International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2016): 79.57 | Impact Factor (2017): 7.296

Imposing Secure & Privacy-Maintaining in Distributed Information Sharing

G. Priyanka[1], Dr. P. Venkata Subba Reddy[2]

[1]M Tech Dept. of Computer Science and Engineering

[2]SVU College of Engineering SVU College of Engineering

Abstract: ***Today's organizations raise an increasing need for information sharing via on-demand access. Information brokering systems (IBSs) have been proposed to connect large-scale loosely federated data sources via a brokering overlay, in which the brokers make routing decisions to direct client queries to the requested data servers. Many existing IBSs assume that brokers are trusted and thus only adopt server-side access control for data confidentiality. However, privacy of data location and data consumer can still be inferred from metadata (such as query and access control rules) exchanged within the IBS, but little attention has been put on its protection. In this paper, we propose a novel approach to preserve privacy of multiple stakeholders involved in the information brokering process. We are among the first to formally definet wo privacy attacks, namely attribute-correlation attack and inference attack, and propose two countermeasure schemes automaton segmentation and query segment encryption to securely share the routing decision-making responsibility among a selected set of brokering servers. With comprehensive security analysis and experimental results, we show that our approach seamlessly integrates security and enforcement with query routing to provide system-wide security with insignificant overhead.***

Keywords: Access control, information sharing, privacy, information brokering system

1. Introduction

In many realms ranging from business to government agencies, there is an increasing need for inter organizational information sharing to facilitate extensive collaboration. While many efforts have been devoted to reconciling data heterogeneity and provide interoperability, the problem of balancing peer autonomy and system coalition is still challenging. Most of the existing systems work on two extremes of the spectrum, others or "pouring" data into a centralized repository becomes impractical. To address the need for autonomy, federated database technology has been proposed to manage locally stored data with a federated DBMS and provide unified data access. However, the centralized DBMS still introduces data heterogeneity, privacy, and trust issues. While being considered a solution between "sharing nothing" and "sharing everything", peer-to-peer information sharing framework essentially need to establish pair wise client-server relationships between each pair of peers, which is not scalable in large scale collaborative sharing.

In the context of sensitive data and autonomous data providers, a more practical and adaptable solution is to construct a data-centric overlay consisting of data sources and a set of brokers that make routing decisions based on the content of the queries. Such infrastructure builds up semantic-aware index mechanisms to route the queries based on their content, which allows users to submit queries without knowing data or server location. In our previous study such a distributed system providing data access through a set of brokers is referred to as *Information Brokering System* (IBS).

As shown in Fig applications atop IBS always involve some sort of consortium (e.g., RHIO) among a set of organizations. Databases of different organizations are connected through a set of brokers, and metadata are "pushed" to the *local brokers*, which further "advertise" (some of) the metadata to other brokers. Queries are sent to the local broker and routed according to the metadata until reaching the right data server(s). In this way, many information sources in different organizations are loosely federated to provide a unified, transparent, and on-demand data access.

While the IBS approach provides scalability and server autonomy, privacy concerns arise, as brokers are no longer assumed fully trustable - the broker functionality may be outsourced to third-party providers and thus vulnerable to be abused by insiders or compromised by outsiders.

