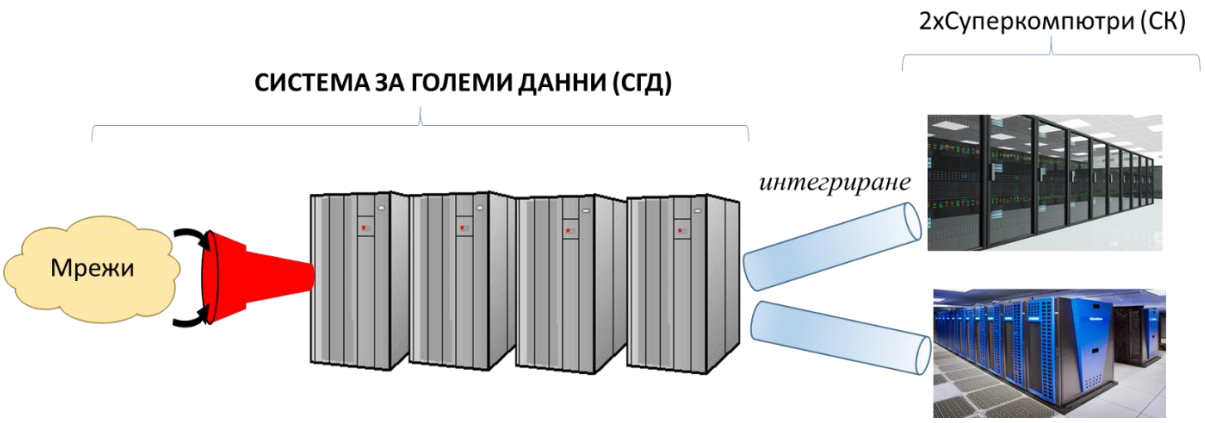

Fig.1

The Big Data System at the UNWE is connected to networks for complex reception and transmission of data. These networks are the following 5 pieces:

- MAN with a common data covers of 40Gbps transmitted and with direct connections to Plovdiv, Gabrovo, Ruse and Varna with 10Gbps each;
- WAN with 10Gbps;
- National LoRaWAN network covering the main cities in Bulgaria;
- TTN – International LoRaWAN network providing data transmission with over 110 countries around the world;
- 4G/5G network for devices located in these networks.

2. Functionality of the Big Data System provided to a Supercomputer

When integrating the Hadoop Hadoop System of UNWE with a Supercomputer, the Big Data System can perform the following functions:

a) Transit data transmission for a Supercomputer. This means, using the multiple network systems with high reliability of the DGS, to accept data from different sources and, after potential preprocessing, to feed it directly to a Supercomputer for processing;

b) Data storage on a Supercomputer. This means
   - to receive data from different sources and store it in its disk space for subsequent use by a Supercomputer – data storage on a Supercomputer,
   - to accept certain semi-structured and complete non-structured data for Reconversion and store them for post-processing by a Supercomputer

- store resulting data from the processing of the Supercomputer for subsequent Analysis, subsequent transmission from the DGS networks to the End User, or simply for storage due to lack of sufficient disk space in the Supercomputer.

c) To perform Analysis of results obtained from the operation of the Supercomputer due to the presence of multiple software tools for data analysis in Big Data Systems.

d) The supercomputer will receive as a result partially formed dzanni to be transmitted to the Big Data System for subsequent processing through Impala, Spark or user-created programming code.

e) To represent a Supercomputer Data Cybersecurity Object;

f) To act as a Cybersecurity Data Tool for a Supercomputer;

3. Supercomputer functionalities provided to the Big Data System

The supercomputer can be seen as an additional server to the Big Data System to perform specialized high-performance computing. A natural architectural extension of the Big Data System is to work together with SQL Databases (SQL based Relational Databases), with ERP (Enterprise Resource Planning) systems, with Business Process Management (BPM), with CMS (Content Management System), or with NoSQL Databases, which are not integrated into Big Data Systems.

