IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

FINE GRAINED ACCESS CONTROL IBS IN DISTRIBUTED SHARING

[#1]P.MADHAVI - M.Tech Pursuing,
[#2] R.SATISH KUMAR -Assistant Professor,
Department of Computer Science & Engineering,
MALLAREDDY ENGG.COLLEGE FOR WOMEN, Hyderabad.

Abstract*— In contrast with the situations when the information seeker knows where the needed data is located, XML Information Brokering System (IBS) needs to help each information seeking query "locate" the corresponding data source(s). Unlike early information sharing approaches that only involve a small number of databases, new information sharing applications are often assumed to be built atop a large volume of geographically distributed databases, such as emergence health care. Moreover, with increasing concerns on protecting the sensitive and/or proprietary data, the organizations prefer sharing data in a more secure and privacy-preserving manner, instead of establishing a purely full trust relationship and releasing the control over the shared data. In this paper, we explore new information sharing infrastructures to address the new challenges on *security, privacy, load balancing, trust management* and *scalability.

In this work, we present a flexible and scalable XML IBS using a broker-coordinator overlay network. Through a novel automaton segmentation scheme, in-network access control, distributed load balancing, trust management and query segment encryption, our system integrates security enforcement and query forwarding while preserving system-wide privacy. We first explore access control deployment strategies in distributed information sharing and the impacts of different deployment strategies on system-wide performance and security. From our study, we are motivated to enforce in-network access control by combining query security checking function with query routing function in *Query Brokers and coordinators* while maintaining distributed load balancing among peers.

We perform a formal presentation of the threat models with a focus on two attacks: *attribute-correlation attack* and *inference attack*. Then, we propose a broker-coordinator overlay, as well as three schemes, automaton segmentation scheme, query segment encryption scheme and trust management scheme to share the secure query routing function among a set of brokering servers. We carried out a comprehensive analysis on privacy, end-to-end performance, load balancing and scalability, the proposed system integrate security enforcement, load balancing, trust management between peers and query routing while preserving system-wide privacy with reasonable overhead.

***Keywords*— Privacy, XML, Access Control, load balancing, information sharing, peer to peer, PPIB**

I. INTRODUCTION

XML appears in fact a natural choice as the basis for the common security-policy language, due to the ease with which its syntax and semantics can be extended and the widespread support that it enjoys from all the main platform and tool vendors. XML information brokering is the process of collecting and re-distributing XML information to the information seekers [2]. Accessing information on the global Internet has become an essential requirement of the modern economy. Information is however one of, if not the most valuable asset of an organization. An important requirement of any system is then to protect its data and services against unauthorized disclosure (secrecy or confidentiality) and unauthorized or improper modifications (integrity), while at the same time ensuring their availability to legitimate users (no denial-of-service or availability) [10, 13, 18]. However, as technology advances and information management systems become more and more powerful, the problem of enforcing information security and privacy also becomes more critical. A fundamental component in enforcing protection is represented by the access control service whose task is to control every request to data/services maintained by a system and determining whether the request should be granted or denied. The access control service establishes the kinds of regulations (policies) that can be stated, through an appropriate specification language, and then enforced by the access control mechanism enforcing the service. By using the provided interface, security administrators can specify the access control policy that should be obeyed in controlling access to the managed resources.

In Distributed, information sharing the data entities is shared among several inter-communicating computers, wherein at least one of said computers is a server computer, and wherein each computer, which is not a server computer, is a client computer. e.g. Let us consider a medicare network scenario. Each organization (e.g., hospital) participates as a data source that holds its own patient database. Since the records are highly sensitive and private, intensive privacy and security enforcement is desired. Diverse users (e.g., doctors, assistants, pharmacists, and administrators) are to access local or remote patient data according to certain access control policies. Furthermore, users who ask queries from their own terminals do not have to have prior knowledge of data distribution. For instance, when a doctor wants to retrieve all the historical records of a patient, her query may be forwarded to all data sources

IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

that hold related information. However, the user does (and should) not need to known where the data comes from.

