EPJ Web of Conferences **214**, 04049 (2019) https://doi.org/10.1051/epjconf/201921404049
CHEP 2018

Using Lustre and Slurm to process Hadoop workloads and extending to the WLCG

Daniel Traynor[1,*] and *Terry* Froy[1,**]

[1]School of Physics and Astronomy, Queen Mary University of London, Mile End Road, E1 4NS, UK

Abstract. The Queen Mary University of London Grid site has investigated the use of its Lustre file system to support Hadoop work flows. Lustre is an open source, POSIX compatible, clustered file system often used in high performance computing clusters and is often paired with the Slurm batch system. Hadoop is an open-source software framework for distributed storage and processing of data normally run on dedicated hardware utilising the HDFS file system and Yarn batch system. Hadoop is an important modern tool for data analytics used by a large range of organisation including CERN. By using our existing Lustre file system and Slurm batch system, the need to have dedicated hardware is removed and a single platform only has to be maintained for data storage and processing. The motivation and benefits of using Hadoop with Lustre and Slurm are presented. The installation, benchmarks, limitations and future plans are discussed. We also investigate using the standard WLCG Grid middleware Cream-CE service to provide a Grid enabled Hadoop service.

1 Introduction

The Queen Mary WLCG tier two site has successfully operated a reliable site utilising Lustre and Slurm for several years (more details can be found here [1] [2], [3]). The site has over 5PB of Lustre storage and over 4000 jobs slots all connected by 10Gb/s and 40Gb/s Ethernet network and supports over 20 High Energy Physics (HEP) and Astronomy projects. The main user of the site is the ATLAS experiment [4].

Lustre [5] is an open-source (GPL), POSIX compliant, parallel file system used in over half of the worlds Top 500 supercomputers. Lustre is made up of three components: One or more Meta Data Servers (MDS) connected to one or more Meta Data Targets (MDT), which store the namespace metadata such as filenames, directories and access permissions; One or more Object Storage Servers (OSS) connected to one or more Object Storage Targets (OST) which store the actual files; and clients that access the data over the network using POSIX filesystem mounts. The network is typically either Ethernet or Infiniband.

The Slurm Workload Manager [6], is a free and open-source job scheduler, used by many of the world's supercomputers and computer clusters. At Queen Mary, the batch system is fronted by a Cream Compute Element (Cream-CE) [7] which accepts job submissions from the Grid, interfacing with the local Slurm batch system.

*e-mail: d.traynor@qmul.ac.uk
**e-mail: t.froy@qmul.ac.uk

Hadoop [8] and Spark [9] are open-source software frameworks for distributed storage and processing. Normally Hadoop and Spark require clusters of dedicated hardware utilising the HDFS file system and Yarn batch system. The aim of this paper is to demonstrate that using the Magpie [10] software package together with an existing Lustre file system and Slurm batch system, without any additional hardware deployment, Hadoop and Spark workloads can run using standard Grid commands.

2 The Problem with Hadoop/Spark

Hadoop is a data distribution and processing framework made up of several tools (Yarn, HDFS, MapReduce) and an ecosystem of related projects (Pig/Hive/Storm/Spark). The Hadoop ecosystem is hugely popular in the commercial world and is under active development. Hadoop clusters are constructed with commodity servers (nodes) with local hard drives to run the HDFS file system. A Hadoop cluster will have at least one "namenode" that controls the resource allocation and many "datanodes" that do the actual data storage and processing. The Hadoop system is designed to tolerate failure of hard drives or nodes by having multiple copies of a data file across different nodes.

There has been an increasing interest in the use of Hadoop/Spark in HEP recently. However, usage in HEP still remains small. There are a number of obstacles that prevent and hinder users from using Hadoop/Spark at scale,

- Most computing resources are setup as High Performance (HPC) or High Throughput (HTC) clusters. These do not natively run Hadoop/Spark workloads.
- Creating a dedicated Hadoop cluster will take resources, hardware and manpower, away from existing HEP computing needs, there is yet not enough demand to justify this for most providers of computing resources in HEP.
- Many HEP users do not have access to the finance to use a commercial cloud solution which could provide Hadoop/Spark services.
- Private cloud and containerisation may provide a future solution to resource provision. However, their availability to the HEP community still remains low.