4. Construction of a single-point connection of the Multi-Cluster Hadoop Big Data System at the UNWE with the Switch in TechPark Sofia, based on the MAN network

A dedicated one-way connection (point-in-point) between the Hadoop Big Data System at the UNWE has been built, which is essentially a multi-cluster Hadoop system (consisting of 4 Hadoop clusters connected in a multi-cluster system) with a MAN switch in Tech Park Sofia, to which the Supercomputer is connected. The multi-cluster Hadoop system consists of a Central Cluster at the UNWE, a cluster at Plovdiv University "Paisii Hilendarski", a Cluster at the Technical University of Gabrovo and a Cluster at the University of Ruse "Angel Kanchev".

The conceptual Architecture of the created multi-cluster connection with the Supercomputer is presented in Figure 2.

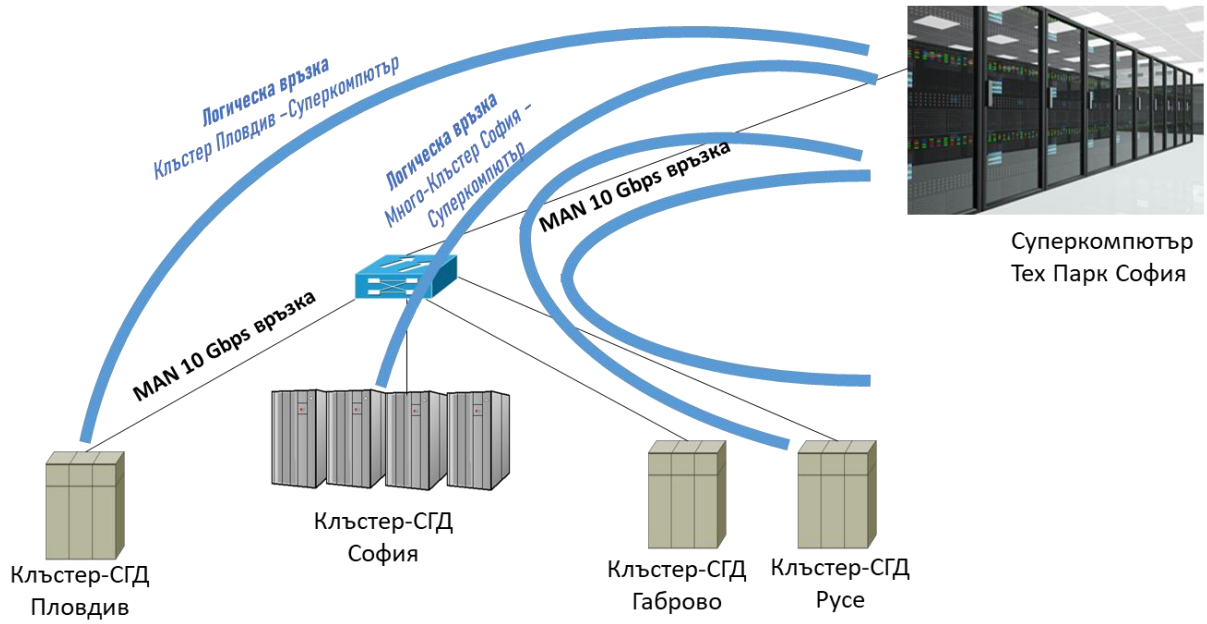

Fig.2

Physically, all these 4 clusters of DGS are connected to a Switch located at the UNWE – Sofia, which in turn is connected to the Supercomputer in Tech Park Sofia. This means that each of the 4 Hadoop clusters are logically connected to the Supercomputer, i.e. the following Logical connections between Hadoop cluster and supercomputer have been made:

- Cluster-SGS at the UNWE – Sofia with a Supercomputer in Tech Park Sofia;
- Cluster-DGS at Plovdiv University "Paisii Hilendarski" – Plovdiv with a Supercomputer in Tech Park Sofia;
- Cluster-DGS at the Technical University of Gabrovo with a Supercomputer in Tech Park Sofia;
- Cluster-SGS at the University of Ruse "Angel Kanchev" – Ruse with a Supercomputer in Tech Park Sofia.

