Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 7, July 2015, pg.489 – 496

RESEARCH ARTICLE

BIG-DATA ANONYMIZATION USING THE MAPREDUCE ON CLOUD

K.Bhuvaneswari[1], Dr. V.Palanisamy[2]
M.Phil Research Scholar[1] and Professor & Head[2]
Department of Computer Science &Engineering[12]
Alagappa University, Karaikudi, TamilNadu, India
Kbuvana1991@gmail.com [1]

ABSTRACT:

An expansive number of cloud administrations oblige clients to impart private information like electronic wellbeing records for information examination or mining, bringing security concerns. Anonymizing information sets through speculation to fulfill certain security prerequisites, for example, k anonymization are a generally utilized classification of protection protecting procedures. At present, the size of information in numerous cloud applications increments immensely as per the Big Data pattern, consequently making it a test for generally utilized programming devices to catch, oversee, and process such huge scale information inside an average glided by time. Thus, it is a test for existing anonymization ways to reach to protection safeguarding on security delicate extensive scale information sets because of their deficiency of flexibility. The proposed is a worker node process contacts the cache manager each time before it begins processing an input data file. The worker node process sends the file name and the operations that it plans to apply to the file to the cache manager. The cache manager mechanically identifies the best-matched cache item to feed each reducer, which one is the maximum overlap in the original input file in the map phase.

Keywords—Big-data, Map Reduce, Hadoop, distributed file system, cache management.

I. INTRODUCTION

Map reduce, and its open-source implementation Hadoop[12], are a software framework for largescale distributed computing on large amounts of data.applications specify the computation in terms of a map and reduce function working on partitioned data items. The map reduce framework schedules computation on unprecedently large-scale dataset. However ,there is a limitation of the system,i.e.,the efficiency in incremental processing .incremental processing refers to the applications that incrementaly grow the input data and continuously apply computation on the input in order to generate output . There are potential duplicate computation being performed in this process.however map reduce does not have the mechanisms to identitfy such duplicate computations.motivation by this observation

The proposed, a data aware cache manager system for Big-data applications using the Map Reduce framework. It aims at extending the Map Reduce framework and provisioning a cache layer for efficiently identifying and accessing cache items in a Map Reduce job. The following technical challenges need to be addressed before implementing this proposal: 1) Cache description scheme. Data-aware caching requires each data object to be indexed by its content. In the context of Big-data applications, this means that the cache description scheme needs to describe the application framework and the data contents. Although most big-data applications run on standardized platforms, their individual tasks perform completely different operations and generate different intermediate results. 2) Cache request and reply protocol. The size of the aggregated intermediate data can be very large. Usually the programs are moved to data nodes in order to avoid network communications. The protocol should be able to collate cache items with the worker processes potentially that need the data, so that the transmission delay and overhead are minimized. This in the scheme identifies the source input from which a cache item is obtained, and the operations applied on the input. In the reduce phase, we devise a mechanism to take into consideration the partition operations applied on the output in the map phase.A method for reducers to utilize the cached results in the map phase to accelerate their execution. The implement two phase in the Hadoop[11] approach by extending the relevant components. Our implementation follows a non-intrusive approach, so it only requires minimum changes to the application code.

II. Related work

K. LeFevre, D.J. DeWitt, and R. Ramakrishnan[1], has addressed problem of scalability .They introduced two techniques based on scalable decision trees and sampling in order to allow anonymization algorithms to be applied to datasets larger than main memory. Also they stated quality of data is best judged with respect to the workload for which the data will ultimately be used. These algorithms are highly efficient and result will be high quality data. But this approach fails in top down specialization approach and fails to work in multiple data sets.

K. Zhang, X. Zhou, Y. Chen, X. Wang, and Y. Ruan[3], Sedic is designed to protect data privacy in map reduce operations. Sedic offers a privacy-aware hybrid computing paradigm. Sedic schedules Map's such that jobs on private clouds handles sensitive data while jobs on public clouds operate on nonsensitive data. It automatically extracts Combiner's from Reduce functions that allow public clouds to process data.

X. Zhang, C. Liu, S. Nepal, S. Pandey, and J. Chen,[4] Upper bound privacy leakage constraint-based approach to identify which intermediate data sets need to be encrypted and which do not. Privacy-preserving cost of intermediate data sets can be significantly reduced significantly. But this process is highly complicated, efficient processing of data is quite challenging.

N. Cao, C. Wang, M. Li, K. Ren, and W. Lou,[2] Prism is privacy preserving search system specially designed for cloud computing. It provides storage and query processing. Prism is designed to features like parallelism, efficiency of Map Reduce. If we want to use prism, we doesn't require to modify the underlying system. Use of prism will not introduce extra overhead on system. Disadvantage of prism is we could not secure public clouds using this approach.