Traditional information sharing approaches always assume the use of trustable servers, such as the central data warehousing server or database servers. However, the honest or semi-honest assumptions (e.g., honest-but-curious assumption as adopted in [2]) may not hold for brokers. In practice, they may either be abused by insiders or compromised by outsiders. It is obvious that the brokers become the most vulnerable privacy breach of XML IBS, which leads to inevitable security and privacy risks. On one hand, the survival of information brokering depends on the trust of brokers to enforce authentication, access control, peer trust management as well as query forwarding, while on the other hand, failing to provide proper protection of information released in this process may create circumstances that harm the privacy of user, data and the system.

To satisfy such privacy protection requirements, therefore, in this work we propose a novel XML IBS, named as Privacy Preserving Information Brokering system (PPIB). As shown in Fig. 1, PPIB contains a broker-coordinator overlay network, in which the brokers are responsible for providing authentication, load balancing and trust management including forwarding user queries to coordinators concatenated in tree structure while preserving privacy. The coordinators, each holding a segment of access control automaton and routing guidelines, are mainly responsible for access control and query routing.

PPIB takes an innovative automaton segmentation approach to enforce secure and privacy protection. In particular, two critical forms of privacy, namely query content privacy and data object distribution privacy (or data location privacy), are enabled by a novel automaton segmentation scheme, with a "little" help from an assisting query segment encryption scheme. Distributed load balancing by using virtual servers to store load information on each peer to dynamically shift the query processing from heavily loaded peers to lightly loaded peers is enhanced in this system. PPIB perform trust management essential to evaluate the trustworthiness of participating peers and to combat the selfish, dishonest, and malicious peer behaviors.

While providing "full" capability to do in-network access control and to route queries to the right data sources, automaton segmentation scheme and trust management scheme ensure the information that a (curious, corrupted or broken) coordinator can gather is far from being enough to infer either "which data is being queried" or "where the data is located". Second, the automaton segmentation scheme can also provide high-quality privacy protection to metadata (e.g., access control policy). Third, user location privacy is protected by multilateral security, a design principle of PPIB.

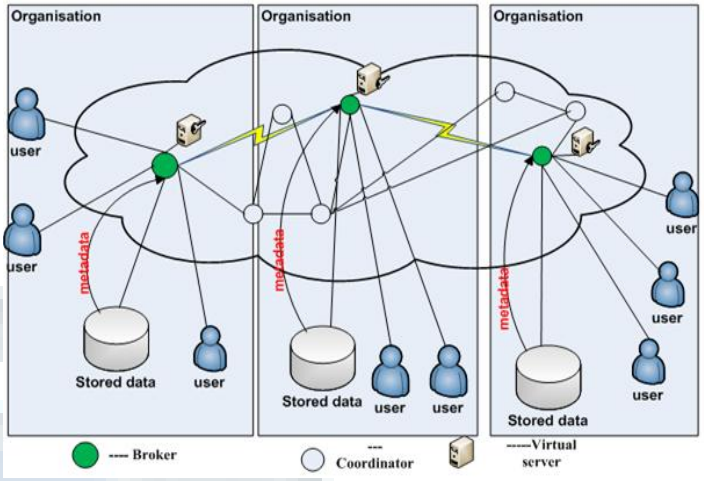


Fig. 1 System architecture of PPIB system

II. BACKGROUND

This section gives the background of this work as follows:

A. *XML requirement*

To support rich semantics, we assume data are queried and exchanged in XML format. Data is assembled into XML documents, conforming to XML syntax and semantic rules. An XML document consists of elements, attributes, and text nodes. An element has a set of attributes, and may contain other XML elements and text nodes. Thus, these elements collectively form a tree-base data structure. The widely adopted XML standard allows people to abstract naive data representations (e.g., patient records) into semi-structured XML data which can be retrieved by expressive yet simple XPath queries. XPath is a restricted variation of regular path expressions, which can refer to all or part of the nodes in an XML document using axes [1, 2, 17]. Axes represent the structural relationships between nodes. In particular, an axis defines a set of nodes relative to the current node. For example, "/" denotes the child node, "//" denotes the current node itself and all the descendant nodes, and"@"denotes the attribute. Although several query languages using different query algebras have emerged recently, most of them use XPath for locating nodes in XML documents. Thus, although our system is applicable to any regular path expression and any query language based on it, we focus on XPath in this paper. This paper will focus on semantically rich applications such as health care.

