Fermilab FERMILAB-CONF-07-102-E

Upgrade of the CDF Run II Data Logger

I.Bizjak*, F.Chlebana†, G.M.Guglielmo†, T.Masubuchi‡, K.McFarland§, W.Sakumoto§, R.E.Sarkis§, M.Shimojima‡, F.D.Snider†, G.Yu§, D.Zhang†

*University College London, Gower Street, London, WC1E 6BT, UK
†Fermilab, P.O.Box 500, Batavia, Illinois 60510, USA
‡Universiry of Tsukuba, Tsukuba, Ibaraki 305-8571, Japan
§Universiry of Rochester, Rochester, New York 14627, USA

***Abstract*— The Consumer-Server/Logger (CSL) is the final component in the CDF Data Acquisition chain before data is archived to tape. The CSL buffers data in separate data streams, records file meta information into a database and sends a fraction of events to online processors for real time monitoring of data quality. Recently, the CSL was upgraded in order to increase the logging capacity to 80 MB/s. The upgrade consists of commodity servers running Linux. A "Receiver node" distributes data via Gigabit Ethernet to eight parallel "Logger nodes" connected to external disk arrays via a Fibre Channel network. A redundant design and the availability of inexpensive large capacity disk arrays provides a highly available system that is scalable and easy to maintain. We present a description of the CSL upgrade and discuss the experience gained through commissioning to operations.**

I. Introduction

The CDF data acquisition (DAQ) and trigger selects interesting physics events from a high rate of background. The Tevatron beam spacing is 396 ns with a crossing rate of 25 MHz. This high crossing rate is reduced in three stages by a pipelined trigger to about 100 Hz of events. The event size depends on detector occupancy and is typically about 200 KBytes. The Consumer Server/Logger (CSL) is the last stage in the data acquisition system coordinating data archival as well distributing a fraction of events to the real time data quality monitors.

The original CSL was designed to log data at a rate of about 75 Hz and was based on specialized hardware that is becoming increasingly difficult to support. A description of the CSL can be found in Ref [1]. Improvements to the Tevatron has lead to stores with higher luminosity and the DAQ needs to be able to handle the associated higher data logging rates. An important feature of the CDF trigger is the ability to relax trigger prescales as the luminosity decreases over the length of a store making full use of the available bandwidth. This puts a greater demand on down stream data handling which has to run at higher rates for longer times. It was necessary to upgrade the CSL in order to handle the higher sustained logging rates and to ensure that the system can be supported for the life of the experiment. In addition to being able to support higher logging rates, the upgraded system is more fault tolerant and long term support needs are reduced by adopting common down stream logging solutions for both collider experiments at the Tevatron. The upgraded CSL has a modular, highly redundant design that uses off-the-shelf hardware and allows for future upgrades.

II. Design Requirements

The upgraded CSL needs to be able to log data at a sustained rates of 50 to 60 MB/s. In order to allow sufficient headroom, a design target for the CSL upgrade was set to 80 MB/s. The original CSL was centered around a Silicon Graphics server with no direct replacement. The use of specialized hardware and a operating system that is not widely used at the lab made the support of the old CSL increasingly difficult and reliant on a costly hardware support contract. The upgraded CSL uses off-the-shelf servers running Linux for which there is a greater pool of expertise for both the hardware and operating system. Fully configured spares are available to replace any failed server. The designed placed emphasis on making the system both redundant and fault tolerant in order to reduce long term support requirements. Since the upgrade is for a running experiment, it was necessary to retain a similar interface to external components and to develop and test the new system without interfering with the data taking.

III. Architecture

A. Hardware

The design incorporates an over-capacity of bandwidth, disk buffering and processing power to allow any failed component to be bypassed without impacting data logging. An overview of the upgraded CSL is shown in Fig. 1. Data from the Level 3 filter farm is sent to the Receiver node through a Gigabit ethernet switch. The Receiver node will forward data to one of eight Logger nodes depending on which triggers the event passes. Sorting data based on the trigger information simplifies down stream data handling. In order to increase the network bandwidth multiple Gigabit ethernet interfaces are "bonded" together so they appear as a single interface with the load balanced between the interfaces.

