INTEGRATION OF HBASE WITH HDFS FOR OPTIMIZING THE STORAGE AND PROCESSING EFFICIENCY OF ENCRYPTED PATIENTS HEALTH RECORDS

----------***----------

ABSTRACT-*The advance of computerized innovation throughout the years has prompt information blast "Big Data" which requires large propelled handling, accessing, analyzing and visualizing techniques. As innovation continues in advancing and ascend in different enterprises, such as medicinal services, science, education, gaming, etc. Personal Health Record data contains variety of health related information such as allergy data, family history and some laboratory test results. The PHR data requires a major storage for storing the massive PHR data. The design focuses on handling various file size of the encrypted PHR data on distributed storage and also provide an API for Patient Health Record system (PHR) to upload/ download the encrypted PHR data from a distributed storage. HBase and Hadoop are utilized in this work to store the encrypted PHR data. The proposed system resolves the Namenode memory issues of HDFS by classifying the encrypted PHR documents into small and large files. Hadoop stores the large PHR files and HBase stores the small files.*

***Key Words*: Hadoop, HBase, PHR,**

Storage 1. INTRODUCTION

These days healthcare technology can improve high-quality of lives. Every day lifestyles activity information can be a source for predicting a sickness or preventing one. Personal health record (PHR) [1] is a idea that emerges recently. The PHR owner can store any health related statistics into the PHR storage and the PHR system should make sure that the PHR owner has a complete control over his/her facts [1]. In contrast, the electronic medical record(EMR) stores the patient related data and the EMR belongs to the healthcare facilities even as the PHR stores any health associated statistics and the PHR belongs to the PHR owner [2]. This manner, the user can save and retrieve his/her health statistics without delay from the PHR storage while the person cannot access any EMR facts directly.

Even though the patient statistics can be asked but the ownership of the EMR is the healthcare facility nevertheless. With the PHR concept, a man or woman can keep any of his/her health related records of his/her entire lifestyles and the records may additionally encompass some sorts of EMRs. The quantity of the PHR information can grow very speedy because the PHR statistics may be any health related information and the data of every man or woman will be introduced every day. According to [1], the PHR records carries variety of health related information such as hypersensitive reaction information, circle of relative's records, and a few laboratory test results. For this reason, the PHR data requires a massive storage for storing the massive PHR records and the facts within the storage must be accurate and on hand.

In different Phrases, PHR is some other utility that requires a big data management. Because the PHR belongs to the PHR owner, the access control at the PHR must ensure the protection of records. Furthermore, the PHR can also incorporate some sensitive records. Therefore, a safety mechanism to prevent any facts leakage should be applied on the PHR facts. Several PHR structures offer encryption methods for securing the PHR information and the encrypted information is later saved on a cloud storage [3].This way, the customers of these structures can ensure that their PHR statistics is safe due to the fact the information is encrypted and most effective the owner of the facts or the consumer who has the permission can decrypt it. But, these PHR structures save the encrypted PHR statistics immediately to the cloud storage.

There may be a trouble of the memory management because most PHR document is small. Hence, the small file length can consumes a huge memory of the cloud storage system that may lead to a device availability trouble. Therefore, a mechanism to address a various size of the encrypted PHR records on a cloud storage that also preserves the get entry to manage on such encrypted PHR statistics is needed. Every other challenge of this work is the encrypted PHR information which is not accessible by means of the storage system.

The storage device does not have any records on the data. Any preprocessing or categorizing mechanism on the encrypted records for performances is hard [4]. Such approach can store the encrypted PHR records. However, such technique isn't appropriate for retrieving the facts by using time and owner attributes as it takes a long term. The encrypted PHR information has some specific characters.

The PHR has a specific owner, size and arrival time of the information. The access sample of the PHR facts normally involves those three attributes. These three attributes of the encrypted PHR is used to enhance the overall performance of the distributed storage.

In this paper, we propose a design of an allotted storage for encrypted PHR facts. The design specializes in managing numerous report sizes of the encrypted PHR records. Both Hadoop [5] and HBase are used in our project. For evaluation purposes, Hadoop distributed file system (HDFS) that is storage of Hadoop is used as a general huge statistics framework to save the encrypted PHR data. The evaluation parameters consist of the memory intake of the proposed device.

2. LITERATURE SURVEY

Current PHR cloud storages are reviewed on this phase. This storage is designed for various necessities of the PHR system particularly an ability to store and retrieve massive PHR statistics successfully.

