DEVELOPMENT OF J-PARC TIME-SERIES DATA ARCHIVER USING DISTRIBUTED DATABASE SYSTEM

N. Kikuzawa#, A. Yoshii, H. Ikeda, Y. Kato, J-PARC, Tokai-mura, Naka-gun, Ibaraki, Japan

Abstract

J-PARC (Japan Proton Accelerator Research Complex) consists of many equipments. In Linac and 3 GeV rapid cycling synchrotron ring (RCS), date of about 64,000 EPICS records have been collected for control of these equipments. Data volume is about 2 TB in every year, and the stored total data volume is about 10 TB. The data have been being stored by a Relational Database (RDB) system using PostgreSQL since 2006, but it is not enough in availability, performance, and flexibility for increasing data volume. The new database system is required to have high availability, high performance, and high flexibility of storage expansion.

The purpose of our work is to examine the next-generation archive system using Apache Hadoop of a distributed processing framework and Apache HBase of a distributed database. The purpose of this paper is to report the present status of this archive system and future plan.

INTRODUCTION

J-PARC is controlled with a lot of equipment, and we have been archiving time series operation data of about 64,000 EPICS records in Linac and RCS [1] since 2006. PostgreSQL has been used in the present data archiving system, but it has some problems of capacity, extensibility, data migration and so on. In order to deal with the problem, we proposed a next-generation archive system using Apache Hadoop [2] of a distributed processing framework and Apache HBase [3] of a distributed database.

The Hadoop has become a widely used open source cloud computing framework for large scale data processing. The Hadoop core is a file system, called HDFS (Hadoop Distributed File System) and the HBase is working on it. The HBase is a distributed, scalable Big Data store on clusters built with commodity hardware. The Hadoop and the HBase are scale-out type architecture, and we can extend a system dynamically by adding nodes if needed. We can therefore achieve efficient investment. Moreover, since the Hadoop and the HBase are designed based on the assumption of frequent node failures, they have the tolerance to them and are easy to restore. The HBase is a type of "NoSQL" database and is optimized for data storing and acquisition by restricting modification of data, transaction, and table combination, as compared with an RDB. The HBase is suitable for the use of write-once read-many.

We consider that this database is the best for the time series data archiving system. Our purpose is to evaluate the performances of the Hadoop and HBase system in comparison with those of the present PostgreSQL system implementation.

#kikuzawa.nobuhiro@jaea.go.jp

DATABASE SYSTEM CONFIGURATION

Hardware Configuration

The Hadoop and HBase cluster has two types of machines: masters (NameNode, JobTracker, HBaseMaster, and Zookeeper) and slaves (DataNodes, TaskTrackers, and RegionServers). About 50 TB of the HDFS is built using nine slave nodes with commodity servers, and the HBase is working on it. Each slave node contains four 2 TB local hard disk storages constituted in

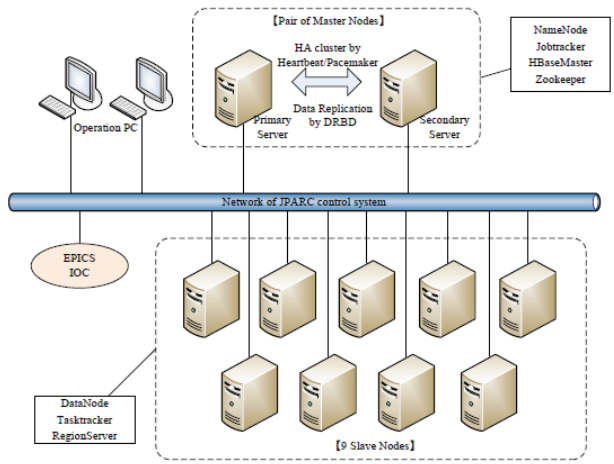

Figure 1: The test system of the archive system.

Table 1: Spec of Hadoop and HBase System

Role	Composition
Master Node (Primary)	DELL PowerEdge R610 CPU: Intel Xeon E5620 (4Core 2.4GHz) MEM: 24GB (8GB x 3, 1333MHz, DDR3) HDD: 600GB SAS 10 krpm x 4 (RAID10)
Master Node (Secondary)	DELL PowerEdge R200 CPU: Intel Xeon X3210 (4Core 2.13GHz) MEM: 8GB (2GB x 4, DDR2) HDD: 160GB SATA 7200rpm x 1
Slave Node	DELL PowerEdge R410 CPU: Intel Xeon E5620 (4Core 2.4GHz) MEM: 24GB (8GB x3, 1333MHz, DDR3) HDD: 2TB SAS 7200rpm x 4 (RAID5)
Software	OS: CentOS6.3 (Japanese) Java: JDK 1.6.0_45 Hadoop: 1.0.4 HBase: 0.94.5 Heartbeat: 3.0.7 Pacemaker: 1.0.12 DRBD: 8.4.1 Ganglia: 3.4.0

ISBN 978-3-95450-139-7

RAID 5. Each server is interconnected using Gigabit Ethernet (GbE). Figure 1 shows the system configuration. According to our experience, HBase requires much memory; although the slave nodes started operation with 12 GB memory, which had to be extended to 24 GB. Table 1 shows the spec of the Hadoop and HBase system.

