Elastic Data Sharing Services for Cloud Based On Best Peer-To-Peer Networks

SHAIK SHAMEERBABU[1], CH. SUBBA RAO[2], P.BABU[3]
[1]PG Scholar, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.
[2]Associate Professor, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.
[3]Associate Professor, Dept of CSE, Quba College of Engineering and Technology, Nellore, AP, India.

Abstract: The corporate network is often used for sharing information among the participating companies and facilitating collaboration in a certain industry sector where companies share a common interest. It can effectively help the companies to reduce their operational costs and increase the revenues. However, the inter-company data sharing and processing poses unique challenges to such a data management system including scalability, performance, throughput, and security. In this paper, we present BestPeer++, a system which delivers elastic data sharing services for corporate network applications in the cloud based on BestPeer—a peer-to-peer (P2P) based data management platform. By integrating cloud computing, database, and P2P technologies into one system, BestPeer++provides an economical, flexible and scalable platform for corporate network applications and delivers data sharing services to participants based on the widely accepted pay-as-you-go business model. We evaluate BestPeer++ on Amazon EC2 Cloud platform. The benchmarking results show that BestPeer++ outperforms HadoopDB, a recently proposed large-scale data processing system, in performance when both systems are employed to handle typical corporate network workloads. The benchmarking results also demonstrate that BestPeer++ achieves near linear scalability for throughput with respect to the number of peer nodes.

Keywords: BestPeer++, HadoopDB, Peer-To-Peer (P2P).

I. INTRODUCTION

Companies of the same industry sector are often connected into a corporate network for collaboration purposes. Each company maintains its own site and selectively shares a portion of its business data with the others. Examples of such corporate networks include supply chain networks where organizations such as suppliers, manufacturers, and retailers collaborate with each other to achieve their very own business goals including planning production-line, making acquisition strategies and choosing marketing solutions. From a technical perspective, the key for the success of a corporate network is choosing the right data sharing platform, a system which enables the shared data (stored and maintained by different companies) network-wide visible and supports efficient analytical queries over those data. Traditionally, data sharing is achieved by building a centralized data warehouse, which periodically extracts data from the internal production systems (e.g., ERP) of each company for subsequent querying. Unfortunately, such a warehousing solution has some deficiencies in real deployment. the corporate network needs to scale up to support thousands of participants, while the installation of a large-scale centralized data warehouse system entails nontrivial costs including huge hardware/software investments (a.k.a total cost of ownership) and high maintenance cost (a.k.a total cost of operations). BestPeer++ achieves its query processing efficiency and is a promising approach for corporate network applications, with the following distinguished features.

A. Motivations

The era of cloud computing reigns with advancements in technology, the technology provides various services to the human’s need and also it urges the more necessity for the emerging technology. Could computing provides a platform for other advanced technologies like big data, mobile computing to inculcate its service and provide the QOS to the customers. The cloud has grown to a vast extend over the period of years. All the services that are provided to the customer are done using could as their backbone, it give vast amount of resources and infrastructure to consumer who acts as vendors to small scale business and cloud could provide services to fully fledged organization with less cost. Organizing the service and extending the service depending upon the growing needs of the customer could be achieved by cloud service and infrastructure. The major issue is the resources, while any service needs to be extended, the resources with the service vendor plays a vital role. Investing huge sum of dollars on hardware is just one part of extension, maintaining the hardware along the services provided would carry tons of dollars. Where cloud provides space for extending the services as a service provider and also it can provide infrastructure service to small scale service vendors.

B. Objective

The main objective of this work is BestPeer++ is deployed as a service in the cloud. To form a corporate network, companies simply register their sites with the BestPeer++ service provider, launch BestPeer++ instances in the cloud and finally export data to those instances for sharing. BestPeer++ adopts the pay-as-you-go business model popularized by cloud computing. Through a web console interface, companies can easily configure their access control policies and prevent undesired business partners to access their shared data. The data are indexed by the table name, column name and data range for efficient retrieval. It can effectively help the companies to reduce their operational costs and increase the revenues. However, the inter-company data sharing and processing poses unique challenges to such a data management system including scalability, performance, throughput, and security. In this paper, we present BestPeer++, a system which delivers elastic data sharing services for corporate network applications in the cloud based on BestPeer—a peer-to-peer (P2P) based data management platform. By integrating cloud computing, database, and P2P technologies into one system, BestPeer++ provides an economical, flexible and scalable platform for corporate network applications and delivers data sharing services to participants based on the widely accepted pay-as-you-go business model. We evaluate BestPeer++ on Amazon EC2 Cloud platform.

