IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 2,February 2015.
www.ijiset.com

ISSN 2348 – 7968

# Securing Health Care Information by Using Two-Tier Cipher Cloud Technology

**M.Divya[1] J.Gayathri[2] A.Gladwin[3]**

[1]B.TECH Information Technology, Jeppiaar Engineering College, Chennai

[2]B.TECH Information Technology, Jeppiaar Engineering College, Chennai

[3] Assistant Professor Information Technology,Jeppiaar Engineering College,Chennai

**Abstract – Cloud Computing technology is an enticing technology that provides massive computation power and storage capacity by using a large number of computers together, enables the users to deploy applications in a shared environment. Because of its salient feature, cloud is promising for users to handle the personal health records .Personal Health Records (PHR) are user friendly online solution that helps patients to manage their health details. However cloud technology has a large number risks and vulnerabilities. One of the major problems in moving to cloud computing is its security and privacy issues. Cloud computing provides powerful and economical infrastructure for users to handle ever increasing data sets in health care applications. But sharing privacy-sensitive health records on cloud probably leads to privacy concerns because of multi-tenancy system. Dataset encryption and anonymization are the two widely-adopted ways to prevent privacy breach. The encryption technique that is used is not suitable for data that is processed and shared frequently and anonymizing big data and managing anonymized data sets are still challenges for securing details of patients. In this paper, we propose a scalable two-phase top-down specialization (TDS) approach to anonymize large-scale data sets using the MapReduce framework on cloud. In both the phases of our approach, we design a group of innovative MapReduce algorithms to concretely accomplish the specialization computation in a highly scalable way and ensure efficiency.**

**Index Terms— Personal Health Record, Data anonymization, top-down specialization, MapReduce, cloud, privacy preservation.**

## I.INTRODUTION

Electronic health records are usually managed by different health care units such as physicians, therapists, hospitals and pharmacies. Using PHRs in cloud environment, secure and flexible access control can be achieved. Electronic health records are usually deemed extremely sensitive although these data can offer significant human benefits if they are analyzed and mined by organizations. Data privacy can be breached with less effort by malicious cloud users or providers because of the failures of some traditional privacy protection measures on cloud. Data anonymization refers to hiding identity and/or sensitive data for owners of data records. Then, the privacy of an individual can be effectively preserved while certain aggregate information is exposed to data users for diverse analysis and mining. But one of the main problems with this is scalability. This could be overcome by using framework like MapReduce which is used to perform parallel processing and thus the scalability problem of Top-Down approach could be overcome.

A highly scalable two-phase TDS approach for data anonymization based on MapReduce is used to secure the personal health record. The first stage involves the use of MapReduce for data anonymization. Second, a two-phase TDS approach to gain high scalability by allowing specializations to be conducted on multiple data partitions in parallel during the first phase.

## II RELATED WORKS

We have reviewed the recent research papers on data privacy preservation in Map Reduce and cloud computing environments.

LeFevre et al addressed the scalability problem of anonymization algorithms by introducing scalable decision trees and sampling techniques. Iwuchukwu et al proposed an R-tree index-based approach by building a spatial index over data sets to achieve high efficiency. However, the above method aims at multidimensional generalization that fails to work in the Top-Down Specialization (TDS) approach. Fung et al proposed the Centralized TDS approach that produces anonymous data sets without the data exploration problem. A data structure Taxonomy Indexed Partition (TIPS) is exploited to improve the efficiency of TDS. But the approach is centralized which leads to its inadequacy in handling large-scale data sets. Several distributed algorithms are proposed to preserve privacy and increase the efficiency. Jiang et al and Mohammed et al proposed distributed algorithms to anonymize vertically partitioned data from different data sources without disclosing privacy information from one user to another. Jurczyk et al and

ISSN 2348 – 7968

Mohammed et al proposed distributed algorithms to anonymize horizontally partitioned data sets retained by multiple users. However, the above distributed algorithm mainly aims at securely integrating and anonymizing multiple data but does not ensures the privacy of the data sets. Our paper mainly focuses on the scalability issue of TDS anonymization, and ensures the optimization of large data sets.