Medcloud [6] is a health care computing device that's design to observe the HIPAA privateness and some protection rules. Medcloud most important purpose is exchanging the health care records among health care carriers. The solution is moving the health care statistics to a cloud computing platform as a common location for sharing the records. This manner, the Medcloud consumer can keep his/her EMR or PHR information at the device. From the technical standpoint, Medcloud uses HBase for storing and retrieving the healthcare information.

Medoop [7] utilizes HDFS to keep the merged CDA documents correctly, organize the feature records in CDA documents in keeping with common business queries. Medoop makes use of Hadoop and HBase as its underlining framework. Medoop merges all documents to a massive one and creates an indexing report containing the records of all merged files. Medoop stores every the indexing document and it will merge the records on HDFS. Any often used records, but can be saved one after the alternative on HBase.

The improvement of CACSS [8] is an accepted cloud storage machine. The corporation or academic group which wishes to apply personal cloud storage for processing or storing their massive data can implement this type of system.

The system stores the information in HDFS and stores the information of the records (metadata) such as record name, file owner, and time on HBase. The system is capable of keep unstructured files. But, the storage does not design for fast retrieving in keeping with the PHR owner or the PHR arrival time attributes. Retrieving the particular PHR facts of a specific owner will take a long time because the system need to test all the metadata saved on HBase so as to clear out the requested PHR records.

Wiki-Health [9] is not only designed to solve the problems of managing, storing the high volume, velocity and variety of data, but also to offer support tools for users to create applications and analysis. The sensor records are stored on HBase because of its fast real-time statistics having access to overall performance. The difficulty rises all through the data query because the database schema of the system is designed specifically for handling the sensor facts.

Many cloud storage had been evolved on top of Hadoop and HBase due to the fact each are the main popular open supply huge facts frameworks. Hadoop provides reliability, parallel processing capability, high write throughput, and scalable storage functionality. Usually, a utility that calls for a huge storage and is a batch processing will use Hadoop. Actual time packages will use HBase. HDFS is the storage.

3. SYSTEM DESIGN

Systems design is the method of defining the structure, components, modules, interfaces, and facts for a device to meet particular requirements.

The PHR statistics is created by using the PHR consumer. The PHR consumer may be a patient or an individual who desires to save his/her health related records on the PHR machine. A PHR service allows a patient to create, manage, and control her personal health data in one place through the web, which has made the storage, retrieval, and sharing of the medical information more efficient.

Moreover, the PHR owner must be able to manage and control all authorized access to his/her PHRs. To achieve such features, the PHR system should allow the PHR owner to define an access control policy on his/her PHRs, which must be enforced by the PHR system. Thus, an individual can access a PHR if and only if that individual has been granted the authority by the PHR owner via an access control policy.

The below figure 1 shows a general block diagram describing the activities performed by this project.

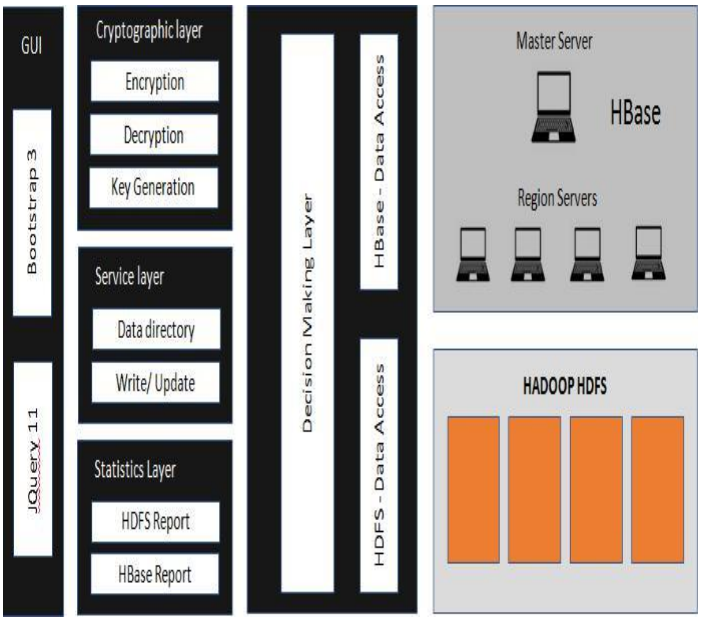

Fig -1: overview of the proposed system

The PHR API (Fig 1.) consists of the Data Access Layer Account Operations, Patients Registration, Patients Record Listing, Patients Health Data, Statistics and Read only Patients Records .

1) Data Access Layer

Data access layer is the one which exposes all the possible operations on the data base to the outside world. It will contain the Data Access Operations (DAO) classes, DAO interfaces. Plain old java objects (POJO) and Utils as the internal components. All the other modules of this project will be communicating with the DAO layer for their data access needs.