High Availability Cluster

It is mentioned that a master node is SPOF (single point of failure) as a problem of the Hadoop and the HBase. So, we make two servers into HA (High Availability) cluster structure using Heartbeat and Pacemaker [4], and raise system availability. Moreover, important metadata for data management is protected by the data replication between two servers using DRBD (Distributed Replicated Block Device). The Heartbeat needs to be combined with a Cluster Resource Manager (CRM) to start/stop services for that cluster. The Heartbeat controls the master process of the Hadoop and the HBase, and the master of DRBD. The Pacemaker is the preferred CRM for the Heartbeat.

We set up a Virtual IP (VIP) address using the Heartbeat and the Pacemaker, and then associate it with the active master node. If the active master node has crashed, the Heartbeat and the Pacemaker will detect it and assign the VIP address to the standby master node, and then start the NameNode there. Alive monitoring of each service is performed for every 1 minute, and failover to the secondary server is performed when a timeout occur of the services. When it is judged that the Heartbeat itself fell into the down state, the failed node is removed from the cluster using STONITH (Shoot The Other Node In The Head).

Since it is necessary to perform starting of the HBase system service after the initialization processing of the Hadoop system service start-up is completed, the time lag of 180 seconds has been prepared for starting of the Hadoop system service. It will take about 5 minutes for failover and failback in our system.

TABLE STRUCTURE AND PERFORMANCE TEST

Table Structure

In order to register many data with the present PostgreSQL database system with low write processing load, the large number of the acquired data is divided into some groups and gathered into a row for each group. In this case, the system load at the data registration is low, on the other hand, data retrieval speed become slow, because it is necessary to extract required data from the combined data.

The HBase is a type of column-oriented database, a simple structure of key-value is suitable. It is possible to have huge size tables as compared with the conventional database system. Then schema design in the column-oriented database is very different from schema design in an RDBMS (Relational Database Management System).

We have so far carried out examinations about table structure [5]. In the HBase, since data is stored in ascending order of row (primary key), if all row keys start with the same value and monotonically increasing like timestamp, they will be written only in the same region and thus the same server [6], which results in slow writing speed. In order to avoid this problem, the EPICS record name of the data is attached to the head of the primary key.

Figure 2 shows the image of the data table structure. Since the data are located in a line by the time series for every EPICS record, this structure can achieve efficient retrieving the time series data. Moreover, process of data writing is naturally distributed because the postscript place of new data is distributed (the node holding the data of a postscript place is distributed).

About 6,500 polling type data in a cycle of 1-60 seconds and about 13,000 the event type data in Linac are collected, and it can be archiving stably.

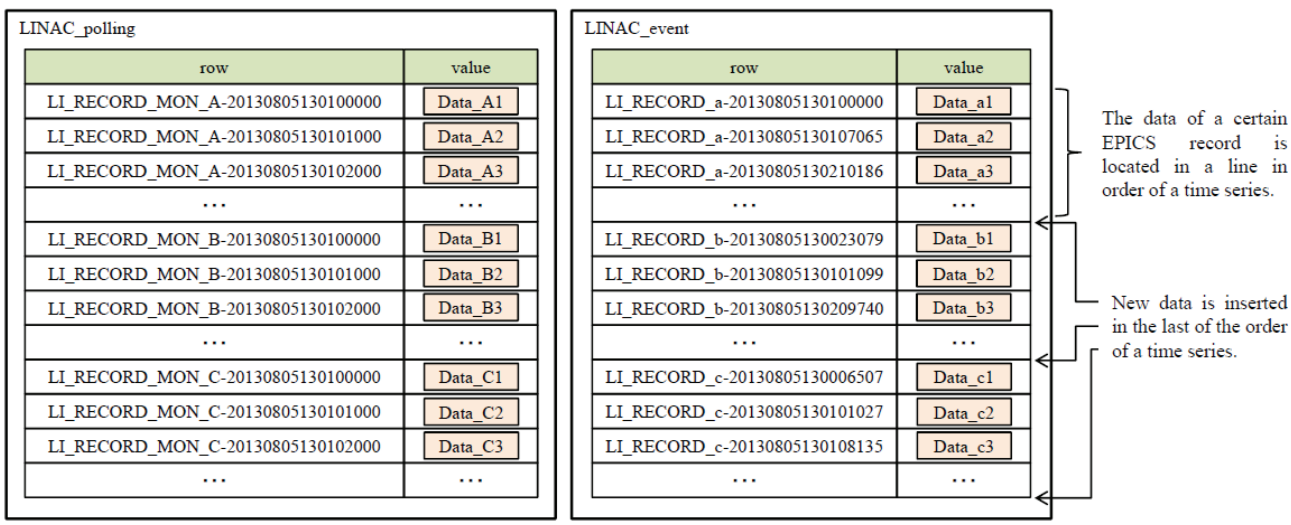

Figure 2: Image of data table structure.