The Logger nodes write the data to external buffer disks distributed across two storage arrays. The Nexsan SATABeast storage arrays were selected because they are cost effective, have a high IO throughput, are reliable and are well supported in Linux. Connection between the host servers and the disk arrays is via a Fibre Channel (FC) network.

A high logging rate is achieved by distributing the file writing task to the logger nodes. Events which are typically

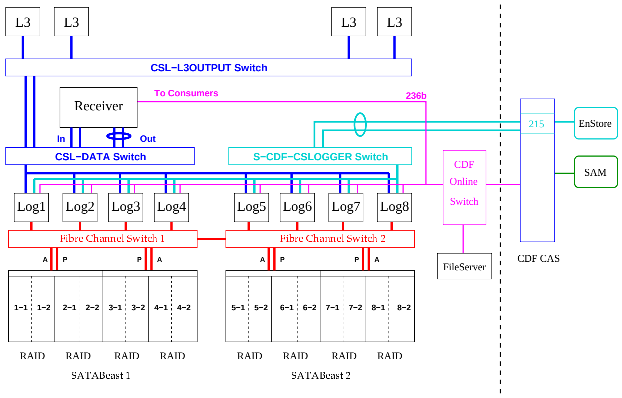

Fig. 1. Overview of the CSL Upgrade

200 KBytes are accumulated in files of one GigaByte. Meta information is written in a database and is later used by the data handling system to retrieve the data files. In order to achieve optimum tape streaming performance, multiple files are accumulated on disk before initiating the tape writing job. The data is transported to a separate facility (Feynman Computing Center, FCC) via a private Gigabit network where it is finally archived to tape.

A key design requirement for the upgraded CSL is that the system has a high reliability and availability and requires minimum support. The Receiver and Logger nodes utilize the same hardware and each can be reconfigured to take on the role of the other. Fully configured spares are available for both the Receiver and Logger nodes. The spares, which can also be used for code development and testing, can quickly take on the role of a failed node by stopping the CSL and updating configuration files. Two separate external disk arrays (SATABeast1 and SATABeast2) are connected to the logger nodes via a FC network (Fibre Channel Switch 1 and 2). Each logger node has access to two separate buffer areas, one on each of the disk arrays. In this configuration four logger nodes write to separate file systems on one SATABeast while the remaining four would write to the second SATABeast. This configuration allows the output from all eight logger nodes to be directed to one disk array in case one of the disk arrays is not available. Two Fibre channel switches are configured with redundant paths to the disk buffers such that if one switch fails, all traffic is directed through the working switch.

The total disk capacity of the upgraded CSL is 24 TeraBytes providing three days of data buffering while running at the full design specification of 80 MB/s. This deep buffer allows for an extended loss of the network connection between B0 and FCC and provides sufficient time to respond to problems or service equipment. When data logging is resumed, the bandwidth over-capacity allows us to ”catch up” and empty the buffer disks at the same time as data is being collected. More details about testing and performance are included in a later section.

B. Software

The first-generation of the CSL code was pseudo-monolithic in design and written in C and C++. The code consisted of several co-operating processes. Event reception, sorting, monitoring and file writing were all performed on one server node. The new CSL code was redesigned in order to map onto the new hardware architecture. The disk logging ”csl_logger” process was modified to have a new networking abstraction layer. Rather than saving events to a local disk buffer on the Receiver node, the logger process now acts as a network-level event router for sorting and sending events to the correct remote Logger node which in turn runs a client ”csl_logger_client” process that receives events and writes them to the local disk buffer on the Logger node.