III.PROPOSED SYSTEM

A highly scalable two-phase TDS approach for data Anonymization based on Map Reduce on cloud. To make full use of the parallel capability of Map Reduce on cloud, specializations required in an Anonymization process are split into two phases. In the first one, original data sets are partitioned into a group of smaller datasets, and these data sets are Anonymized in parallel, producing intermediate results. In the second one, the intermediate results are integrated into one, and further Anonymized to achieve consistent k-anonymous[9] datasets. The leverage Map Reduce to accomplish the concrete computation in both phases. A group of Map Reduce jobs is deliberately designed and coordinated to perform specializations on data sets collaboratively. We evaluate our approach by conducting experiments on real-world

datasets. Experimental results demonstrate that with our approach, the scalability and efficiency of TDS can be improved significantly over existing approaches.

The major contributions of our research are threefold. First, we creatively apply Map Reduce on cloud to TDS for data Anonymization[10] and deliberately design a group of innovative Map Reduce jobs to concretely accomplish the specializations in a highly scalable fashion. Second, we propose a two-phase TDS approach to gain high scalability via allowing specializations to be conducted on multiple data partitions in parallel during the first phase. Third, experimental results show that our approach can significantly improve the scalability and efficiency of TDS for data Anonymization over existing approaches. This scheme identifies the source input from which a cache item is obtained, and the operations applied on the input.

The benefit of this approach is no scalability issue in large scale data sets,Efficiently handling the large scale data sets using distributed approach,Map-Reduce[7] using TDS Approach, Less computational over head,Less storage space.

METHODOLOGY

A. Develop Top-Down Specialization

Generally, Top-Down Specialization (TDS) is an iterative process starting from the Top most 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, in order *to* expose the maximum data utility. The goodness of a specialization is measured by a search metric. We adopt the Information Gain per Privacy Loss (IGPL), a trade-off metric that considers both the privacy and information requirements, as the search metric in our approach. A specialization with the highest IGPL value is regarded as best one and selected of each round. We describe briefly how we calculate the value of IGPL subsequently to make readers understand our approach well. Interested reader scan refer to for more details.

B. Job Allocation Methodology

The Master Node runs multiple processes including a job tracker and name node. The job Tracker is Responsible for managing running jobs in the cluster. The Name node on the other

hand manages the hadoop distributed file system. The data partition is performed on the cloud. Here it collects the large no of data sets. It are split the large into small data sets. Then provides the random no for each data sets. Partitioning is the process of determining which reducer instance will receive which intermediate keys and values. Each mapper must determine for all of its output (key, value) pairs which reducer will receive them. It is necessary that for any key, regardless of which mapper instance generated it, the destination partition is the same. If the key cat is generated in two separate (key, value) pairs, they must both be reduced together.

C. Cache Manager Methodology

The cache manager maintains a copy of the mapping between the cache descriptions and the file names of the cache items in its main memory to accelerate queries. It also flushes the mapping file into the disk periodically to avoid permanently losing data. A worker node/process contacts the cache manager each time before it begins processing an input data file. The worker process sends the file name and the operations that it plans to apply to the file to the cache manager. The cache manager automatically identifies the best-matched cache item to feed each reducer, which is the one with the maximum overlap in the original input file in the Map phase.

D. Map Phase Process Methodology

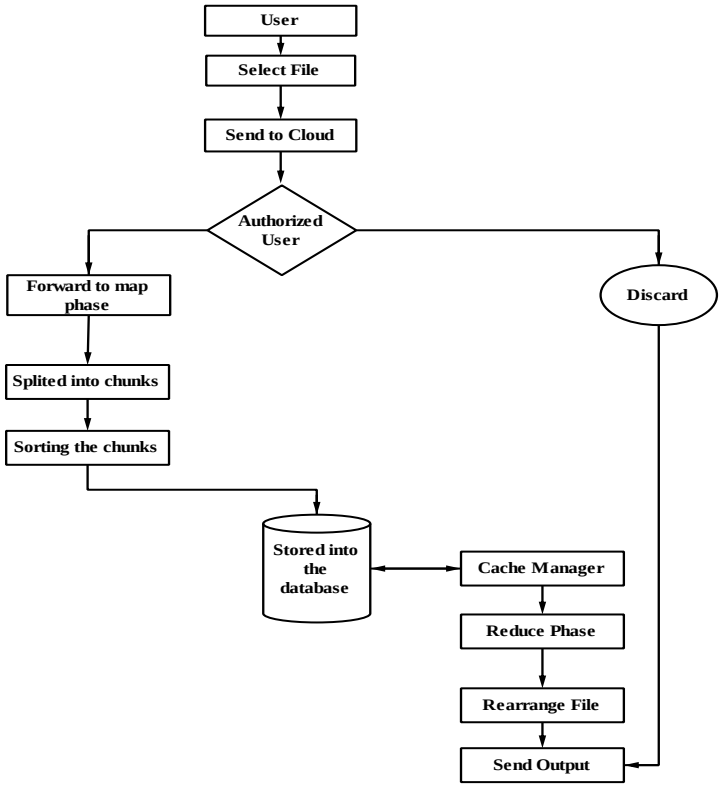
Map Reduce[7] is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster. A Map Reduce program is composed of a Map() procedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name) and a Reduce() procedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies). The Map Reduce[7] System is orchestrates by marshalling the distributed servers, running the various tasks in parallel, managing all communications and data transfers between the various parts of the system, providing for redundancy and fault tolerance, and overall management of the whole process. The model is inspired by the map and reduces functions commonly used in functional programming, although their purpose in the Map Reduce framework is not the same as their original forms.