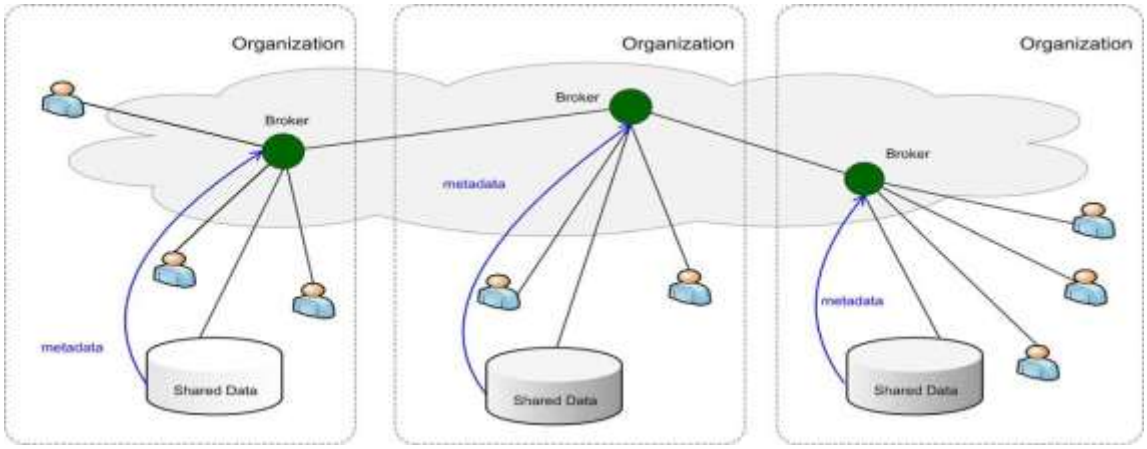


Figure: Architecture of IBS

2. Proposed Method

In this paper, we present a general solution to the privacy-preserving information sharing problem. First, to address the need for privacy protection, we propose a novel IBS, namely *Privacy Preserving Information Brokering* (PPIB). It is an overlay infrastructure consisting of two types of brokering components, *brokers* and *coordinators*. The brokers, acting as mix anonymizer, are mainly responsible for user authentication and query forwarding. The coordinators, concatenated in a tree structure, enforce access control and query routing based on the embedded nondeterministic finite automata—the *query brokering automata*. while providing integrated in-network access control and content-based query routing, the proposed IBS also ensures that a curious or corrupted coordinator is not capable to collect enough information to inferprivacy, such as "which data is being queried", "where certain data is located", or "what are the access control policies", etc. Experimental results show that PPIB provides comprehensive privacy protection for on-demand information brokering, with insignificant overhead and very good scalability.

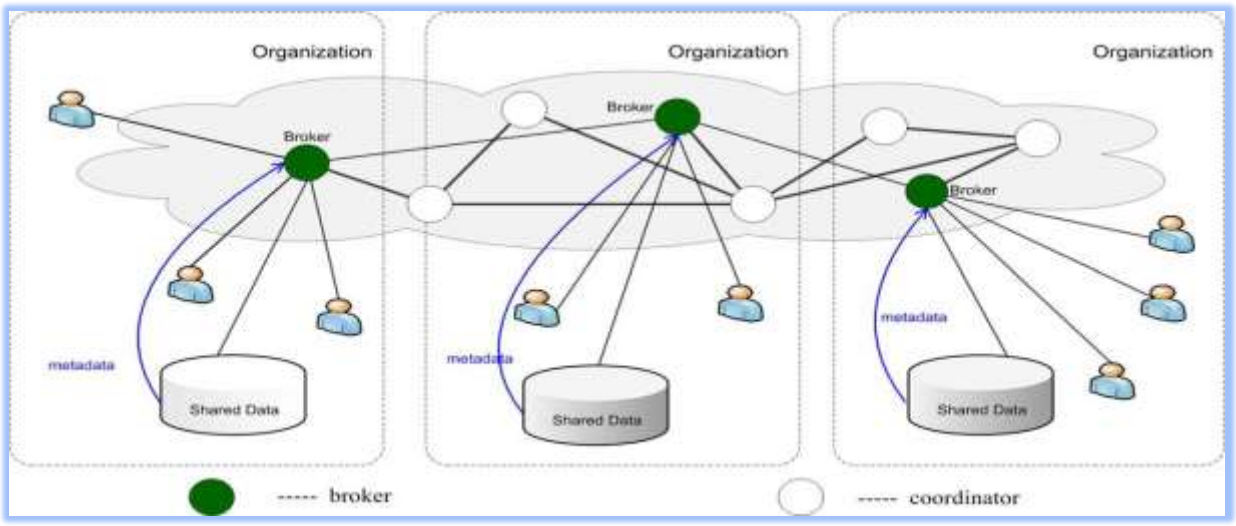


Figure: Architecture of PPIB

3. The Problem

Vulnerabilities and the Threat Model

In a typical information brokering scenario, there are **three** types of stakeholders, namely *data owners*, *data providers*, and *data requestors*. Each stakeholder has its own privacy: (1) the privacy of a data owner (e.g., a patient in RHIO) is the identifiable data and sensitive or personal information carried by this data (e.g., medical records). Data owners usually sign strict privacy agreements with data providers to prevent unauthorized use or disclosure. (2) Data providers store the collected data locally and create two types metadata, namely *routing metadata* and *access control metadata*, for data brokering. Both types of metadata are considered privacy of a data provider. (3) Data requestors may reveal identifiable or private information in the querying content.