2) Account Operations

Account operations module provides the following functionalities to the end users of our project.

- Register a new seller/ buyer account
- Login to an existing account
- Logout from the session
- Edit the existing Profile
- Change Password for security issues
- Forgot Password and receive the current password over an email
- Delete an existing Account

Account operations module will be re-using the DAO layer to provide the above functionalities.

3) Patients Registration

Here, the doctors can register a new patient record by providing the required information about the patient like (patient name, patient email, phone number, gender, DOB, etc).

Upon clicking the Register button, the service layer will be reading all the inputs provided and construct the Patients POJO and make use of DAO (Data Access Operations) layer to persist the data in the database. Before the DAO layer comes into picture, the decision-making layer will be performing the encryption operation on the patients' POJO using RSA algorithm, and computes the size of the record in bytes. If the size is greater than a defined threshold, the DAO layer will persist the record in HDFS, else it will persist the record in HBase.

4) Patients Record Listing

Here, the doctors can retrieve the list of all registered patients' information. The request will be received by the service layer within which one of the servlet will be processing this by making use of DAO Layer, Decision making layer, and then the Cryptographic system. The service layer will first pull the records from both the HDFS and HBase and iterates over each record. For each record, the decision making layer will invoke the cryptographic system to decrypt each of the records and then returns the patients records listing to the user interface.

5) Patients Health Data

Here, the doctors can add or view the health data about any of the registered patients. Doctors will be able to access this module by clicking on a pointer link from the patients' record listing page. The health data page will consist of multiple panels each one of them corresponding to a visit of a patient in the hospital. For each visit, the doctor will have to enter the complaint (reason for visiting the hospital), observations, and then the treatment recommended.

6) Statistics

This module helps to doctors to understand the utilization of HDFS data model and HBase data model within the organization. This does so by retrieving all the records stored in HDFS and all the records stored in HBase separately and calculate the total size of all the records across each of those end points. Also, it calculates the average size of all the records across each of those end points. It also shows the individual patients records which are stored in each of those end points.

7) Read only Patients Records

This module can be used by the patients whose registration will be done by the doctors. Upon successful registration, the patients will be getting a welcome email to their registered email address. In this mail, they will able to view a link clicking upon which will take them to a page where they can access their personal information and the health-related information.

However, this information will be presented to them with the read only access thus avoiding the patients from modifying the health/personal data by themselves either by mistake or intentionally.

4. RESULTS

In this section we will present some of the screenshots of this project.

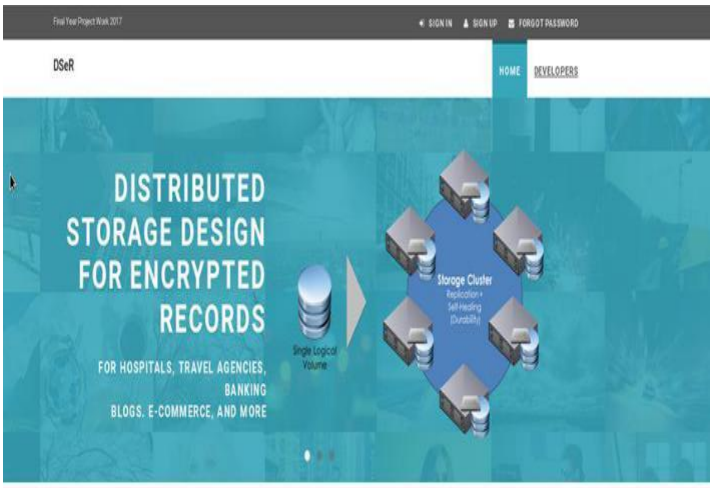

Fig-2: Front End Design of the project

Above Figure 2 shows the front end design of the project which shows sign in, sign up and forgot password.

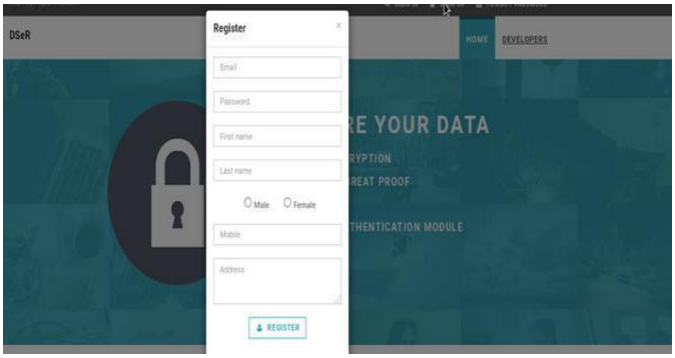

Fig-3: Registration of user

Above Figure 3 shows the registration form for the doctors.