## III Problem Analysis

The scalability and privacy problems of existing TDS approach is analysed**.** The centralized TDS approaches uses the data structure TIPS to improve the scalability and efficiency by indexing large data records and retaining statistical information in TIPS. The data structure used here speeds up the process because indexing structure avoids frequent scanning of the entire data sets and stores the statistical results. On the other hand, the amount of metadata retained to maintain the statistical information and linkage information of record partitions is relatively very large compared with data sets, thus consuming considerable memory. The overheads incurred by maintaining the linkage structure and updating the statistic information will be huge when date sets become large. Hence, centralized approaches probably suffer from low efficiency and scalability when handling large-scale data sets. In centralized approach, there is an assumption that all data processed should fit in memory. But this assumption often fails to hold in most data-intensive cloud applications. In cloud environments, computation is provisioned in the form of virtual machines (VMs). Usually, cloud compute services offer several VMs. As a result, the centralized approaches are difficult in handling large-scale data sets on cloud using just one single VM even if the VM has the highest computation and storage capability. A distributed TDS approach is thus proposed to address the distributed anonymization problem which mainly concerns privacy protection against other parties, rather than scalability issues. Further, the approach only employs information gain, rather than its combination with privacy loss, as the search metric when determining the best specializations. A Distributed TDS algorithm without considering the privacy loss probably chooses a specialization that leads to a quick violation of anonymity requirements. Hence, the distributed algorithm fails to produce anonymous data sets exposing the same data utility as centralized ones. Besides, the issues like communication protocols and fault tolerance must be kept in mind when designing such distributed algorithms. As such it is inappropriate to leverage existing distributed algorithms to solve the scalability problem of TDS. Thus to ensure both scalability and privacy issues, MapReduce with two-phase TDS is used.

## IV Two-Phase Top-Down Specialization (TPTDS)

TDS is an iterative process starting from the topmost domain values in the taxonomy trees of attributes. Each round of iteration consists of three main steps, namely, finding the best specialization, performing specialization and updating values of the search metric for the next round. Such a process is repeated until k-anonymity is violated, to expose the maximum data utility. The two phases are based on the two levels of parallelization provisioned by MapReduce on cloud. MapReduce on cloud has two levels of parallelization, job level and task level. Job level parallelization means that multiple MapReduce jobs can be executed simultaneously to make full use of cloud infrastructure resources. Combined with cloud, MapReduce becomes more powerful and elastic as cloud can offer infrastructure resources on demand. Task level parallelization refers to that multiple mapper tasks in a MapReduce job are executed simultaneously over data splits.

### 1. Algorithm for Two-Phase Top-Down specialization

**Input**: Data set D, anonymity parameters k, $k^I$ and the number of partitions p.
**Output**: Anonymous data set D*.
1: Partition D into Di, 1<= i <=p.
2: Execute MRTDS (Di, $K^I$, $AL^0$) function in parallel for multiple MapReduce jobs.
3: Merge all intermediate anonymization levels into one merge function.
4: Execute MRTDS (Di, $K^I$, $AL^0$) to achieve k-anonymity.
5: Specialize D according to AL*, Output D*.

The influence of p and $K^I$ the efficiency is analyzed as follows. Greater p means higher degree of parallelization in the first phase, and less $K^I$ indicates more computation is conducted in the first phase. Thus, greater p and less $K^I$ improve the efficiency.

### 2. Data Partition

When D is partitioned into $D_i$, it is required that the distribution of data records in Di is similar to D.
A data record can be treated as a point in an m-dimension space, where m is the number of attributes. Thus, the intermediate anonymization levels derived from Di, 1 <= i <=p, can be more similar so that we can get a better merged

ISSN 2348 – 7968

anonymization level. Random sampling technique is adopted to partition D, which can satisfy the above requirement. Specifically, a random number rand, 1 <= rand <=p, is generated for each data record. A record is assigned to the partition Drand.