In the CSL set of cooperating processes, the ”csl_receiver” process receives events from the Level 3 farm and places them in a large Sys V shared memory segment queue. At this point, other CSL processes are able to inspect the events in the queue for such purposes as as monitoring, while the csl_logger process is responsible for removing events in order to send them over the network to selected Logger node based on event triggers. These changes result in a modular, redundant and a highly reliable software architecture that can allow for an arbitrary number of logger nodes.

The event handling by the csl_logger process has been significantly modified to make use of a multi-threaded producer-consumer design. Using mutex synchronization, each thread uses a large (but tunable) memory ring buffer to ensure that the network bandwidth is optimally utilized. As events are received and placed into the shared memory queue, the new csl_logger takes these events, encapsulates and then places them in an outgoing ring buffer.

On the remote Logger node, the csl_logger_client uses the same, basic producer-consumer design of the Receiver node’s csl_logger. The sole purpose of this daemon is to receive the encapsulated events sent from the csl_logger process on the Receiver node and intelligently aggregate these events into large data files whose size is determined by a configuration file parameter. All event sorting and cataloging is done on the Logger node side.

The new CSL networking architecture incorporates a network fail-over mechanism that is designed to maintain the integrity of the event streams. In the event of a failed Logger node, the receiver will momentarily pause the outgoing event distribution and select a spare Logger node from a user-defined pool of node hosts. If a spare node is found, a handshake protocol re-establishes the stream and it continues the event streaming starting at a well defined boundary. If no spare logger can be found, one of the existing loggers is chosen to host the re-established stream.

IV. Archiving Data to Tape

The primary functions of the tape writing sub-system are to copy data files from the disk buffers of each Logger node to tape in the Enstore Mass Storage System[2], record metadata for each file in the SAM[3] data-handling system database and delete the data files from disk once successful completion of all operations has been verified. In order to have the capacity to drain a potential backlog of buffered data and to cover an overlap in the events contained in the eight data streams (up to 20%) while simultaneously logging data at full rate, the tape writing rate specification is 160 MB/s for the entire system. To deal with imbalances in the data volume into each data stream, the specification for each Logger node is 30 MB/s.

For the CSL upgrade, CDF has adapted the tape staging software already in use by the DØ experiment at Fermilab. This system, which is based upon the SAM data-handling system used by both CDF and DØ, has a proven record of reliable operation at DØ, broadens the base of support for the tape stagers at both experiments and provides a more sustainable support model through the end of the experiment.

Independent tape staging processes run on each of the Logger nodes. The processes monitor a set of input directories into which data to be archived is copied. Each stager instance can be configured to transfer data stored in the input buffer associated with any other Logger node, thereby allowing data to be cleared in the event that a Logger node fails before logging all buffered data. Each staging process can also log data belonging to any of the data streams, so that the system will seamlessly continue logging data should a hardware failure require sending two data streams to a single Logger node.

The tape logging procedure requires interactions with several external non-dedicated systems, such as Enstore, the SAM data-handling system and a database. Requests to these services are queued in order to control the load imposed by data logging activities. Responses are returned to shared memory queues so that the stager processes can manage the workload in a stable and predictable manner. A series of timeouts and re-tries allow the system to weather routine interruptions and downtime in these services. Files that cannot be transferred due to repeated failures are preserved in a special directory. Monitoring programs alert operators to the presence of these files, who can then take appropriate actions. Under typical conditions, an operator intervention of this type occurs once every few months.

In order to optimize tape drive utilization, the system initiates tape writes for a data stream only after accumulating 50 files, or waiting two hours since the previous transfer. (Both values are tunable.) The current system uses STK-9940B tape drives. For 1 GB files, we observe an average transfer rate of 20–25 MB/s during multiple sequential file transfers. The number of simultaneous tape drives in use for a single data stream (currently three) and the maximum number of transfers from a single Logger node are independently configurable to further optimize system performance. During full scale tests using all eight Logger nodes, we obtained a sustained rate to tape of 180 MB/s across the entire system. The system will migrate to LTO-3 commodity tape drives in the near future, a change which will further increase the overall tape logging rate capability.