For example, a query about AIDS treatment reveals the disease of the requestor. We adopt the *semi-honest* assumption for the brokers, and assume two types of adversaries, *external attackers* and *curious or corrupted brokering.*

Components: External attackers passively eavesdrop communication channels. Curious or corrupted brokering components, while following the protocols properly to fulfill brokering functions, try their best to infersensitive or private

information from the querying process. Privacy concerns arise when identifiable information is disseminated with no or poor disclosure control. For example, when data provider pushes routing and access control metadata to the local broker, a curious or corrupted broker learns *query content* and *query location* by intercepting a local query, *routing metadata* and *access control metadata* of local data servers and from other brokers, and *data location* from routing metadata it holds. Existing security mechanisms focusing on confidentiality and integrity cannot preserve privacy effectively. For instance, while data is protected over encrypted communication, external attackers still learn *query location* and *data location* from eavesdropping. Combining types of unintentionally disclosed information, the attacker could further infer the privacy of different stakeholders through *attribute-correlation attacks* and *inference attacks.*

Attribute-correlation attack: Predicates of an XML query describe conditions that often carry sensitive and private data (e.g., name, SSN, credit card number, etc.) If an attacker intercepts a query with multiple predicates or composite predicate expressions, the attacker can "correlate" the attributes in the predicates to infer sensitive information about data owner. This is known as the *attribute correlation attack.*

Inference attack: More severe privacy leak occurs when an attacker obtains more than one type of sensitive information and learns explicit or implicit knowledge about the stakeholders through association. By "implicit", we mean the attacker infers the fact by "guessing". For example, an attacker can guess the identity of a request or from her query location (e.g., IP address).Meanwhile, the identity of the data owner could be explicitly learned from query content (e.g., name or SSN). Attackers can also obtain publicly-available information to help his inference.

For example, if an attacker identifies that a data server is located at a cancer research center, he can tag the queries as "cancer-related".

Privacy preserving query Brokering Scheme

The QBroker approach has severe privacy vulnerability. If the QBroker is compromised or cannot be fully trusted, the privacy of both requestor and data owner is under risk. To tackle the problem, we present the PPIB infrastructure with two core schemes. We first explain the details of *automata segmentation* and *query segment encryption* schemes, and then describe the 4-phasequery brokering process in PPIB.

Automaton Segmentation

In the context of distributed information brokering, multiple organizations join a consortium and agree to share the data within the consortium. While different organizations may have different schemas, we assume a global schema exists by aligning and merging the local schemas. Thus, the access control rules and index rules for all the organizations can be crafted following the same shared schema and captured by a global automaton. The key idea of automaton segmentation scheme is to *logically* divide the global automaton into multiple in dependent yet connected segments, and *physically* distribute the segments onto different brokering components, known as coordinators.

Segmentation: The atomic unit in the segmentation is an NFA state of the original automaton. Each segment can hold one or several NFA states. We further define the *granularity level* to denote the greatest distance between any twoNFA states contained in one segment. Given a granularity level, for each segmentation, the next states will be divided into one segment with a probability. Obviously, with a larger granularity level, each segment will contain more NFA states, resulting in less segments and smaller end-to-end overhead in distributed query processing. However, a coarse partition is more likely to increase the privacy risk. The trade-off between the processing complexity and the degree of privacy should be considered in deciding the granularity level. As privacy protection is of the primary concern we need to reserve the logical connection between the segments after segmentation, we define the following

Heuristic segmentation rules: (1) NFA states in the same segment should be connected via parent-child links; (2) sibling NFA states should not be put in the same segment without their parent state; and (3) the "accept state" of the original global automaton should be put in separate segments. To ensure the segments are logically connected, we also make the last states of each segment as "dummy" accept states, with links pointing to the segments holding the child states of the original global automaton.

Algorithm: The automaton segmentation algorithm:
Input: Automaton State
Output: Segment Address:
for each symbol in **do**
addr = deploySegment
(S.StateTransTable(k).nextState)
DS= createDummyAcceptState()
DS.nextState←addr
S.StateTransTable(k).nextState ←DS
end for
Seg = CreateSegment()
Seg.addSegment()
Coordinator = getCoordinator()
Coordinator.assign Segment(Seg)
returnCoordinator. address