B. *Access control representation*

Access control is required in most if not all XML IBS. We adopt the popular XML access control model proposed in [13, 20]. In this model, users are members of appropriate roles; and an access control policy consists of a set of role-based 5-tuple access control rules (ACR): R = {subject, object, action, sign, type}, where (1) subject is a role to whom an authorization is granted; (2) object is a set of XML nodes specified by XPath; (3) action is one of

IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

"read,""write," and "update"; (4) sign ∑ {+, −} refers to access "granted" or "denied," respectively; and (5) type ∑{LC,RC} refers to either "Local Check" (i.e., authorization is only applied to attributes or textual data of context nodes-"self::text() | self::attribute()"), or "Recursive Check" (i.e., authorization is applied to context nodes and propagated to all descendants-"descendant-or-self::node()"). When an XML node does not have either explicit (via LC rules) or implicit (via RC rules) authorization, it is considered to be "access denied." It is possible for an XML node to have more than one relevant access control rule. If conflict occurs between "+" and "–" rules, "–" rules take precedence. In our XML IBS, each owner contributes a policy governing the access to her data objects, and the system-wide access control policy is simply the union of all the per-owner policies.

C. Automaton-based Access Control Enforcement

View-based access control enforcement suffers from excessive storage requirement and expensive maintenance. Many view-free XML access control mechanisms are proposed to overcome the disadvantage. In our approach, we adopt and extend a view-free automaton-based access control mechanism proposed in [15]. It uses XPath expressions in access control rules (ACR) to build a Non-deterministic Finite Automaton (NFA). We call such a NFA an access control automaton. Each incoming query is checked against the NFA. As a result, each query could be (1) accepted: when user is allowed (by ACR) to access all the requested nodes, the query is kept as is. (2) rewritten: when user is allowed to access part of the requested content, the query is rewritten into a safe one, which asks for authorized content only. (3) denied: when user is not allowed to access any requested node, query is rejected.

D. Distributed load balancing

A node's (coordinator's) load may vary greatly over time since the system can be expected to experience continuous insertions and deletions of objects (data items), skewed object arrival patterns, and continuous arrival and departure of coordinators. We propose a load balancing algorithm that uses the concept of virtual servers previously proposed in [7]. A virtual server represents a peer in the DHT (Distributed Hash Table), that is, the storage of data items and routing happen at the virtual server level rather than at the physical node level. A physical node (broker) hosts one or more virtual servers. Load balancing is achieved by moving virtual servers from heavily loaded physical nodes to lightly loaded physical nodes.

E. Reputation management system for Trust level on each peer

A major challenge for large-scale PPIB systems is how to establish trust between different peers (broker-coordinators) without the benefit of trusted third parties or authorities. Usually the coordinators don't have any pre-existing relationship and may reside in different security domains. Sometimes even when there are some authorities available, e.g., an authentication server or certification authority, it is inadvisable to assume that these authorities can monitor transactions and then declare the trustworthiness of different peers. In this work we use Reputation-based trust systems, which use information considering previous interactions with an entity to establish a reputation measure that will support a trust decision [4] [6]. This category of trust systems, due to their wide applicability in P2P systems is examined.

III. Privacy Preserving Information Brokering approach (PPIB)

In this section, we propose an innovative Privacy Preserving Information Brokering (PPIB) framework to address the user/data/metadata privacy vulnerabilities associated with existing distributed information brokering systems, including the IBS presented in [13].