II. EXISTING SYSTEM

Such a warehousing solution has some deficiencies in real deployment. First, the corporate network needs to scale up to support thousands of participants, while the installation of a large-scale centralized data warehouse system entails nontrivial costs including huge hardware/software investments (a.k.a total cost of ownership) and high maintenance cost (a.k.a total cost of operations). In the real world, most companies are not keen to invest heavily on additional information systems until they can clearly see the potential return on investment (ROI).Second, companies want to fully customize the access control policy to determine which business partners can see which part of their shared data.

Disadvantages of Existing System: Most of the data warehouse solutions fail to offer such flexibilities. Solution has not been designed to handle such dynamicity.

III. PROPOSED SYSTEM

The main contribution of this paper is the design of BestPeer++ system that provides economical, flexible and scalable solutions for corporate network applications. We demonstrate the efficiency of BestPeer++ by benchmarking BestPeer++ against Hadoop DB, a recently proposed large-scale data processing system, over a set of queries designed for data sharing applications. The results show that for simple, low-overhead queries, the performance of BestPeer++ is significantly better than Hadoop DB as shown in Fig.1. The unique challenges posed by sharing and processing data in an inter-businesses environment and proposed BestPeer++, a system which delivers elastic data sharing services, by integrating cloud computing, database, and peer-to-peer technologies.

Advantages of Proposed System: Our system can efficiently handle typical workloads in a corporate network and can deliver near linear query throughput as the number of normal peers grows. BestPeer++ adopts the pay-as-you-go business model popularized by cloud computing. The total cost of ownership is therefore substantially reduced since companies do not have to buy any hardware/software in advance. Instead, they pay for what they use in terms of BestPeer++ instance's hours and storage capacity. BestPeer++ extends the role-based access control for the inherent distributed environment of corporate networks. BestPeer++ employs P2P technology to retrieve data between business partners. BestPeer++ is a promising solution for efficient data sharing within corporate networks.

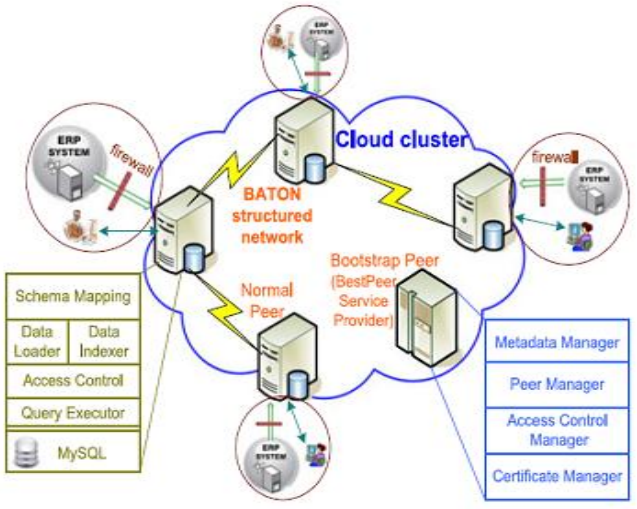


Fig.1. System Architecture.

The BestPeer++ core contains all platform-independent logic, including query processing and P2P overlay. It runs on top of the Cloud adapter and consists of two software components: bootstrap peer and normal peer. A BestPeer++ network can only have a single bootstrap peer instance which is always launched and maintained by the BestPeer ++ service provider, and a set of normal peer instances. The architecture is depicted. This section briefly describes the functionalities of these two kinds of peer. Individual components and data flows inside these peers are presented in the subsequent sections.