ISBN 978-3-95450-139-7

Speed of Data Retrieval

Several benchmarks have been carried out to evaluate the performance. The past data was copied to the HBase and retrieval speed of the HBase system was compared with that of the present system. Table 2 shows spec of the present system. The present system has two storages; one is the high spec storage of StorNext [7] for data input, another is lower spec storage for data retrieving.

Table 2: Spec of PostgreSQL System

Role	Composition
for Data input	DELL PowerEdge R200 CPU: Intel Xeon X3210 (4Core 2.13GHz) MEM: 8GB (2GB x 4, DDR2) Internal HDD: 160GB SATA HDD for Storing: 10TB (External file server storage is shared using "StorNext" in 10Gb network.)
for Data retrieving	DELL PowerEdge 860 CPU: Intel Xeon 3060 (2Core 2.4GHz) MEM: 4GB (2GB x 2, DDR2) Internal HDD: 250GB SATA HDD for Storing: 4TB (1TB x 4, Using external disk of USB connection)

The data copied to the HBase was Linac data for one year, and volume of the data was about 1 TB. Because of the high speed processing and reasonable compression rate, we chose Snappy [8] as compression/decompression library.

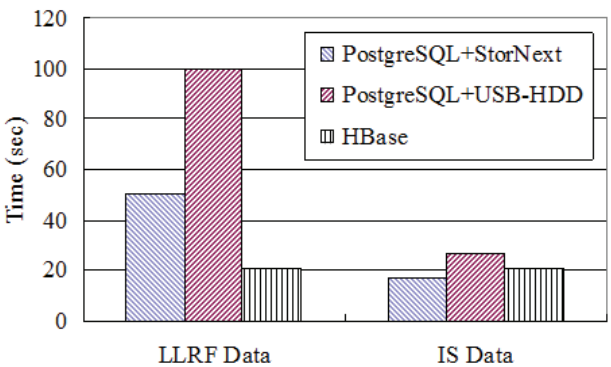

Figure 3: Data retrieval performance comparison.

We carried out a benchmark test on the PostgreSQL and the HBase system. The time taken to retrieve the data of 5 days of Low Level RF (LLRF) and Ion Source (IS) are shown in Figure 3. About retrieving of the IS data, since the number of the EPICS records in the IS group is 45 and there is a small number of data in a row, there is almost no difference in retrieving time. On the other hand, retrieving time has a big difference about LLRF data because the number of the EPICS records in the LLRF group is 1846, then extracting the required data took much time. About the difference between the PostgreSQL systems, it originates from the difference of their storage performances. Retrieving time of the HBase system is shortened to about 1/5 as compared with the PostgreSQL system.

Future Plan

We will have to develop applications corresponding to the HBase for the system operation. Since it is necessary to use Java API fundamentally to cooperate with the HBase, it is thought that much time is required for porting the applications developed with other development languages. To reduce the software transplantation work, we are also considering accessing through gateway API like Thrift, REST, and Avro, and using of JNI (Java Native Interface).

CONCLUSION

We proposed the next-generation archive system using the Apache Hadoop of the distributed processing framework and the Apache HBase of distributed database. The demonstration system of the archive system using the Hadoop and the HBase was built with commodity servers. About 50 TB of the HDFS is built using nine slave nodes, and HBase is working on it. About 6,500 polling type data in a cycle of 1-60 seconds and about 13,000 the event type data in Linac has been collected stably under a test operation for 3 months. Moreover, as for the data retrieval of large group data, response time is shortened to about 1/5 as compared with the present PostgreSQL system. From the benchmark test results, we conclude that the new archiving system using the Hadoop and the HBase is suitable for the time series data archiving.

REFERENCES

[1] S. Fukuta, et al., "Development Status of Database for J-PARC RCS Control System (1)", Proceedings of the 4th Annual Meeting of Particle Accelerator Society of Japan, August 2007. [in Japanese]
[2] http://hadoop.apache.org/
[3] http://hbase.apache.org/
[4] http://www.linux-ha.org/wiki/Heartbeat
[5] A. Yoshii et al., "J-PARC operation data archiving using Hadoop and HBase" Proceedings of the 9th Annual Meeting of Particle Accelerator Society of Japan. [in Japanese]
[6] Apache HBase Reference Guide (http://hbase.apache.org/book.html)
[7] http://cn.teldevice.co.jp/product/detail/stornextt
[8] http://code.google.com/p/snappy/

ISBN 978-3-95450-139-7