Deployment: We employ physical brokering servers, called *coordinators*, to store the logical segments. To reduce the number of needed coordinators, several segments can be deployed on the same coordinator using different port numbers. Therefore, the tuple uniquely identifies a segment. For the ease of presentation, we assume each coordinator only holds one segment in the rest of the article. After the deployment, the coordinators can be linked together according to the relative position of the segments they store, and thus form a tree structure. The coordinator holding the root state of the global automaton is the root of the coordinator tree and the coordinators holding theaccept states are the leaf nodes. Queries are processed along the paths of the coordinator tree in a similar way as they are processed by the global automaton: starting from the root coordinator, the first XPath step (token) of the query is compared with the tokens in the root coordinator. If

matched, the query will be sent to the next coordinator, and so on so forth, until it is accepted by a leaf coordinator and then forwarded to the data server specified by the outpointing link of the leaf coordinator. At any coordinator, if the input XPath step does not match the stored tokens, the query will be denied and dropped immediately.

Replication: Since all the queries are supposed to be processed first by the root coordinator, it becomes a single point of failure and a performance bottleneck. For robustness, we need to replicate the root coordinator as well as the coordinators at higher levels of the coordinator tree. We adopt the passive path replication strategy to create the replicas for the coordinators along the paths in the coordinator tree and let the *centralized authority* to create or revoke the replicas. The CA maintains a set of replicas for each coordinator, where the number of replicas is either a preset value or dynamically adjusted based on the average queries passing through that coordinator.

Handling the Predicates: In the original construction of NFA, a predicate table is attached to every child state of an NFA state. The predicate table stores predicate symbols if any, in the corresponding query XPath step. An empty symbol means no predicate. To handle the predicates, either from the query or from the ACR, the original strategy is *lookup-and-attach*. That is, if an XPath step in the query matches a child state in the state transition table predicate carried in that XPath step or predicate stored in the predicate table will be attached to the corresponding XPath step in the safe query. The real evaluation of the predicate is left to the data servers, which inevitably causes unnecessary communication and processing overhead if the predicate conditions conflict.