A. Automaton Segmentation Algorithm

It uses XPath expressions in access control rules (ACR) to build a Non-deterministic Finite Automaton (NFA). Each incoming query is checked against the NFA. As a result, each query could be accepted, rewritten or denied. In PPIB, it adopt the view-free automaton-based access control mechanism [10, 16], and extend it in a decentralized manner with this Automaton Segmentation scheme. The idea of automaton segmentation comes from the concept of multilateral security: split sensitive information to largely meaningless shares held by multiple parties who cooperate to share the privacy-preserving responsibility.

Automaton segmentation scheme first divides the global access control automaton into several segments. Granularity of segmentation is controlled by a parameter partition size, which denotes how many XPath states in the global automaton are partitioned and put into one segment. By and large, the granularity is a choice of the system administrator. Higher granularity leads to better privacy preserving, but also more complex query processing. Each accepts state of the global automaton is specially partitioned as a separate segment. Then assign each segment to one independent site. As a result, a site in essence holds a small automaton. At run-time, it conducts NFA-based access control enforcement as a stand-alone component. However, in the state transition table of the last state of each segment, the "next state" points to a root state at a remote site, instead of a local state.

In PPIB, a site is actually a logical unit. So a physical coordinator (i.e., a machine) can in fact hold multiple sites. For convenience, add dummy accept states to each automaton segment. The dummy accept states do not accept queries. Instead, they are used to store the location of actual "next states," i.e. the address(es) of the coordinators who hold the next segment of the global automaton. At runtime,

IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

they are used to forward the halfway processed query to the next coordinators. On the other hand, only the sites holding original accept states accept queries and forward them to the data servers. As a result, access control and query brokering are seamlessly integrated at coordinators, and the global automaton-based query brokering mechanism is decentralized and distributed among many coordinators.

Algorithm The automaton segmentation algorithm:
Deploy_Segment()

Input: Automaton State S
Output: Segment Address: addr
1: **for each** symbol k in S: StateTransTable **do**
2: addr=deploy_Segment(S: StateTransTable (k): nextState)
3: DS=createDummyAcceptState ()
4: DS: nextState ← addr
5: S: StateTransTable (k): nextState ← DS
6: **end for**
7: Seg= createSegment ()
8: Seg.addSegment(S)
9: Coordinator = getCoordinator ()
10: Coordinator.assignSegment (Seg)
11: **return** Coordinator.address

In its simplest (and inefficient) form, an access control automaton can be segmented to the finest granularity to best preserve privacy. In this case, each automaton state is divided into one segment and deployed at one site. The algorithm demonstrates a recursive algorithm for finest-granularity automaton segmentation and deployment.

B. Query Segment Encryption Scheme

To protect user/data privacy that may be revealed by the queries, proposed query segment encryption scheme, which is a good instance that combines data avoidance principle (i.e. encrypting sensitive data) with multilateral security principle (i.e. multiple parties cooperate to take one task, while each party only holds one share of sensitive information). When an XPath query is being processed at a particular state in the NFA, the query content naturally splits into two parts: XPath steps that has been processed by NFA (accepted or rewritten), and XPath steps to be processed fig. 2.

Although the whole query will be forwarded to the coordinator who holds the next NFA state, NFA will only take the unprocessed steps as input. The idea of query segment encryption scheme is to encrypt the processed part of a query so that subsequent coordinators have only an incomplete view of the query content. For encryption, a trusted authority is needed for key distribution and management. In this scheme, this trustee is the super node.

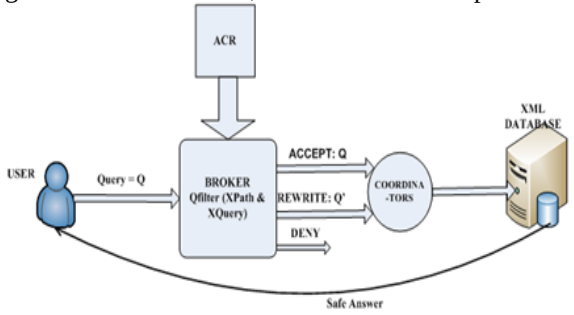


Fig. 2 Access Control Rules enforcement using NFA

C. Automatic scheme for dynamic site distribution

In this section we propose an automatic scheme that does dynamic site distribution depending on workload at each peer, trust level of each peer, and privacy conflicts between automaton segments is explained as follows.