Minimizing the possibility of data loss due to a malfunction of any type anywhere in the system was a paramount design consideration for the stager software. To achieve this goal, each operation in the tape staging procedure is verified before moving to the next. For each data file to be archived, the tape staging processes also maintain a history file that includes all completed and verified operations. This file is stored on a separate disk from that used for the data to be written to tape. In the event that the tape writing processes are interrupted for any reason, including system crashes, hardware failures or power outages, the state of the system can be restored and re-started from the last completed operation based upon the content of the history files.

To provide redundancy for the history files, each tape stager uses a sequence of directories on the corresponding buffer disks to specify the state of files in the tape writing procedure. Each directory contains the files that are currently undergoing a specific set of operations. As these are completed and verified, files are moved to the next directory in the sequence. Should the history files become corrupted or lost for any reason, they can be re-constructed based upon the content of the state directories. During the standard automated system recovery, the content of these directories is compared to the history files to ensure proper accounting for all files.

In almost six years of combined operation at CDF and DØ and approaching 2 M files stored, no data has been lost due to a malfunction of the tape staging software.

V. Testing and Performance

During the development of the CSL upgrade, CDF was actively collecting data and it was essential that the testing and commissioning of the upgrade would not interfere with data taking. As part of the hardware evaluation, the components were extensively tested to ensure they met the requirements and were robust. Extended network bandwidth tests were performed in which the Receiver node simultaneously sent data to the eight Logger nodes. A sustained bandwidth of 224 MB/s was achieved by using two Gigabit network interfaces bonded in software to appear as one logical unit. Using two interfaces also provides network redundancy. In case one interface failed the designed rate of 80 MB/s could still be achieved with one interface. The disk performance was tested by writing to the disk at the same time as a second process read data from the disk. In order to determine the total disk IO capacity the test was run on all eight Logger nodes simultaneously. Since the Logger nodes were connected to the external disk arrays through the fibre channel switch, this procedure also tests the reliability and performance of the SAN fabric and the SATABeasts. The total write rate achieved was 440 MB/s while the total read rate was 580 MB/s.

In order to test the system while operating under more realistic conditions, previously recorded data was sent to the CSL by a software process. The software process replaced the functionality of the Level 3 filter farm while everything down stream would behave as it would during real data taking. While running in this test mode it was possible to optimize the performance extensively and test the failure and recovery of the system's components to ensure that they work as designed. During this test a sustained throughput of about 100 MB/s was measured while sending data all the way to tape. The rate was limited by the software process that was sending the Receiver data.

VI. Monitoring

The CSL software is designed to automatically address a large set of problems, thereby minimizing the need for user intervention. An overview of the flow of monitoring information is shown in Fig 2. CSL software specific information is gathered from Logger and Receiver nodes in order to help understand unexpected failure modes. More general server statistics is also collected using Ganglia [4]. Information on the overall performance is displayed on a dedicated Java based monitor (CSLMon). A summary with history plots can be viewed from a web browser making this information remotely available. Critical information requiring immediate attention is directed to the Error Handler. The Error Handler is an "expert system" which gathers and logs error messages from all DAQ components. Specific recovery instructions can be displayed to the shift crew or, for the more common errors, an automated recovery procedure initiated. The majority of error conditions can be recovered without requiring any human intervention.

Information from the Logger nodes is sent to the Receiver node, where it is combined with additional information available only in the Receiver node. The status information is then packaged into a well defined message structure which is periodically sent out. Message passing between components of the CDF DAQ uses a publish/subscribe protocol. The component from which the message originates is encoded in the message subject. Clients subscribed to the subject can then receive the message. Diagnostic and status messages are received by CSLMon. Logger nodes write out additional status information to a file which is also available to CSLMon. CSLMon can determine if an error condition is present, based on all available information, and if so will send an "Alarm" to the Error Handler. Error conditions and the corresponding behavior is defined in configuration files enabling easy addition, removal and modification of error condition information. As our experience with the system increases additional error conditions can easily be added.

