Real-time Health Monitoring System

Patients Vital Info Pipeline

1. There is a centralized RDS that contains patients' vital information like heartbeat, bp along with the customerId(patientid).
2. We need to create a Kafka topic to import the patient's vital information from RDS through a python script. For this, we need to set up the Kafka server and create a topic.
3. To set up Kafka server, we update java environment , download and extract Kafka package.
4. **Start Zookeeper server**
   Open a new terminal window and login to ec2 server. Navigate to Kafka folder and run below command to start zookeeper server.

- cd kafka_2.12-2.8.2
- bin/zookeeper-server-start.sh config/zookeeper.properties

5. **Start Kafka server**

   Open new terminal window
   1. Navigate to Kafka folder bin/kafka-server-start.sh config/server.properties

6. **Creating topic in Kafka server**

   Open new terminal window
   bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 -- partitions 1 --topic vitals.
7. Run the python script **kafka_produce_patient_vitals.py** to insert one message to kafka topic every second in order to achieve streaming. This python job will insert a total of 1800 messages for 30 minutes.
8. Create a EMR cluster with services such as spark, Hadoop, Hive, Sqoop.
9. Using the spark job **kafka_spark_patient_vitals.py,** read data from the kafka topic and insert into HDFS in parquet format.
10. Build a hive table "patients_vital_info" to query vital information being written to HDFS.

Threshold Reference Pipeline

1. Open hbase shell and create a hbase table 'Threshold_data' to store threshold data.
2. Create a Hive table 'Threshold_Reference' to query threshold data.

Patients Contact Info Pipeline

1. Patients Contact info data is stored in RDS.
2. We use sqoop to import the data to hive table.
3. Run below commands to install Sqoop

wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz

tar -xvf mysql-connector-java-8.0.25.tar.gz
cd mysql-connector-java-8.0.25/

sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/

4. Run sqoop import command to copy data from RDS to hive table “Patients_Contact_Info”.

Generate Alerts

1. We create a spark streaming application **kafka_spark_generate_alerts.py** to compare patients’ vital info with threshold data.
2. If a patient’s vital information matches with threshold data for which alert flag is set, we need to send notifications.
3. For such cases, we fetch the patient’s contact info and along with vital information, we insert into a Kafka topic for sending alert notifications.
4. Using AWS SNS console, we create a SNS topic and subscribe to this topic using an email address so as to receive alert email messages.
5. A python script **kafka_consume_alerts.py** reads messages from the Kafka topic and sends it to SNS topic for sending alert email message.
6. Thus, we get an email message whenever a patient’s vital info is abnormal.