IV. IMPLEMENTATION AND ALGORITHM

The normal peer software consists of five components: Schema mapping, data loader, data indexer, access control, and query executor. We present the first four components in this section. Query processing in BestPeer++ will be presented in the next section. There are two data flows inside the normal peer: an offline data flow and an online data flow. In the offline data flow, the data are extracted periodically by a data loader from the business production system to the normal peer instance. In particular, the data loader extracts the data from the business production system, transforms the data format from its local schema to the shared global schema of the corporate network according to the schema mapping, and finally stores the results in the MySQL databases hosted in the normal peer.

A. Schema Mapping

Schema mapping is a component that defines the mapping between the local schema of each production system and the global shared schema employed by the corporate network. Currently, BestPeer++ only supports relational schema mapping, namely both local schema and the global schema are relational. The mapping consists of metadata mappings (i.e., mapping local table definitions to global table definitions) and value mappings (i.e., mapping local terms to global terms). Besides schema level mapping, BestPeer++ can also support instance level mapping, which complements the mapping process when there is not sufficient schema information. In general, the schema mapping process requires human to be involved and is rather time consuming. However, it only needs to perform once. Furthermore, BestPeer++ adopts templates to facilitate the mapping process. Specifically, for each popular production system (i.e., SAP or PeopleSoft), we provide a mapping template which defines the transformation of local schemas of those systems to a global schema. What the business only needs is to modify the mapping template to meet its own needs. We found that this mapping template approach works well in practice and significantly reduces the service setup efforts.

B. Data Loader

Data Loader is a component that extracts data from production systems to normal peer instances according to the result of schema mapping. While the process of extracting and transforming data is straightforward, the main challenge comes from maintaining consistency between raw data stored in the production systems and extracted data stored in the normal peer instance (and subsequently data indices created from these extracted data) while the raw data being updated inside the production systems. When the data loader first extracts data from the production system, besides storing the results in the normal peer instance, the data loader also creates a snapshot of the newly inserted data3. After that, at interval times, the data loader re-extracts data from the production system to create a new snapshot. This snapshot is then compared to the previously stored one to detect data changes. Finally, the changes are used to update the MySQL database hosted in the normal peer.

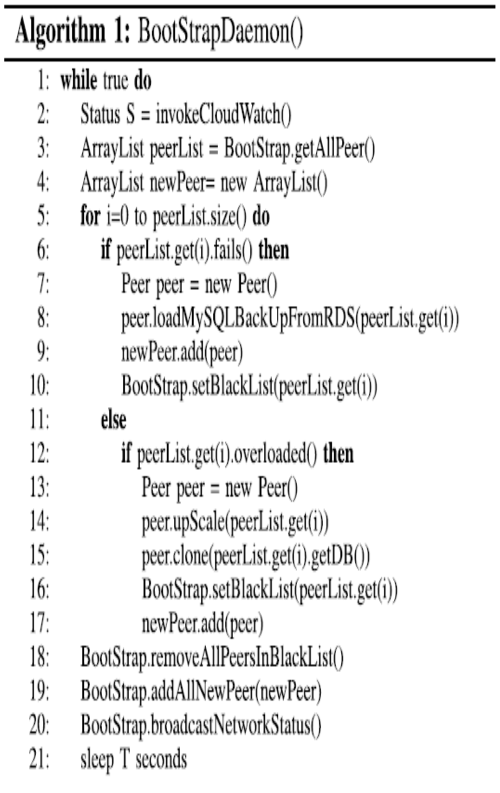
C. Data Indexer

In the BestPeer++, the data are stored in the local MySQL database hosted by each normal peer. Thus, to process a query, we need to locate which normal peers host the tables involved in the query. For example, to process a simple query like select R.a from R where R.b=x, we need to know the location of the peers store tuples belonging to the global table R. We adopt the peer-to-peer technology to solve the data locating problem and only send queries to normal peers which host related data. In particular, we employ BATON a balanced binary tree overlay protocol to organize all normal peers. Fig. 3 shows the structure of BATON. Given a value domain [L, U], each node in BATON is responsible for two ranges. The first range, R0, is the sub domain maintained by the node. The second range, R1, is the domain of the sub tree rooted at the node. For example, R0 and R1 are set to 38Þ for node D, respectively. For a key k, there is one unique peer p responsible for k and k is contained by p: R0. For a range l; u, there is also one unique peer p for it and

- u is a sub-range of p: R1; and
- If u is contained by bp: R1, bp must be p’s ancestor node. If we traverse the tree via in-order, we can access the values in consecutive domains.