This means that the Multi-Cluster Hadoop System provides integration of 4 universities in Bulgaria with the Supercomputer in Tech Park Sofia - UNWE, Plovdiv University "Paisii Hilendarski", Technical University - Gabrovo and University of Ruse "Angel Kanchev". After the establishment of the MAN connections between the 4 mentioned universities a connection between the UNWE and Tech Park Sofia was built. The establishment of this relationship and its testing is presented below:

4.1. Activation of physical connectivity

Figure 3 shows the physical connectivity activity (gate status "UP").

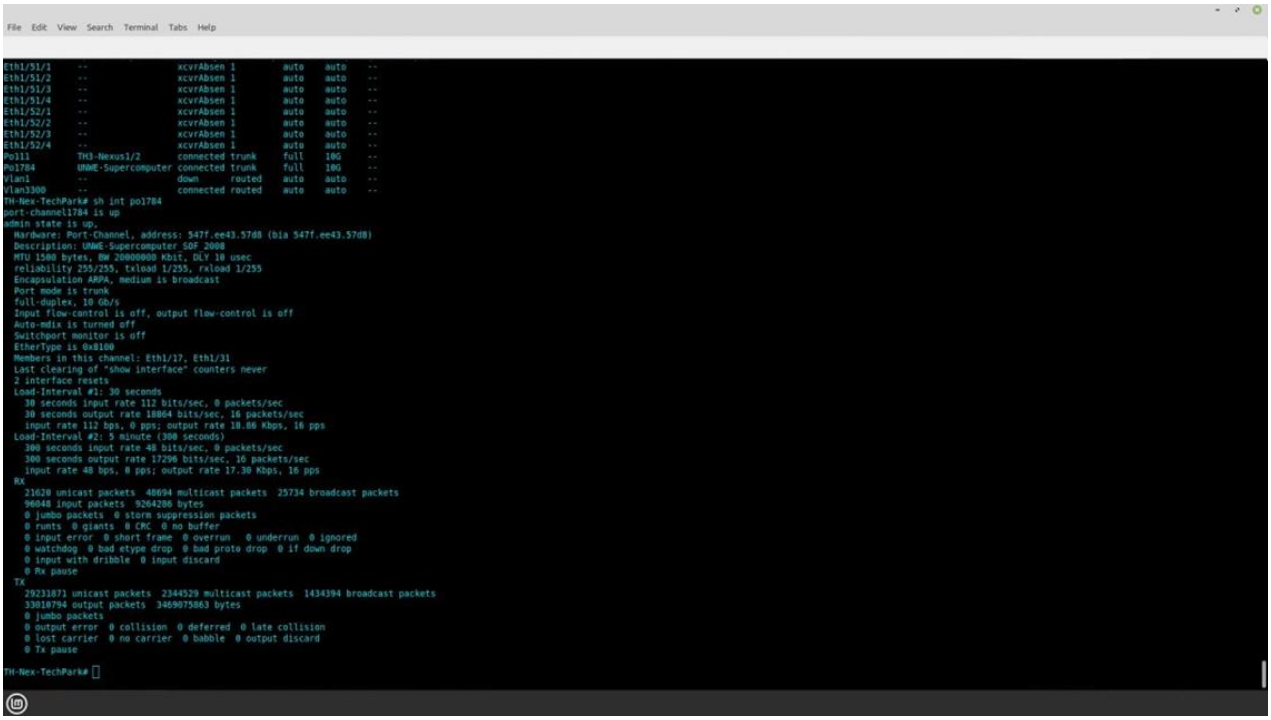

Fig.3

4.2. The presence of two-way MAC addresses

Figure 4 shows the representation of bidirectional MAC addresses of both parts of physical connectivity.