Many providers of computing resources to HEP now also have to serve other communities beyond HEP which do make greater use of Hadoop/Spark. It would be beneficial to serve these communities without out permanently splitting resources.

3 A Solution With Magpie

The Magpie project provides a way to integrate Hadoop and Spark into existing HPC and HTC infrastructures. It supports running over any generic network filesystem (including Lustre/NFS/HDFS) and several scheduler/resource manager (such as Slurm/LFS). Magpie is primarily developed by Al Chu at LLNL (https://github.com/LLNL/magpie) and is being actively maintained and developed. It also supports a large part of the Hadoop ecosystem (figure 1) and a very wide range of software versions.

Magpie accomplishes this task by instantiating a temporary Hadoop/Spark cluster within the context of a batch job, rather than on a persistent, dedicated cluster. The batch job will consume job slots to create a Hadoop cluster over several nodes. If required a transient HDFS file system can be created using local scratch space on the node or a network file system. Data can then be copied in to the cluster, processed, and a copy of the results can be made to persistent storage. At the end of the job, all traces of the cluster will be destroyed. At which point, the jobs slots are available for other types of workloads.

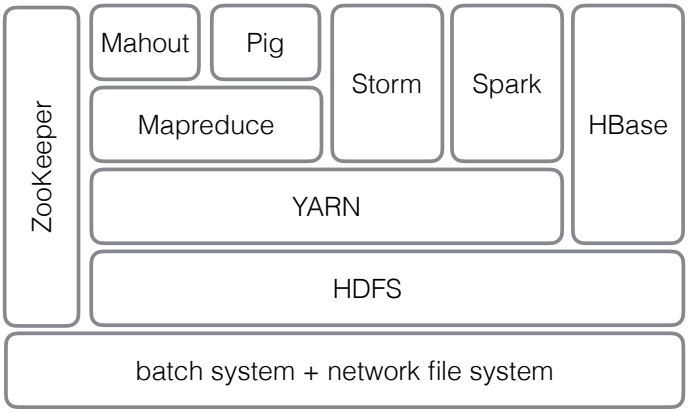

Figure 1. Magpie can support a large part of the Hadoop ecosystem on top of a traditional batch and file system

4 Setup and Use

The following will specify a typical recipe used to run a Hadoop job on the Queen Mary cluster via the Grid using a Cream-CE.

1. Hadoop and Magpie software must be available on every node in a cluster. These directories will not be written to, so they can be provided via readonly file system such as CVMFS.

2. A user submits a standard Grid job to a batch queue via a Cream-CE. The job submission will need to specify the number of job slots (CpuNumber) and required hosts (HostNumber), the magpie/hadoop configuration script and the actual Hadoop workload script to run.