In BATON, each node maintains log2N routing neighbours in the same level, which are used to facilitate the search process in this index structure. To achieve a balanced structure, BATON employs two flexible load balancing schemes. A node can balance its load with adjacent nodes when there exists under loaded ones.

Algorithm 1:



Algorithm 1: BootStrapDaemon()

```
while true do
    Status S = invokeCloudWatch()
    ArrayList peerList = BootStrap.getAllPeer()
    ArrayList newPeer= new ArrayList()
    for i=0 to peerList.size() do
        if peerList.get(i).fails() then
            Peer peer = new Peer()
            peer.loadMySQLBackUpFromRDS(peerList.get(i))
            newPeer.add(peer)
            BootStrap.setBlackList(peerList.get(i))
        else
            if peerList.get(i).overloaded() then
                Peer peer = new Peer()
                peer.upScale(peerList.get(i))
                peer.clone(peerList.get(i).getDB())
                BootStrap.setBlackList(peerList.get(i))
                newPeer.add(peer)
    BootStrap.removeAllPeersInBlackList()
    BootStrap.addAllNewPeer(newPeer)
    BootStrap.broadcastNetworkStatus()
    sleep T seconds
```

D. Distributed Access Control

The access to multi-businesses data shared in a corporate network needs to be controlled properly. The challenge is for BestPeer++ to provide a flexible and easy-to-use access control scheme for the whole system; at the same time, it should enable each business to decide the users that can access its shared data in the inherent distributed environment of corporate networks. BestPeer++ develops a distributed role-based access control scheme. The basic idea is to use roles as templates to capture common data access privileges and allow businesses to override these privileges to meet their specific needs.

Algorithm 2:

```
Algorithm 2: Adaptive Query Processing
  Input: Query Q
  Output: Query configuration on a specific query
          engine
  TableSet $S \longleftarrow TableParser(Q)$ ;
  Cost $C_{min} \longleftarrow MAX\_VALUE$ ;
  QueryPlan $Target \longleftarrow null$ ;
  QueryPlanSet $QS \longleftarrow \emptyset$ ;
  foreach Table $T \in S$ do
      // Generate Processing Graphs
         rooted on T
      GraphSet $GS = GraphGen(T)$;
      // Iterate through all Processing
         Graph rooted on T
      foreach Graph $G \in GS$ do
          QueryPlan $P_1 = P2PPlanGen(G)$;
          QueryPlan $P_2 = MapredPlanGen(G)$;
          $QS = QS \cap \{P_1\}$;
          $QS = QS \cap \{P_2\}$;

  foreach QueryPlan $P \in QS$ do
      if $CostEst(P) < C_{min}$ then
          $C_{min} = CostEst(P)$;
          $Target = P$;

  return $Target$;
```

E. Modules

Data Owner: In this module, the data owner has to register in a cloud server and fire walls (fire wall1 and fire wall2), after registration he has to login. Data owner uploads their data file into the cloud server and the cloud server will connect to fire wall, the data file is stored in a normal peer and the backup file will be stored in Bootstrap peer. The Data owner can have capable of manipulating the stored data file.

Cloud Cluster: The cloud cluster consists of fire walls (firewall1 and firewall2) and peers (normal peer1, normal peer2 and Bootstrap peer).The cloud server is responsible for data storage and file authorization for an end user. The normal peer software consists of five components: schema mapping, data loader, data indexer, access control, and query executor. The data file will stored in normal peer with their tags such as file name, secret key, digital sign, and owner name. If the end user requested file is correct then the data will be sent to the corresponding user and also will check the file name, end user name and secret key. If all are true then it will send to the corresponding user or he will be captured as attacker.