E. Reducephase Methodology

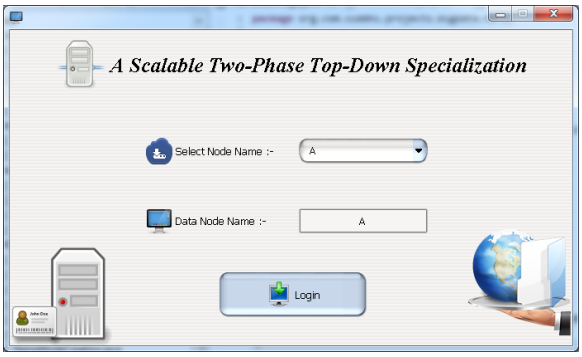
The input for the reduce phase is also a list of key-value pairs, where the value could be a list of values. Much like the scheme used for the map phase cache description, the original input and the applied operations are required. The original input is obtained by storing the intermediate results of the map phase in the DFS. The applied operations are identified by unique IDs that are

specified by the user. The cached results, unlike those generated in the Map phase, cannot be directly used as the final output.

IV. SYSTEM MODEL



V. SIMULATION/EXPERIMENTAL RESULTS



The Namenode or Admin node successfully registered, the Admin node upload the file to cloud .The cache manager maintains the copy of the upload file. When the user needs the file the users track the file easily and get the result in timely manner.

The cache manager maintains all upload file list and forward to each name node the user get the result through cache manager. Otherwise the file does not exist the cache manager immediately inform the user

VI. CONCLUSION

The conclusion of the proposed work is it using the two phase top down approach to provide ability to handles the high amount of the large data sets. And here it provides the privacy by effective anonymization approaches. In the future work is reducing the handling effect of large amount of the data sets. Its implements the optimized balancing scheduling. Where it's based on the time and size of the data sets. In this paper it have investigated the scalability problem of large-scale data anonymization by Top-Down Specialization and proposed a highly scalable two-phase TDS approach using Map Reduce on cloud. Datasets are partitioned and anonymized in parallel in the first phase producing intermediate results.

References

[1] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Workload-Aware Anonymization Techniques for Large-Scale Data Sets," ACM Trans. Database Systems, vol. 33, no. 3, pp. 1-47, 2008.

[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] K. Zhang, X. Zhou, Y. Chen, X. Wang, and Y. Ruan, "Sedic: Privacy-Aware Data Intensive Computing on Hybrid Clouds," Proc. 18th ACM Conf. Computer and Comm. Security (CCS '11), pp. 515-526, 2011.

[4] X. Zhang, C. Liu, S. Nepal, S. Pandey, and J. Chen, "A Privacy Leakage Upper-Bound Constraint Based Approach for Cost-Effective Privacy Preserving of Intermediate Data Sets in Cloud," IEEE Trans. Parallel and Distributed Systems, to be published, 2012.

[5] L. Hsiao-Ying and W.G. Tzeng, "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 6, pp. 995-1003, 2012.

[6] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "Gupt:Privacy Preserving Data Analysis Made Easy," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '12), pp. 349-360, 2012.

[7]. I.Roy,S.T.V. Setty,A.K.Ilzer,v.Shmatikov,and E.Witchel, "Airavat : Security and Privacy For Map Reduce " proc.Seventh USENIX Conf.Networked Systems Design and implementation(NSDI'10),pp.297-312,2010

[8] Amazon Web Services, "Amazon Elastic Map Reduce," http://aws.amazon.com/elasticMap Reduce/, 2013.

[9]. W.Jiang And C.Clifton,"A Secure Distributed Framework For Achieving K-Anonymity ," VLDB ,J.,vol.15, No.4,pp.316-333,2006.

[10] N. Mohammed, B. Fung, P.C.K. Hung, and C.K. Lee, "Centralized and Distributed Anonymization for High-Dimensional Healthcare Data," ACM Trans. Knowledge Discovery from Data, vol. 4, no. 4,Article 18, 2010.

[11]. Apache,"Hadoop ,http://hadoop.apache.org,2013.

[12].Y. Bu, B. Howe,M.Balazinska, and M.D.Ernst,"The Hadoop approach to large scale iterative Data Analyzes ," VLDB J.,vol.21,No .2,pp.169-190,2012