3. A user sets magpie/hadoop configuration at job submission in the magpie_hadoop_script.sh which define the parameters of the Hadoop cluster. This script runs at job start up. A typical script is shown below. Detailed documentation can be found on the Magpie web site. However, important configuration variables are: MAGPIE_SCRIPTS_HOME and HADOOP_HOME which specify the directory where the Magpie and Hadoop software is stored in step one ; MAGPIE_LOCAL_DIR and HADOOP_LOCAL_DIR which specify the directory where the job specific Magpie and Hadoop configuration file are kept; HADOOP_FILESYSTEM_MODE, HADOOP_HDFS_REPLICATION and HADOOP_HDFS_PATH_CLEAR, HADOOP_PER_JOB_HDFS_PATH set several storage settings; and HADOOP_HDFSOVERLUSTRE_PATH, HADOOP_HDFSOVERLUSTRE_REMOVE_LOCKS define specific Lustre configuration settings. The lines beginning with srun are responsible for creating the Hadoop cluster, running the Hadoop job and killing the cluster at the end of the job.

```
export JAVA_HOME="/usr/lib/jvm/jre/"
export MAGPIE_SUBMISSION_TYPE="sbatchsrun"
export MAGPIE_SCRIPTS_HOME="/opt/shared/magpie"
export MAGPIE_LOCAL_DIR="${LOCAL_HOME}/magpie"
export MAGPIE_JOB_TYPE="hadoop"
export HADOOP_SETUP=yes
export HADOOP_SETUP_TYPE="MR2"
export HADOOP_VERSION="2.8.2"
export HADOOP_HOME="/opt/shared/hadoop-${HADOOP_VERSION}"
export HADOOP_LOCAL_DIR="${LOCAL_HOME}/hadoop"
export HADOOP_MODE="script"
export HADOOP_FILESYSTEM_MODE="hdfsoverlustre"
export HADOOP_HDFS_REPLICATION=1
export HADOOP_HDFS_PATH_CLEAR="yes"
export HADOOP_PER_JOB_HDFS_PATH="yes"
export HADOOP_HDFSOVERLUSTRE_PATH="/mnt/lustre/hadoop/"
export HADOOP_HDFSOVERLUSTRE_REMOVE_LOCKS=yes
export HADOOP_SCRIPT_PATH="${LOCAL_HOME}/my-job-script.sh"
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-check-inputs
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-setup-core
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-setup-projects
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-setup-post
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-pre-run
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-run
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-cleanup
srun --no-kill -W 0 $MAGPIE_SCRIPTS_HOME/magpie-post-run
```

4. The job is started by the local batch system. It then launches daemons on all reserved nodes creating a Hadoop cluster as requested. Then the user script is run. Data can then be copied into HDFS, if needed, and processed. The final results are then copied back to the user directly via the CE. An example user script which runs a typical Hadoop benchmark process is shown below.

```
yarn jar /usr/local/hadoop/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.8.2.jar teragen
-Dmapreduce.jobs.maps=1000 100000000 random-data
yarn jar /usr/local/hadoop/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.8.2.jar terasort random-data sorted-data
yarn jar /usr/local/hadoop/share/hadoop/mapreduce/
hadoop-mapreduce-examples-2.8.2.jar teravalidate sorted-data report
```

5 Optimisations

An important choice when creating a HDFS file system is whether to create the file system on local scratch disks or on a network file system. A locally attached hard drive should have high bandwidth to jobs running on the same node. A network file system can aggregate the performance of 100s of hard disks to overcome the penalty of remote, network accessed, storage. Two sets of benchmarks were used to investigate the performance of local vs remote storage options. The benchmarks were carried out with three dedicated nodes each with 40 job slots and three 1TB disks each. Nodes are connected to the network with 10Gb/s Ethernet. Results with Lustre uses the existing production Lustre storage system.

The first test uses IOzone [11] in cluster mode, using multiple threads on each node either writing to different directories on Lustre or to different disks on the local node. The Lustre performance is consistently better than that for local disks (figure 2).

Hadoop DFSIO is a standard benchmark program for measuring IO performance that comes with Hadoop. Setting up a Hadoop cluster with one namenode and two datanodes with each data node having three 1TB hard drives per node. The HDFS over Lustre performance is consistently better than that for HDFS using local disks (figure 3).

In both cases running HDFS over Lustre is quicker than using local disk. While increasing the number of nodes, and hence local disks, will improve the performance of HDFS using

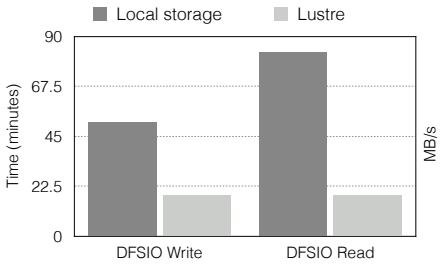

Figure 2. IOzone benchmark results for four different data throughput tests (read, reread, write and rewrite). Results are in MB/s, Higher throughput indicates better performance. Lustre (light gray) is consistently better than local disks (dark gray).