### 2.1 Algorithm for Data Partition

The Algorithm shows the MapReduce program of data partition. The number of Reducers should be equal to p, so that each Reducer handles one value of rand, exactly producing p resultant files. Each file contains a random sample of D.

**Input**: Data record (IDr, r), r Є D, partition parameter p.

**Output**: Di, 1<= i <= p.

Map: Generate a random number rand, where 1<= rand l<= p; emit (rand, r).

Reduce: For each rand, emit (null, list(r)).

Once partitioned data sets Di,1<=i<=p, are obtained, we run MRTDS (Di,kI,AL0) on these data sets in parallel to derive intermediate anonymization levels $AL_i$* , 1<= i <=p.

All intermediate anonymization levels are merged into one in the second phase. The merging of anonymization levels is completed by merging cuts.

## 3. Data Specialization

The original data set D is concretely specialized for anonymization in a one-pass MapReduce job. After obtaining the merged intermediate anonymization level $AL^I$, MRTDS (D, k, $AL^I$) on the entire data set D, and get the final anonymization level AL*. Then, the data set D is anonymized by replacing original attribute values in D with the responding domain values in AL*.Details of Map and Reduce functions of the data specialization MapReduce job are described in Algorithm. The Map function emits anonymous records and its count. The Reduce function simply aggregates these anonymous records and counts their number. An anonymous record and its count represent a QI-group. The QI-groups constitute the final anonymous data sets.

# IV MAP REDUCE VERSION OF CENTRALIZED TDS

## 1. MRTDS Driver

Usually, a single MapReduce job is inadequate to accomplish a complex task in many applications. Thus, a group of MapReduce jobs are orchestrated in a driver program to achieve such an objective. MRTDS consists of MRTDS Driver and two types of jobs, i.e., IGPL Initialization and IGPL Update. The driver arranges the execution of jobs. MRTDS Driver is where a data set is anonymized by TDS. It is the algorithmic design of function MRTDS (D, k, AL) →AL0.we leverage anonymization level to manage the process of anonymization. Step 1 initializes the values of information gain and privacy loss for all specializations, which can be done by the job IGPL Initialization.

The best specialization is selected from valid specializations in current anonymization level. A specialization spec is a valid one if it satisfies two conditions. One is that its parent value is not a leaf, and the other is that the anonymity Ac (spec) > k, i.e., the data set is still k-anonymous if spec is performed. Then, the current anonymization level is modified via performing the best specialization. The information gain of the newly added specializations and privacy loss of all specializations need to be recomputed, which are accomplished by job IGPL Update. The iteration continues until all specializations become invalid, achieving the maximum data utility. MRTDS produces the same anonymous data as the centralized TDS, because they follow the same steps. MTRDS mainly differs from centralized TDS on calculating IGPL values. However, calculating IGPL values dominates the scalability of TDS approaches, as it requires TDS algorithms to count the statistical information of data sets iteratively. MRTDS exploits MapReduce on cloud to make the computation of IGPL parallel and scalable. We present IGPL Initialization and IGPL Update subsequently.

## 2. IGPL Initialization

The Map and Reduce functions of the job IGPL Initialization are described in above algorithm. The main task of IGPL Initialization is to initialize information gain and privacy loss of all specializations in the initial anonymization level AL. The number of records in each current QI-group needs computing, so does the number of records in each QI-group after potential specializations.

**Input:** Data record (IDr, r), r Є D; anonymization level AL.

**Output**: Intermediate key-value pair (key, count).

1: For each attribute value vi in r; find its specialization in current AL: spec. Let p be the parent in spec and c is the p's child value that is also an ancestor of vi in TTi.

2: For each vi, emit ((p, c, sv), count)

ISSN 2348 – 7968

3Construct quasi-identifier qid*= (p1, p2, pm) where pi, $1 \leq i \leq m$, is the parent of a specialization in current AL.