Bootstrap PEER: In this module, the bootstrap peer is the entry point of the whole network.It has several responsibilities. First, the bootstrap peer serves for various administration purposes; including monitoring and managing normal peers and also scheduling various network management events. Second, the bootstrap peer acts as a central repository for meta data of corporate network applications, including shared global schema, participant normal peer list, and role definitions.

Data Consumer(End User): The data consumer is nothing but the end user who will request and gets file contents response from the corresponding cloud servers and fire walls. End user should register before downloading any files from the cloud server.

Attacker: If user is entered a wrong secrete key, then considered as an attacker. Attacker is one who is integrating the cloud file by adding malicious data to the corresponding cloud. The peer which is not having more count in attacker list is called as best peer.

V. SIMULATION RESULTS

Simulation results of this paper is shown in bellow Figs.2 to 4.

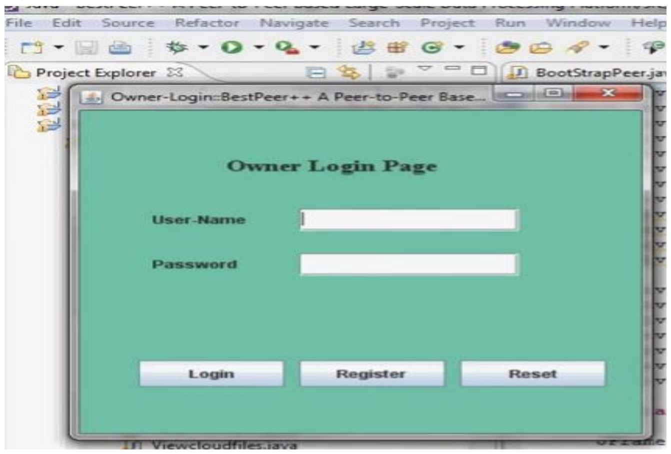

File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
BootStrapPeer.java
Owner-Login::BestPeer++ A Peer-to-Peer Base...
Owner Login Page
User-Name
Password
Login
Register
Reset
Viewcloudfiles.java

Fig.2. Owner login page.

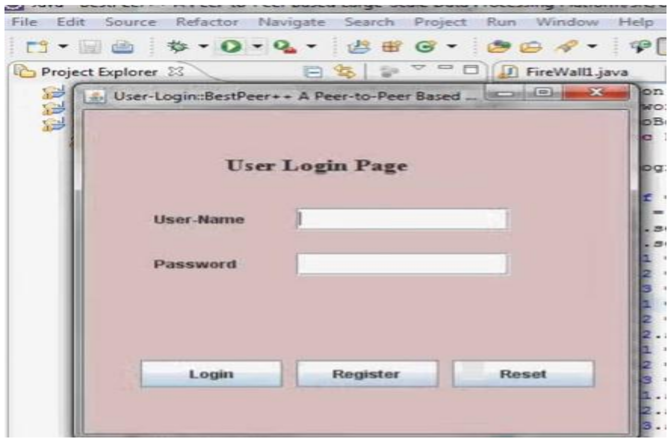

File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
FireWall1.java
User-Login::BestPeer++ A Peer-to-Peer Based ...
User Login Page
User-Name
Password
Login
Register
Reset

Fig.3. User login page.

Registeration::BestPeer++ A Peer-to-Peer Based Large-Scale...

User Registration Form

Username	
Password	
Address	
City	
Email	
Mobile	

Fig.4.User Registration page.

VI. CONCLUSION

The unique challenges posed by sharing and processing data in an inter-businesses environment and proposed BestPeer++, a system which delivers elastic data sharing services, by integrating cloud computing, database, and peer-to-peer technologies. The benchmark conducted on Amazon EC2 cloud platform shows that our system can efficiently handle typical workloads in a corporate network and can deliver near linear query throughput as the number of normal peers grows. Therefore, BestPeer++ is a promising solution for efficient data sharing within corporate networks.