The history of gathered status information is available as graphs of trends, such as disk usage, network traffic and tape writing rates. RRDTool [5] is used to store the information and to generate graphs which can be viewed from CSLMon or a web browser. Web based monitoring offers snapshots of CSLMon that are updated every minute, a history of recent error conditions, history plots of status information gathered on each node and detailed information on the file-logging status. It provides all the essential information needed for remote diagnostics and inspection of the system. The monitoring reflects the modular nature of the system and can easily accommodate possible future extensions.

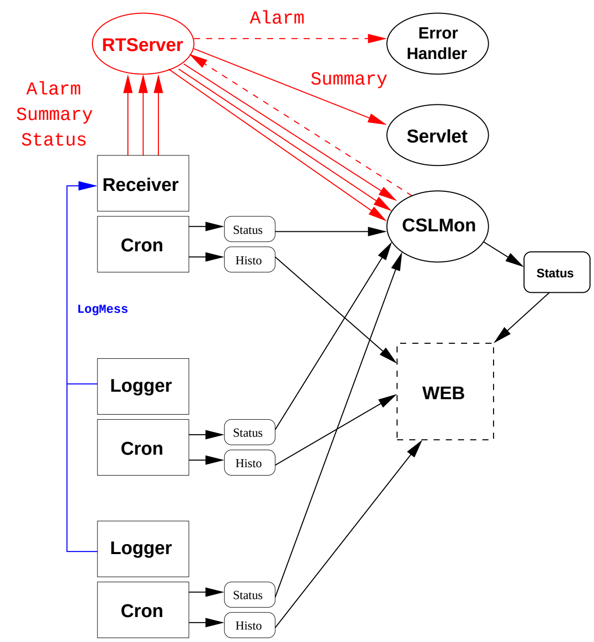

Fig. 2. An overview of the flow of information used to monitor the CSL.

VII. Operational Experience

Once the system is in use it is more difficult to implement changes and a strict testing protocol must be followed. Software versions are controlled using cvs and ups/upd which allows quick switching between test versions and production releases. Test are done at the end of stores when trigger rates are low in order to minimize data loss.

A major concern is the possibility of having files left over on the buffer disk without being recorded in the file handling system. In order to check for "lost files" a record of the file is made when it is first opened for writing. The files in the list are later checked when they are written to tape. Any files older than one day which show up in the file list and are not written to tape are noted on a web based monitoring page. Once a lost file is found, an automated email alert notifies system operators, who can then take appropriate actions. A few exceptions, such as removing test files from disk buffers and moving files that are left over due to abnormal run terminations, are handled by scripts that run periodically.

Scripts have been developed to allow the shift crew to restart the CSL without expert intervention. Restarting the CSL will put the system into a well defined state from which nearly all exceptions can be recovered.

VIII. Conclusion

The Upgraded CSL has been in operation since Nov 2006. The system has been running reliably, requires little ongoing support and has had no hardware related problems to date. The system is very redundant and in most cases the software is able to automatically bypass any failed hardware component. The upgraded CSL exceeds the design specification of being able to log data at a sustained rate of 80 MB/s. The system satisfies the current needs of the experiment and the modular design allows for future expansion of logging capacity.

REFERENCES

[1] M. Shimojima,*et al.*, *Consumer-Server/Logger System for the CDF Experiment* ,11th IEEE NPSS Real Time Conference (RT99), June 14-18, 1999, Santa Fe, New Mexico, published as IEEE Trans. Nucl. Sci. 47 (2000) 236-239.

[2] http://www.fnal.gov/docs/products/enstore/

[3] Sequential data Access via Meta-data, http://d0db.fnal.gov/sam/

[4] M. L. Massie, B. N. Chun, D. E. Culler, *The Ganglia Distributed Monitoring System: Design, Implementation, and Experience.* Parallel Computing, 30 (7), 817-840, 2004.

[5] http://oss.oetiker.ch/rrdtool/