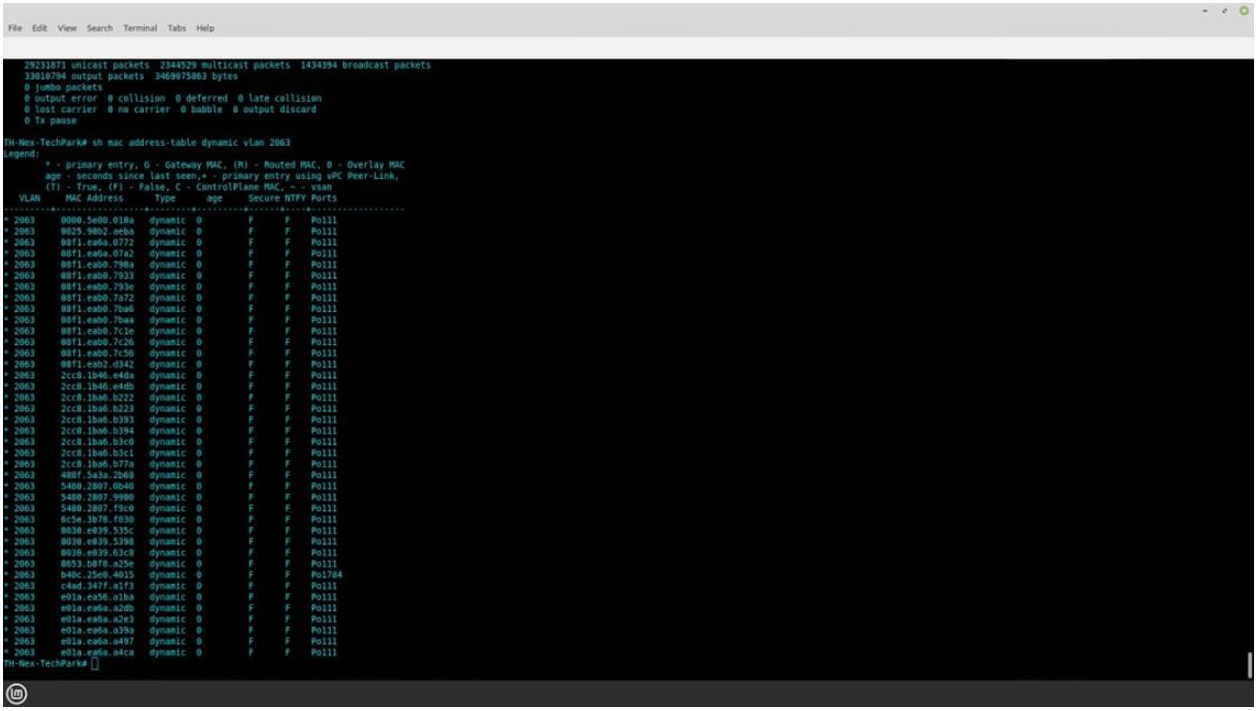

Fig.4

4.3. Testing the speed for data transmission on the built MAN network

In order to test the data rate over the built MAN network, the iperf3 software tool is used.

The perf tool is a tool for measuring and tuning network performance. It is a cross-platform tool that can produce standardized performance measurements for any network. Iperf has client and server functionality and can create data streams to measure bandwidth between ends in one or both directions. A typical iperf output contains a stamped-time report on the amount of data transferred and the measured throughput.

Data streams can be either a Transmission Management Protocol (TCP) or a User Datagram Protocol (UDP):

- UDP: When used to test UDP capacity, iperf allows the user to specify the size of the datagram and provides results on datagram throughput and packet loss.
- TCP: When used to test TCP capacity, iperf measures payload throughput. Iperf uses 1024 × 1024 for mebibytes and 1000 × 1000 for megabytes.

Iperf is open source software written in C, and runs on a variety of platforms including Linux, Unix, and Windows. The availability of the source code allows the user to look closely at the measurement methodology.