VII. REFERENCES

[1] K. Aberer, A. Datta, and M. Hauswirth, “Route Maintenance Overheads in DHT Overlays,” in 6th Workshop Distrib. Data Struct., 2004.
[2] A. Abouzeid, K. Bajda-Pawlikowski, D.J. Abadi, A. Rasin, and A. Silberschatz, “HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads,” Proc. VLDB Endowment, vol. 2, no. 1, pp. 922-933, 2009.
[3] C. Batini, M. Lenzerini, and S. Navathe, “A Comparative Analysis of Methodologies for Database Schema Integration,” ACM Computing Surveys, vol. 18, no. 4, pp. 323-364, 1986.
[4] D. Bermbach and S. Tai, “Eventual Consistency: How Soon is Eventual? An Evaluation of Amazon s3’s Consistency Behavior,” in Proc. 6th Workshop Middleware Serv. Oriented Comput. (MW4SOC ’11), pp. 1:1-1:6, NY, USA, 2011.
[5] B. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, “Benchmarking Cloud Serving Systems with YCSB,” Proc. First ACM Symp. Cloud Computing, pp. 143-154, 2010.
[6] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, “Dynamo: Amazon’s Highly Available Key-Value Store,” Proc. 21st ACM SIGOPS Symp. Operating Systems Principles (SOSP ’07), pp. 205-220, 2007.
[7] J. Dittrich, J. Quian_e-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad, “Hadoop++: Making a Yellow Elephant Run Like a Cheetah (without it Even Noticing),” Proc. VLDB Endowment, vol. 3, no. 1/2, pp. 515-529, 2010.
[8] H. Garcia-Molina and W.J. Labio, “Efficient Snapshot Differential Algorithms for Data Warehousing,” technical report, Stanford Univ., 1996.
[9] Google Inc., “Cloud Computing-What is its Potential Value for Your Company?” White Paper, 2010.
[10] R. Huebsch, J.M. Hellerstein, N. Lanham, B.T. Loo, S. Shenker, and I. Stoica, “Querying the Internet with PIER,” Proc. 29th Int’l Conf. Very Large Data Bases, pp. 321-332, 2003.
[11] H.V. Jagadish, B.C. Ooi, K.-L. Tan, Q.H. Vu, and R. Zhang, “Speeding up Search in Peer-to-Peer Networks with a Multi-Way Tree Structure,” Proc. ACM SIGMOD Int’l Conf. Management of Data, 2006.
[12] H.V. Jagadish, B.C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, “iDistance: An Adaptive B+-Tree Based Indexing Method for Nearest Neighbor Search,” ACM Trans. Database Systems, vol. 30, pp. 364-397, June 2005.
[13] H.V. Jagadish, B.C. Ooi, and Q.H. Vu, “BATON: A Balanced Tree Structure for Peer-to-Peer Networks,” Proc. 31st Int’l Conf. Very Large Data Bases (VLDB ’05), pp. 661-672, 2005.
[14] A. Lakshman and P. Malik, “Cassandra: Structured Storage System on a P2P Network,” Proc. 28th ACM Symp. Principles of Distributed Computing (PODC ’09), p. 5, 2009.
[15] W.S. Ng, B.C. Ooi, K.-L. Tan, and A. Zhou, “PeerDB: A P2P-Based System for Distributed Data Sharing,” Proc. 19th Int’l Conf. Data Eng., pp. 633-644, 2003.
[16] Oracle Inc., “Achieving the Cloud Computing Vision,” White Paper, 2010.
[17] V. Poosala and Y.E. Ioannidis, “Selectivity Estimation without the Attribute Value Independence Assumption,” Proc. 23rd Int’l Conf. Very Large Data Bases (VLDB ’97), pp. 486-495, 1997.
[18] M.O. Rabin, “Fingerprinting by Random Polynomials,” Technical Report TR-15-81, Harvard Aiken Computational Laboratory, 1981.
[19] E. Rahm and P. Bernstein, “A Survey of Approaches to Automatic Schema Matching,” The VLDB J., vol. 10, no. 4, pp. 334-350, 2001.
[20] P. Rodr_ıguez-Gianolli, M. Garzetti, L. Jiang, A. Kementsietsidis, I. Kiringa, M. Masud, R.J. Miller, and J. Mylopoulos, “Data Sharing in the Hyperion Peer Database System,” Proc. Int’l Conf. Very Large Data Bases, pp. 1291-1294, 2005.