To compute the anonymity of data sets before and after a specialization, we finds the smallest number of records out of all current QI-groups and finds all the smallest number of records out of all potential QI-groups for each specialization. The next emits the results of anonymity. Note that there may be more than one key- value pair (spec, A (spec)) for one specialization in output files if more than one reducer is set. But we can find the smallest anonymity value in the driver program. The privacy loss PL (spec) is computed. Finally, IGPL (spec) for each specialization is obtained.

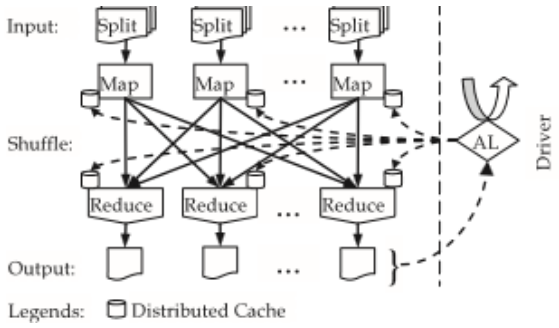
### 3. IGPL Update Job

The IGPL Update job dominates the scalability and efficiency of MRTDS, since it is executed iteratively. So far, iterative MapReduce jobs have not been well supported by standard MapReduce framework like Hadoop. Accordingly, Hadoop variations like Haloop and Twister have been proposed recently to support efficient iterative MapReduce computation. Our approach is based on the standard MapReduce framework to facilitate the discussion herein. The IGPL Update job is quite similar to IGPL Initialization, except that it requires less computation and consumes less network bandwidth. Thus, the former is more efficient than the latter.The Reduce function is the same as IGPL Initialization.

## V IMPLEMENTATION AND OPTIMIZATION

To elaborate how data sets are processed in MRTDS, the execution framework based on standard MapReduce is depicted in Fig.1. The solid arrow lines represent the data flows in the canonical MapReduce framework. The iteration of MapReduce jobs is controlled by anonymization level AL in Driver. The data flows for handling iterations are denoted by dashed arrow lines. AL is dispatched from Driver to all workers including Mappers and Reducers via the distributed cache mechanism. The value of AL is modified in Driver according to the output of the IGPL Initialization or IGPL Update jobs. As the amount of such data is extremely small compared with data sets that will be anonymized, they can be efficiently transmitted between Driver and workers.

To reduce communication traffics, MRTDS exploits combiner mechanism that aggregates the key-value pairs with the same key into one on the nodes running Map functions. To further reduce the traffics, MD5 (Message Digest Algorithm) is employed to compress the records transmitted for anonymity. As anonymity computation causes the most traffic as it emits m key-value pairs for each original record, this can considerably reduce network traffics. The Reduce function in Algorithm can still correctly compute anonymity without being aware of the content of qid.

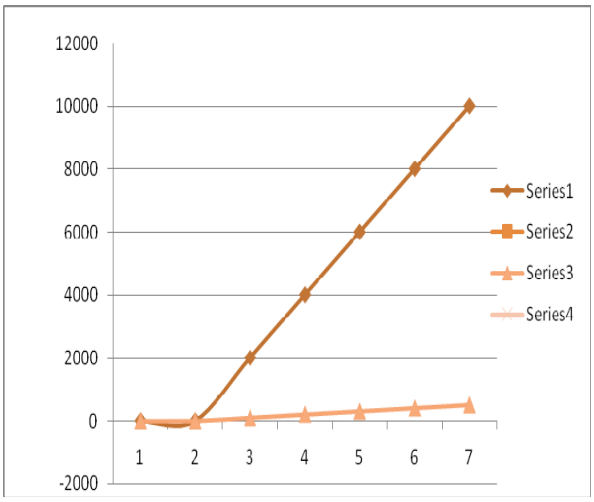



## VI EVALUATION

### 1. Experiment Evaluation and Experiment Settings

Our experiments are conducted in a cloud environment named U-Cloud. U-Cloud is a cloud computing environment at the University of Technology Sydney (UTS). The system overview of U-Cloud has been depicted. The computing facilities of this system are located among several labs at UTS. On top of hardware and Linux operating system, we install KVM virtualization software that virtualizes the infrastructure and provides unified computing and storage resources. To create virtualized data centers, we install OpenStack open source cloud environment for global management, resource scheduling and interaction with users. Further, Hadoop clusters are built based on the OpenStack cloud platform to facilitate large-scale data processing.