a) Distributed load balancing algorithm

Proposed algorithm uses the concept of virtual servers previously proposed in [7]. A virtual server represents a peer in the DHT (Distributed Hash Table), that is, the storage of data items and routing happen at the virtual server level rather than at the physical node level. A physical node (broker) hosts one or more virtual servers. Load balancing is achieved by moving virtual servers from heavily loaded physical nodes to lightly loaded physical nodes. The basic idea of distributed load balancing algorithm is to store load information of the peer coordinators in a number of directories which periodically schedule reassignments of virtual servers to achieve better balance. Thus it essentially reduces the distributed load balancing problem to a centralized problem at each directory.

$$\mu = \frac{\sum Coordinators_n \; l_n}{\sum Coordinators_n \; c_n}$$

The load l_i of a coordinator i at a particular time, is the sum of the loads of the objects stored on that coordinator at that time. Each coordinator i, has a fixed capacity $c_i > 0$, which might represent, for example, available disk space, processor speed, or bandwidth. A coordinator's utilization ui is the fraction of its capacity that is used: $u_i = l_i/c_i$. The system utilization μ is the fraction of the system's total capacity which used:

Coordinator (time period T, threshold k_e)

- *Initialization:* Send (c_n, {l_{v1}, ……, l_{vm}}) to *broker()*
- *Emergency action:* when u_n jumps k_e
  1) Repeat up to twice while $u_n > k_e$
  2) $d \leftarrow$ *broker()*
  3) Send (c_n,{(l_{v1}, ……, l_{vm}}) to d
  4) *PerformTransfer* (v, n') for each transfer v→n' scheduled by d
- *Periodic action*: upon receipt of list of transfer from a virtual server
  1) *PerformTransfer* (v,n') for each transfer v→n'
  2) *Report* (c_n,{(l_{v1}, ……, l_{vm}}) to *broker()*

When $u_n > 1$, we say that coordinator n is overloaded; otherwise node i is said to be underloaded. When a coordinator n's utilization $u_n = l_n/c_n$ jumps above a parameterized emergency threshold ke, it immediately reports to the directory d which it last contacted, without waiting for d's next periodic balance. The directory then schedules immediate transfers from n to more lightly loaded coordinator s. More precisely, each coordinator n runs the following algorithm.

IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

In the above pseudocode, broker() selects two random virtual servers and returns the one to which fewer coordinators have reported since its last periodic balance. This reduces the imbalance in number of coordinators reporting to brokers. *PerformTransfer(v, n')* transfers virtual server v to node n' if it would not overload n', i.e. if $l_{n'} + l_v \leqslant c_{n'}$. Thus a transfer may be aborted if the directory scheduled a transfer based on outdated information. Each server runs the following algorithm.

Server (time period T, threshold k_e,k_p)

- *Initialization: I←{}*
- *Information receipt and emergency balancing:Upon receipt of J=(c_n, {l_{v1}, …., l_{vm}}) from coordinator n:*
  1) *I←I U J*
  2) *If $u_n > k_e$*
  3) *Reassignment←ReassignVS(I,k_e)*
  4) *Schedule transfers according to reassignment*
- *Periodic balancing: Every T seconds:*
  1) *reassignment ←ReassignSV (I,k_p)*
  2) *schedule transfers according to reassignment*
  3) *I←{}*

The subroutine *ReassignVS*, given a threshold k and the load information I reported to a directory, computes a reassignment of virtual servers from coordinators with utilization greater than k to those with utilization less than k.

$$|Positive| + |Negative| > T \quad (1)$$

$$\frac{Positive}{|Positive| + |Negative|} > P \quad (2)$$

Choice of parameters: Setting the emergency balancing threshold ke to 1 so that load will be moved off a coordinator when load increases above its capacity.