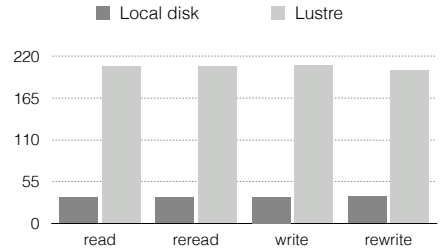

Figure 3. DFSIO benchmark results for reading and writing to a Hadoop cluster. Results are in minutes to complete the benchmark. Shorter time indicates better performance. Lustre (light gray) is consistently better than local disks (dark gray).

scratch, Lustre will provide a consistently high performance independent of the size of the Hadoop cluster.

6 Limitations

A number of limitations with the present service have been identified that limit a full scale production service at Queen Mary.

- A User submitting a job must know local details about the site they are using (e.g. mount points, software paths, software versions) so they can provide the magpie_hadoop_script.sh at job submission.
- Requesting whole nodes only works at present, requesting less than a whole node ends up creating one datanode per core on each node causing a significant performance impact.
- The service only provides a transient HDFS file system, as a result, data has to copied in each time a Hadoop job starts, reducing job efficiency.

- It was hoped that the Lustre plugin for Hadoop could have been used to replace the need to create a HDFS file system for each job (https://github.com/whamcloud/lustreconnector-for-hadoop). However, it was found not to work with the version of Lustre used at Queen Mary (2.8).
- The Hadoop ecosystem is hugely popular in the commercial world. There is still only limited use in HEP. However, rapid ongoing developments, e.g. ROOT Spark plugin (https://github.com/diana-hep/spark-root), may promote demand in the future.
- While in principle running Hadoop/Spark jobs on Grid enabled HPC/HTC clusters works, the service is some distance away from a production service. In order to develop a production service there is a need for a clear use case to support, test and tune the service.

7 Conclusions

A demonstration has been made of how to run Hadoop jobs on a HPC/HTC cluster, as typically used in HEP computing, using a Cream-CE for job submission. The service could be expanded to include more software available in the Hadoop ecosystem that is supported by the Magpie project. Such a service may help bridge the gap in available resources for Hadoop/Spark workloads until demand makes building dedicated clusters feasible. A number of limitations with the present configuration were identified but these should not prevent creating a production Hadoop/Spark service if demand develops.

References

[1] Scalable Petascale Storage for HEP using Lustre: C.J. Walker D.P. Traynor and A.J. Martin. *Journal of Physics: Conference Series* **396** 042063 (2012).

[2] Optimising network transfers to and from Queen Mary University of London, a large WLCG tier-2 grid site: C J Walker, D P Traynor, D T Rand, T S Froy and S L Lloyd. *Journal of Physics: Conference Series* **513** 062048 (2014).

[3] Upgrading and Expanding Lustre Storage for use with the WLCG: D P Traynor, T S Froy, C J Walker. *Journal of Physics: Conference Series* **898**, 062004 (2017).

[4] The ATLAS Experiment at the CERN Large Hadron Collider: ATLAS Collaboration (Aad, G. et al.) *JINST* **3** S08003 (2008).

[5] Lustre: "Lustre" [Software], version 2.8, 2018. Available from http://lustre.org/ [accessed 2016-02-15].

[6] Slurm: Simple Linux Utility for Resource Management: "Slurm" [Software], version 17.02.9, 2017. Available from https://slurm.schedmd.com/ [accessed 2017-11-05]; A. Yoo, M. Jette, and M. Grondona, Job Scheduling Strategies for Parallel Processing. *Lecture Notes in Computer Science* **2862**, 44 (2003).

[7] Design and Implementation of the gLite CREAM Job Management Service: C. Aiftimiei et al *Future Generation Computer Systems* **26** 654-667 (2010) doi:10.1016/j.future.2009.12.006.

[8] Apache Hadoop: "Hadoop" [Software], version 2.8.2, 2017. Available from http://hadoop.apache.org/ [accessed 2018-02].

[9] Apache Spark: "Spark" [Software], version 2.3.0, 2018. Available from https://spark.apache.org/ [accessed 2018-02].

[10] Magpie: Al Chu [Software], version 1.83, 2018. Available from https://github.com/LLNL/magpie [accessed 2018-02].

[11] IOzone Project: "IOzone" [Software], version 3.30, 2015. Available from http://www.iozone.org/ [accessed 2015-03-27].