ISSN 2348 – 7968




**Experiment Process and Results**

We conduct three groups of experiments in this section to evaluate the effectiveness and efficiency of our approach. In the first one, we compare TPTDS with CentTDS from the perspectives of scalability and efficiency. In the other two, we investigate on the tradeoff between scalability and data utility via adjusting configurations. Generally, the execution time and ILoss are affected by three factors, namely, the size of a data set (S), the number of data partitions (p), and the intermediate anonymity parameter (kI).

## VII CONCLUSIONS AND FUTURE WORK

In this paper, we have investigated the scalability problem of large-scale data anonymization by TDS, and proposed a highly scalable two-phase TDS approach using MapReduce on cloud. Data sets are partitioned and anonymized in parallel in the first phase, producing intermediate results. Then, the intermediate results are merged and further anonymized to produce consistent k-anonymous data sets in the second phase. We have creatively applied MapReduce on cloud to data anonymization and deliberately designed a group of innovative MapReduce jobs to concretely accomplish the specialization computation in a highly scalable way. Experimental results on real-world data sets have demonstrated that with our approach, the scalability and efficiency of TDS are improved significantly over existing approaches. In cloud environment, the privacy preservation for data analysis, share and mining is a challenging research issue due to increasingly larger volumes of data sets, thereby requiring intensive investigation.

We will investigate the adoption of our approach to the bottom-up generalization algorithms for data anonymization. Optimized balanced scheduling strategies are expected to be developed towards overall scalable privacy preservation in future.

## VIII REFERENCES

[1] H.Takabi, J.B.D. Joshi, and G. Ahn, “Security and Privacy Challenges in Cloud Computing Environments,” IEEE Security and Privacy, vol. 8, no. 6, pp. 24-31, Nov. 2010.
[2] N.Cao, C. Wang, M. Li, K. Ren, and W. Lou, “Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data,” Proc. IEEE INFOCOM, pp. 829-837, 2011.
[3] B.C.M. Fung, K. Wang, and P.S. Yu, “Anonymizing Classification Data for Privacy Preservation,” IEEE Trans. Knowledge and Data Eng., vol. 19, no. 5, pp. 711-725, May 2007.
[4] V. Borkar, M.J. Carey, and C. Li, “Inside ‘Big Data Management’: Ogres, Onions, or Parfaits” Proc. 15th Int’l Conf. Extending Database Technology (EDBT ’12), pp. 3-14, 2012.
[5] B. Fung, K. Wang, L. Wang, and P.C.K. Hung, “Privacy- Preserving Data Publishing for Cluster Analysis,” Data and Knowledge Eng., vol. 68, no. 6, pp. 552-575, 2009.
[6] W.Jiang and C.Clifton, “A Secure Distributed Framework for Achieving k-Anonymity,” VLDB J., vol. 15, no. 4, pp. 316-333, 2006.
[7] J.Ekanayake, H.Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, “Twister: A Runtime for Iterative Mapreduce,” Proc. 19th ACM Int’l Symp. High Performance Distributed Computing (HDPC ’10), pp. 810-818, and 2010.
[8] Amazon Web Services, “Amazon Elastic Mapreduce,” http:// aws.amazon.com/elasticmapreduce/, 2013. [30] Apache, “Hadoop,”http://hadoop.apache.org, 2013.
[9] Y. Bu, B. Howe, M. Balazinska, and M.D. Ernst, “The Haloop Approach to Large-Scale Iterative Data Analysis,” VLDB J., vol. 21, no. 2, pp. 169-190, 2012.
[10] UCI Machine Learning Repository, ftp://ftp.ics.uci.edu/pub/ machine-learning-databases/, 2013.