D. Reputation management scheme for Trust level on each peer

We propose a novel distributed reputation mechanism to detect malicious or unreliable coordinators in PPIB systems. Moreover, it considers how to effectively aggregate noisy (dishonest or inaccurate) ratings from independent or collusive coordinators using weighted majority techniques. A reputation system receives, aggregates, and provides feedbacks about participants' past behavior. The feedbacks help participants decide who to trust, encourage trustworthy behavior and deter dishonest peers from participation [5, 6]. The candidates are classified as following 3 levels by their file reputation: trustworthy, unknown, untrustworthy. The reputation level of each peer is decided by the following two conditions.

T and P are system-wide parameters. T is a threshold of minimum number of evaluations and P is a threshold of trust ratio. Peers that do not satisfy Conditions (1) are classified as unknown. Namely, they are considered as new, because we do not have enough reputation information on the peer so we can not decide whether the peer is trustworthy or not. The peers which satisfy Conditions (1) and (2) are classified as trustworthy. These peers have been evaluated enough and are considered as trustworthy by the evaluation. Whereas, the peers which satisfy Condition (1) but do not satisfy Condition (2) are classified as untrustworthy. Peer reputation managers only include trustworthy and unknown peers to the response message except untrustworthy peers. After a peer downloading and using the selected file from its peers, the peer evaluates its trustworthiness as positive or negative and sends the evaluation to the file reputation manager. If the file is trustworthy, the evaluation is assigned +1, if not it gets -1.

IV. Load Balancing, Privacy And Security Analysis

This section provides a comprehensive analysis of load distribution in PPIB system, privacy and security. We also carried out the performance analysis of PPIB system.

A. . *Load balancing analysis*

We performed load balancing analysis to evaluate the performance of load balancing scheme. Fig. 3 shows the load distribution with no load balancing. The load is not evenly distributed among the nodes: some of the nodes have very high load (the left side of the graph), and other nodes have just a small load or no load at all (the right side of the graph). Fig. 4 shows the load distribution in exactly the same system, but with our solution taking into account the request popularity, after the second dynamic run of the experiment. As shown in the graph, the highest load is decreased by half.

Moreover, the load in Fig. 3 tends to 0, while in Fig. 4 it remains almost constant (approximately from node 300), showing that most of the nodes have the same load.

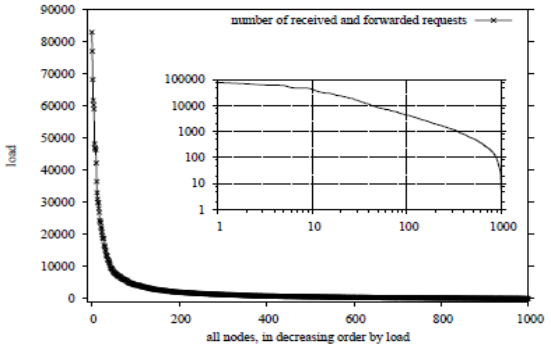


Fig. 3 Load distributions without load balancing

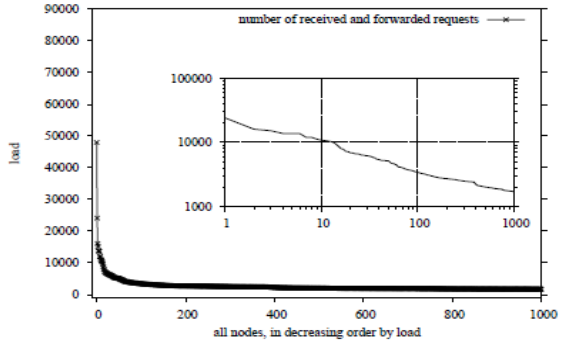


Fig. 4 Load distributions with load balancing

IPHV7I20040X
International Journal Of Advanced Research and Innovation -Vol.7, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

Our algorithm for dynamically updating the virtual servers of the nodes in the system shifts the load from the most loaded nodes to less loaded nodes, by having the less loaded nodes forward most of the traffic instead. This way, the highly loaded nodes will get rid of the traffic that they had to forward, and become less loaded.