Iperf3 is rewriting iperf from scratch to create a smaller, simpler code base. It also includes a version of a library that allows other programs to use the functionality provided. Iperf3 is single-threaded, while iperf2 is multi-threaded. Iperf3 was launched in 2009 with the first edition in January 2014. Iperf3 is not backwards compatible with iperf2. Iperf3 officially only supports Linux.

Figure 5 shows a Printout of iperf3 test made by the Switch in Supercomputer in TechPark Sofia to the Switch of the Multi-Cluster Distributed Hadoop System at the UNWE bidirectionally, proving the data rate 10Gbps.

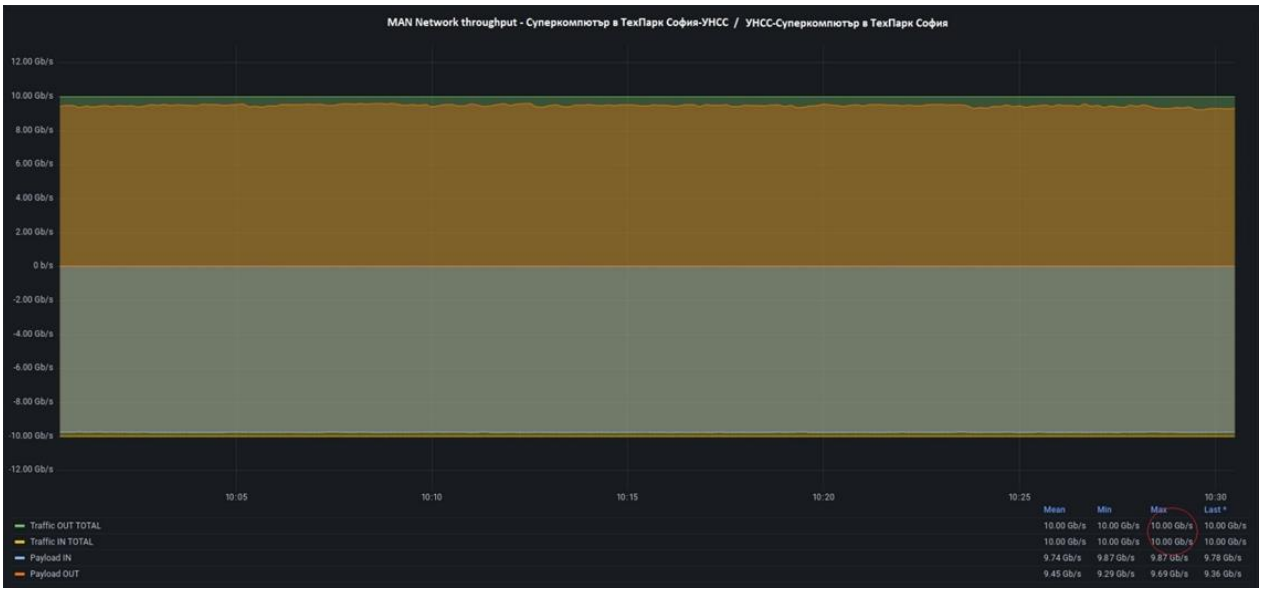

Fig.5

Figure 5 presents a printout of iperf3 test made by the Switch of the Multi-Cluster Distributed Hadoop System at the UNWE to the Switch in the Supercomputer in TechPark Sofia bidirectionally, proving the data rate 10Gbps.