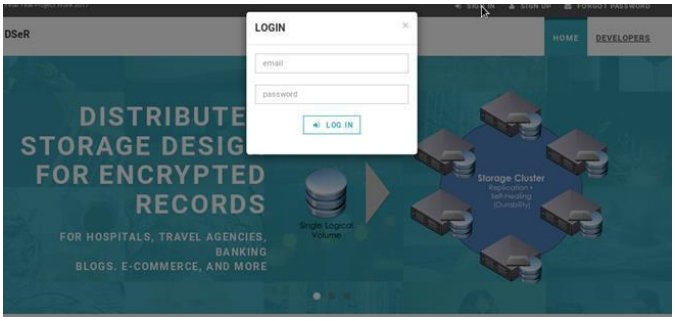

Fig-4: login page

Above figure 4 shows the login page. Registered users can login into the portal to register the patient information, retrieve the patient record stored in HDFS or HBASE and also the statistics of HDFS or HBASE.

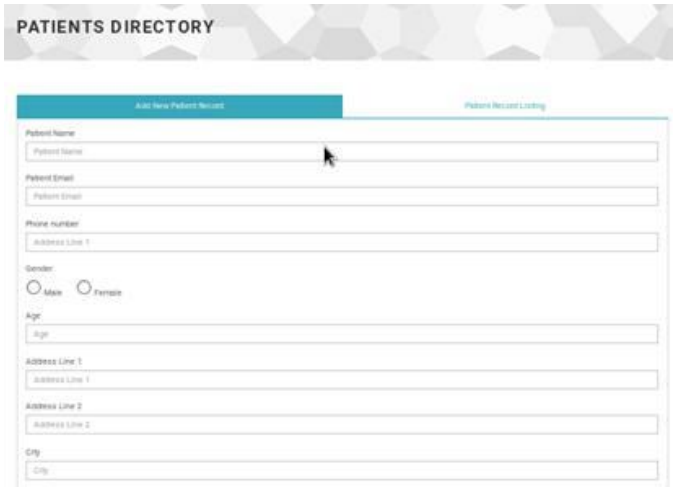

Fig-5: Patients registration form

Above figure 5 shows the patients registration form which contains the information patient name, phone number, address, city, country and pincode.

Fig-6: Retrieve of patient record

Above figure 6 shows the retrieving the encrypted patient record from the HDFS or HBSE through the patients ID.

Fig 7: HDFS statistics

Above figure 7 shows the statistics of HDFS.It provides the information about the total number of files stored in hdfs, Total size of data stored and the average size of the each record in the HDFS.

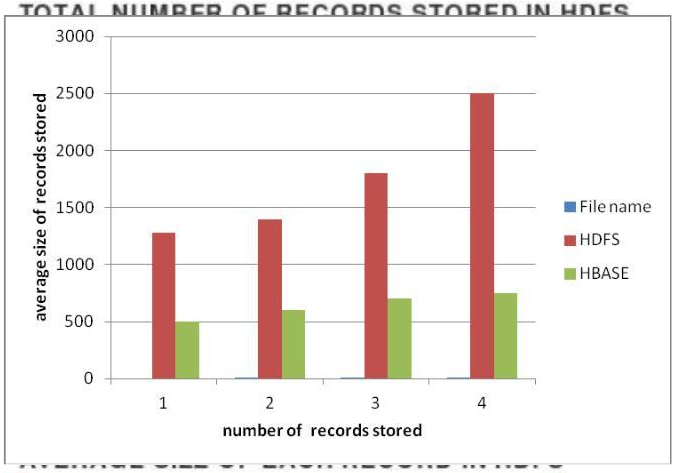

Fig-8: Shows the memory consumption of HDFS and HBASE as the number of files increases.

Above figure 8 shows the memory consumption of HDFS and HBASE .Files stored in the HDFS consumes more memory and HBASE consumes less memory.

5. CONCLUSION

Distributed storage of personal health record is proposed in this work to handle the encrypted PHR data on cloud storage.HBase and Hadoop are used in the proposed in this project. Large files will be stored on HDFS while small files are stored on HBase to solve the Namenode memory issues in HDFS. The encrypted PHR metadata is stored on HBase. In order to support the PHR accessing patterns a new schema of HBase is implemented. Hadoop is a growing ecosystem, which no embrace more and more component about data storage and analysis. The PHR data is typically accessed according to the PHR owner and the PHR arrival time attributes. The experimental results show that HBase schema developed in this work can correctly classify the encrypted PHR data and sends the data to an appropriated storage. This project resolves the Namenode memory issues, Memory consumption will be optimized and Scalable.