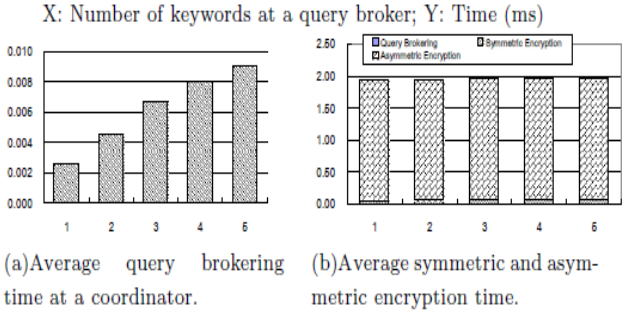
B. *Privacy and security analysis*

If the honest-but-curious assumption about the coordinators holds, the proposed automaton and query segmentation encryption schemes can guarantee that no intermediate coordinator views the complete query content while the coordinators fulfilling the query routing function. Risks exist only when one or multiple brokers are abused by the insiders or compromised by the outside adversaries. However, it can only learn the location of local broker from the captured packets since the content is encrypted and also the eavesdropper cannot distinguish the coordinators and the data servers.

Even though each coordinator can observe traffic on a path routed through it, nothing will be exposed to a single coordinator because (1) the sender viewable to it is always a brokering component; (2) the content of the query is incomplete due to query segment encryption; (3) the ACR and indexing information are also incomplete due to automaton segmentation; (4) the receiver viewable to it is likely to be another coordinator.

C. *Performance analysis*

End-to-end query processing time is defined as the time elapsed from the point when query arrives at the broker until to the point when safe answers are returned to the user. Query evaluation time highly depends on XML databases system, size of XML documents, and types of XML queries. We also evaluate the scalability of the PPIB system against complicity of ACR, the number of user queries, and data size (number of data objects and data servers). When the segmentation scheme is determined, the demand of coordinators is determined by the number of ACR segments, which is linear with the number of access control rules. Fig. 5(a) shows that query brokering time is at milliseconds level, and increases linearly with the number of keywords at a site. Fig. 5(b) shows that symmetric (Triple DES) and asymmetric encryption (RSA) time is at seconds level, and asymmetric encryption time dominates the total query processing time at a coordinator.



(a)Average query brokering time at a coordinator.

(b)Average symmetric and asymmetric encryption time.

Fig. 5 Average query processing time at a coordinator

When data volume increases (e.g. adding more data items into the online auction database), the number of indexing rules also increases. This results in increasing of number of indexing entries at leaf-coordinators. However, in PPIB, query indexing is implemented through hash tables, which is scalable. Thus, the system is scalable when data size increases.

V. Related Work

A number of information integration approaches had been developed to support business applications since 1980's. However, most of the early approaches focus on providing transparency and interoperability among heterogeneous system but neglect the needs for autonomy, scalability and privacy-preserving. Research areas such as information integration, Web search, peer-to-peer file sharing systems, and publish-subscribe systems provide partial solutions to the problem of large scale data sharing. Information integration seeks to provide an integrated view over large numbers of heterogeneous data sources by exploiting the semantic relationship between schemas of different sources [11, 13, 16]. It turns out that the PPIB approach will facilitate but is orthogonal to the information integration technology. On the other hand, Web search focuses on locating data sources with high precision and coverage [14, 18]. However, it only supports keyword queries with limited expressiveness.

Peer-to-peer systems are designed to share files and data sets (e.g. in collaborative science applications). Distributed hash table technology [5, 8] is adopted to locate replicas based on keyword queries. However, although these technologies have recently been extended to support range queries [7], the coarse granularity (e.g. files and documents) still makes them short of our expressiveness needs. Further, P2P file-sharing systems may not provide complete set of answers to a request while we need to locate all relevant data.

Addressing a conceptually dual problem, the XML publish/subscribe systems (e.g. [13, 16]) is probably the closely related technology to our proposed research: while we locate relevant data sources for a given query and route the query to these data sources, the pub/sub systems locate relevant consumers for a given document and route the document to these consumers.