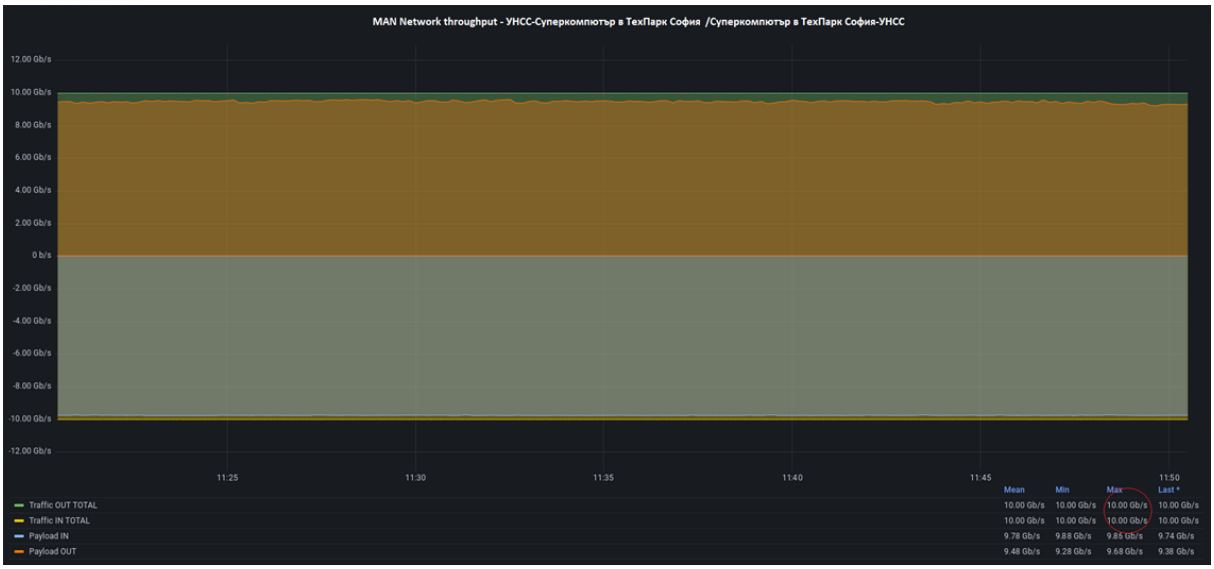

Fig.6

5. Presentation of data in Big Data Systems as NFS files for use by a Supercomputer

The big data system operating with HDFS files offers many serious advantages to the Supercomputer operating NFS files, because HDFS files have built-in error protection (error tolerance) unlike the NFS file system. The Big Data System simulates HDFS files as NFS files so that they can be used by the Supercomputer, but at the same time provide tolerance of error. Basically, the NFS software component in Big Data Systems (also called NFS Gateway) allows you to access files as if the files were located on the local machine, even though the Big Data System is located as HDFS files. The main difference between NFS and HDFS file systems is replication/fault tolerance. The HDFS is designed to provide duration of operation in failures. NFS has no built-in fault tolerance. Besides the fault tolerance, HDFS supports multiple file replicas. This eliminates (or relieves) the usual predicament of many Supercovüter customers who have access to a single file. Because files have multiple replicas on different physical disks located on different physical racks, reading performance scales better than NFS.

The NFS Gateway in the Hadoop Big Data System supports NFSv3 and allows HDFS data as NFS data to be "mounted" to the Supercomputer's operating system (a command to make the Supercomputer make HDFS data available in the Big Data System) and operate it as part of the

Supercomputer's local file system. NFS Gateway currently supports and enables the following usage patterns:

- Supercomputer users can view the HDFS file system through their local file system.
- Supercomputer users can download files from the HDFS file system to their local file system.
- Supercomputer users can upload files from their local file system directly to the HDFS file system.
- Users of the Supercomputer can operate directly with HDFS data. The Supercomputer can expand a file and add a file, but random recording is not supported (due to the specifics of the Big Data System).

NFS Gateway functions in the same way as a client as Hadoop JAR files. For this purpose, NFS Gateway can be installed on the same host (server in the Hadoop system) as DataNode or on NameNode.