VI. Conclusion

With little attention drawn on privacy of user, data, and metadata during the design stage, existing information brokering systems suffer from a spectrum of vulnerabilities associated with user privacy, data privacy, and metadata privacy. In this paper, PPIB, a new approach to preserve privacy and security in XML information brokering is proposed. Through an innovative automaton segmentation scheme, in-network access control, peer trust management and query segment encryption, PPIB integrates security enforcement and query forwarding while providing comprehensive privacy protection. An automatic scheme that does dynamic site distribution is designed. Several factors were considered in this scheme such as the

workload at each peer, trust level of each peer, and privacy conflicts between automaton segments. Future direction is to minimize (or even eliminate) the participation of the administrator node, who decides such issues as automaton segmentation granularity, site distribution and replication. A main goal is to make PPIB self-reconfigurable.

REFERENCES

[1] Fengjun Li, Bo Luo, Peng Liu, Anna Squicciarini, Dongwon Lee, and Chao-Hsien Chu. "Defending against Attribute-Correlation Attacks in Privacy-Aware Information Brokering", Proceedings of the 4th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), November, 2008. Orlando, FL.

[2] Noe Elisa , K. Suresh Babu **"** Survey on Protecting privacy and security in xml information brokering", IJC SMC, Vol.3 Issue.4, April- 2014,

[3] A. C. Snoeren, K. Conley, and D. K. Gifford, Mesh-based content routing using XML, in Proc. SOSP, 2001, pp. 160–173.

[4] Xiong, L., Liu, L., A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities, IEEE International Conference on E-Commerce (CEC), 2003

[5] Yu, B., Singh, M. P., A social mechanism of reputation management in electronic communities, 4th Intl Workshop on Cooperative Information Agents, 2000

[6] Frank Dabek, Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica, "Wide-area Cooperative Storage with CFS," in Proc. ACM SOSP, Banff, Canada, 2001.

[7] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. Communications of the ACM, 43(12):45-48, December 2000.

[8] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. Proceedings of the 2001 ACM SIGCOMM Conference, 2001.

[9] J. Kang and J. F. Naughton. On schema matching with opaque column names and data values. In SIGMOD, pages 205–216, 2003. [10] G. Koloniari and E. Pitoura. Content-based routing of path queries in peer-to-peer systems. In EDBT, 2004.

[11] G. Koloniari and E. Pitoura. Peer-to-peer management of xml data: issues and research challenges. SIGMOD Rec., 34(2):6–17, 2005.

[12] N. Koudas, M. Rabinovich, D. Srivastava, and T. Yu. Routing xml queries. In IEEE ICDE, page 844, 2004.

[13] F. Li, B. Luo, P. Liu, D. Lee, P. Mitra, W. Lee, and C. Chu. In-broker access control: Towards efficient end-to-end performance of information brokerage systems. In Proc. IEEE SUTC, 2006.

[14] H. Lu, J. X. Yu, G. Wang, S. Zheng, H. Jiang, G. Yu, and A. Zhou. What makes the differences: benchmarking xml database implementations. ACM Trans. Inter. Tech., 5(1):154–194, 2005.

[15] B. Luo, D. Lee, W.-C. Lee, and P. Liu. QFilter: Fine-grained run-time XML access control via NFA-based query rewriting. In ACM CIKM, Washington D.C., USA, nov 2004.

[16] I. Manolescu, D. Florescu, and D. Kossmann. Answering xml queries on heterogeneous data sources. In VLDB, pages 241–250, 2001.

[17] M. Murata, A. Tozawa, and M. Kudo. XML access control using static analysis. In ACM CCS, Washington D.C., 2003.

[18] S. Park, A. Khrabrov, D. M. Pennock, S. Lawrence, C. L. Giles, and L. H. Ungar. Static and dynamic analysis of the internet's susceptibility to faults and attacks. In IEEE Infocom, 2003.

[19] M. K. Reiter and A. D. Rubin. Crowds: anonymity for Web transactions. ACM Transactions on Information and System Security, 1(1):66–92, 1998.