NFS Gateway uses a proxy user to proxy all users who have access to NFS mounts. In unprotected mode, the user-administrator of the Big Data System managing the NFS Gateway is a proxy user, while in secure mode the user-administrator in the Big Data System in Kerberos keytab is a proxy user. Suppose that the proxy user is "nfsserver" and the users belonging to the groups "users-group1" and "users-group2" use NFS mounts, then in the core-site.xml of NameNode the following two properties must be set and only NameNode must be restarted after the configuration is changed. Such configuration is as follows:

```
<property>
  <name>hadoop.proxyuser.nfsserver.groups</name>
  <value>root,users-group1,users-group2</value>
  <description>
         The 'nfsserver' user is allowed to proxy all members of the
'users-group1' and
         'users-group2' groups. Note that in most cases you will need to
include the
         group "root" because the user "root" (which usually belonges to
"root" group) will
         generally be the user that initially executes the mount on the NFS
client system.
         Set this to '*' to allow nfsserver user to proxy any group.
  </description>
</property>

<property>
  <name>hadoop.proxyuser.nfsserver.hosts</name>
  <value>nfs-client-host1.com</value>
  <description>
         This is the host where the nfs gateway is running. Set this to '*'
to allow
```

```
        requests from any hosts to be proxied.
  </description>
</property>
```

The above is the only required configuration for NFS Gateway in unprotected mode. For Kerberized hadoop clusters, the following configurations should be added to the hdfs-site.xml for the gateway (NOTE: replace the "nfsserver" string with the proxy username and make sure the user contained in the keytab is also the same proxy user):

```
<property>
   <name>nfs.keytab.file</name>
   <value>/etc/hadoop/conf/nfsserver.keytab</value> <!-- path to the nfs gateway keytab -->
 </property>

 <property>
   <name>nfs.kerberos.principal</name>
   <value>nfsserver/_HOST@YOUR-REALM.COM</value>
 </property>
```

If the HDFS NFS Gateway is required to be accessed by a UNIX-based operating system, the following configuration setting must be set:

```
<property>
  <name>nfs.aix.compatibility.mode.enabled</name>
  <value>true</value>
</property>
```

HDFS super-user is a user with access rights to NameNode and the super-user can do anything so that permission checks never fail for the super-user. If the following property is configured, the NFS client super-user can access any HDFS file. By default, the super user is not configured in the NFS Gateway. Even if the super-user is configured, "nfs.exports.allowed.hosts" still goes into effect. For example, the super-user will not be able to write to HDFS files through the gateway if the NFS client host is not allowed write access to "nfs.exports.allowed.hosts"

```
<property>
  <name>nfs.superuser</name>
  <value>the_name_of_hdfs_superuser</value>
</property>
```

6. Start an application in a Supercomputer from a Big Data System

A scheduler is software that implements a package system on a supercomputer. Supercomputer users do not perform their calculations directly and interactively (as they do on their personal workstations or laptops), instead they send noninteractive batch jobs to the scheduler. The scheduler stores batch tasks, evaluates their resource and priority requirements, and allocates tasks to appropriate computing nodes. Unlike login nodes (for compiling and testing user software) and their interactive use, computing nodes in the Supercomputer are usually not directly accessible (e.g. via ssh). Thus, the scheduler is the interface for users of the Login Supercomputers to send work to the computing nodes. This requires the user to ask the scheduler for memory time and resources and to specify the application in a job script. This job script can then be fed to the batch system through the scheduler, who will first add the job to a to-do queue. Based on the resources the job needs, the scheduler will decide when the job will leave the queue and on which (part of) the rear nodes will run.

Generally speaking, each scheduler has three main objectives:

- Minimizing the time between the submission of the job and the completion of the job: no job should remain in the queue for long periods of time
- optimizing the use of the processors of the Supercomputer: the central processing units of the supercomputer are one of the main resources for a large application; Therefore, only a few time intervals should exist in which the processor is not working
- Maximize work productivity by selling as many tasks as possible per unit of time in the Supercomputer.

SLURM (Simple Linux Utility for Resource Management) is a cluster management and task scheduling system. This is the software most used in Supercomputers to manage resources. It is open source software.

In order to perform a task to a Supercomputer, a connection to a request node must first take place. For each cluster there is at least one query node named <cluster>-gw, e.g.: phoenix-gw, hm-gw, etc.

The main functions of Slurm include:

- Highly scalability (plans up to 100,000 independent jobs per 100,000 sockets)
- High performance (up to 1000 job submissions per second and 600 job executions per second)
- Highly configurable with about 100 add-ons
- Availability of a Timetable for Fair Sharing of Resources
- Preventive and group scheduling (cutting off the time of parallel tasks)
- Integration with a database
- Resource allocation optimized for network topology and node topology (sockets, cores and hyperthreads)
- Advance reservation
- Different operating systems can be run for each job
- Supports arrays of tasks
- Profiles the work (periodically samples CPU usage, memory usage, power consumption, network usage and file system of each task)
- Supports MapReduce.

The observed state of the Supercomputer includes: number of processors, size of RAM, size of temporary disk space and state (UP, DOWN, etc.). Additional information about the Supercomputer includes task weight (preference when allocating work) and functions (arbitrary information such as speed or processor type). The individual servers in the Supercomputer are grouped into partitions. Partition information includes: name, list of connected nodes, status (UP or DOWN), maximum uptime limit, maximum number of nodes per job, group access list, priority (important if nodes are on multiple partitions), and shared node access policy with option level of oversubscription for group scheduling (e.g., YES, NO, or FORCED:2). An example of Slurm configuration is shown below:

```
#
# Sample /etc/slurm.conf
#
```

```
SlurmctldHost=linux0001  # Primary server
SlurmctldHost=linux0002  # Backup server
#
AuthType=auth/munge
Epilog=/usr/local/slurm/sbin/epilog
PluginDir=/usr/local/slurm/lib
Prolog=/usr/local/slurm/sbin/prolog
SlurmctldPort=7002
SlurmctldTimeout=120
SlurmdPort=7003
SlurmdSpoolDir=/var/tmp/slurmd.spool
SlurmdTimeout=120
StateSaveLocation=/usr/local/slurm/slurm.state
TmpFS=/tmp
#
# Node Configurations
#
NodeName=DEFAULT CPUs=4 TmpDisk=16384 State=IDLE
NodeName=lx[0001-0002] State=DRAINED
NodeName=lx[0003-8000] RealMemory=2048 Weight=2
NodeName=lx[8001-9999] RealMemory=4096 Weight=6 Feature=video
#
# Partition Configurations
#
PartitionName=DEFAULT MaxTime=30 MaxNodes=2
PartitionName=login Nodes=lx[0001-0002] State=DOWN
PartitionName=debug Nodes=lx[0003-0030] State=UP Default=YES
PartitionName=class Nodes=lx[0031-0040] AllowGroups=students
PartitionName=DEFAULT MaxTime=UNLIMITED MaxNodes=4096
PartitionName=batch Nodes=lx[0041-9999]
```

Literature

1. SLIURM Workload Manager, https://slurm.schedmd.com/documentation.html
2. What is Slurm and is it Still Relevant for Modern Workloads?, https://www.run.ai/guides/slurm#:~:text=Slurm%20is%20a%20system%20for,user%20to%20a%20compute%20node.
3. Deploy an HPC cluster with Slurm, https://cloud.google.com/hpc-toolkit/docs/quickstarts/slurm-cluster
4. How to test available network bandwidth using 'iperf', https://www.dell.com/support/kbdoc/en-bg/000139427/how-to-test-available-network-bandwidth-using-iperf
5. How to use iPerf3 to test network bandwidth, https://www.techtarget.com/searchnetworking/tip/How-to-use-iPerf-to-measure-throughput
6. Using iPerf to Test Network Speed and Bandwidth, https://woshub.com/testing-network-bandwidth-using-iperf/
7. HDFS NFS Gateway, https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsNfsGateway.html
8. Using the NFS Gateway for accessing HDFS, https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.5/data-storage/content/using_the_nfs_gateway_for_accessing_hdfs.html
9. Adding and Configuring an NFS Gateway, http://188.93.19.26/static/help/topics/admin_hdfs_nfsgateway.html