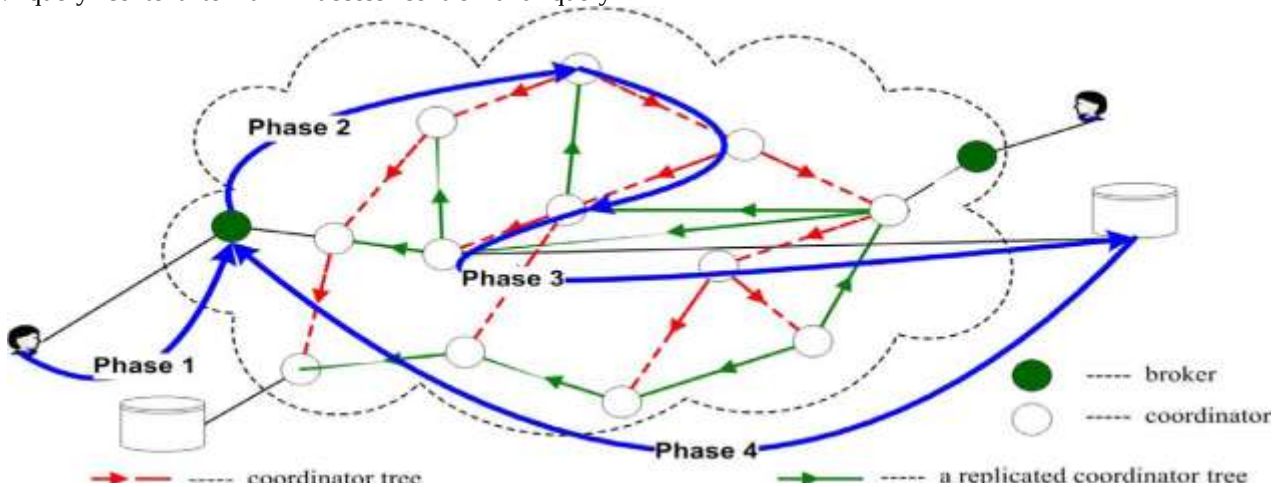
B. Query Segment Encryption

Informative hints can be learned from query content, so itis critical to hide the query from irrelevant brokering servers. However, in traditional brokering approaches, it is difficult, ifnot impossible, to do that, since brokering servers need to view query content to fulfill access control and query routing. Fortunately, the automaton segmentation scheme provides new opportunities to encrypt the query in pieces and only allows a coordinator to decrypt the pieces it is supposed to process. The query segment encryption scheme proposed in this work consists of the *pre encryption* and *post encryption* modules, and a special *commutative encryption* module for processing the double-slash XPath step in the query. Many directions are ahead for future research. First, at present, site distribution and load balancing in PPIB are conducted in an ad-hoc manner. Our next step of research is to design an automatic scheme that does dynamic site distribution. Several factors can be considered in the scheme such as the workload at each peer, trust level of each peer, and privacy conflicts between automaton segments. Designing a scheme that can strike a balance among these factors is a challenge. Second, we would like to quantify the level of privacy protection achieved by PPIB. Finally, we plan to minimize (or even eliminate) the participation of the administrator node, who decides such issues as automaton segmentation granularity. A main goal is to make PPIB self-reconfigurable.

4. The Overall PPIB Architecture

The architecture of PPIB is where users and data servers of multiple organizations are connected via a broker-coordinator overlay. The brokering process consists of four phases:

- **Phase 1:** To join the system, a user needs to authenticate himself to the local broker. After that, the user submits an XML query with each segment encrypted by the corresponding public level keys, and a unique session key. is encrypted with the public key of the data servers toencrypt the reply data.
- **Phase 2:** Besides authentication, the major task of the broker is metadata preparation: (1) it retrieves the of the authenticated user to attach to the encrypted query;(2) it creates a unique for each query and attaches and its own address to the query for data servers to return data.



- **Phase 3:** Upon receiving the encrypted query, the coordinators follow automata segmentation scheme and query segment encryption scheme to perform access control and query routing along the coordinator tree as described. At the leaf coordinator, all query segments should be processed and re-encrypted by the public key of the data server. If a query is denied access, a failure message with will be returned to the broker.
- **Phase 4:** In the final phase, the data server receives a safe query in an encrypted form. After decryption, the data

server evaluates the query and returns the data, encrypted by, to the broker that originates the query.

5. 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, we propose PPIB, a new approach to preserve privacy in XML information brokering. Through an innovative automaton segmentation scheme, in-network access control, and query segment encryption, PPIB integrates security enforcement and query forwarding while providing comprehensive privacy protection. Our analysis shows that it is very resistant to privacy attacks. End-to-end query processing performance and system scalability are also evaluated, and the results show that PPIB is efficient and scalable. Many directions are ahead for future research. First, at present, site distribution and load balancing in PPIB are conducted in an ad-hoc manner. Our next step of research is to design an automatic scheme that does dynamic site distribution. Several factors can be considered in the scheme such as the workload at each peer, trust level of each peer, and privacy conflicts between automaton segments. Designing a scheme that can strike a balance among these factors is a challenge. Second, we would like to quantify the level of privacy protection achieved by PPIB. Finally, we plan to minimize (or even eliminate) the participation of the administrator node, who decides such issues as automaton segmentation granularity. A main goal is to make PPIB self-reconfigurable.

References

[1] W. Bartschat, J. Burrington-Brown, S. Carey, J. Chen, S.Deming, andS. Durkin, “Surveying the RHIO landscape: A description of current{RHIO} models, with a focus on patient identification,” *J. AHIMA*,vol. 77, pp. 64A–64D, Jan. 2006.

[2] A. P. Sheth and J. A. Larson, “Federated database systems for managingdistributed, heterogeneous, and autonomous databases,” *ACM Comput. Surveys (CSUR)*, vol. 22, no. 3, pp. 183–236, 1990.

[3] L. M.Haas, E. T. Lin, andM.A. Roth, “Data integration through databasefederation,” *IBM Syst. J.*, vol. 41, no. 4, pp. 578–596, 2002.

[4] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, “CoolStreaming/DONet:A data-driven overlay network for efficient live media streaming,” in *Proc. IEEE INFOCOM*,Miami, FL, USA, 2005, vol. 3, pp. 2102–2111.

[5] A. C. Snoeren, K. Conley, and D. K. Gifford, “Mesh-based contentrouting using XML,” in *Proc. SOSP*, 2001, pp. 160–173.

[6] N. Koudas, M. Rabinovich, D. Srivastava, and T. Yu, “Routing XMLqueries,” in *Proc. ICDE’04*, 2004, p. 844.

[7] G. Koloniari and E. Pitoura, “Peer-to-peer management of XML data:Issues and research challenges,” *SIGMOD Rec.*, vol. 34, no. 2, pp. 6–17, 2005.

[8] M. Franklin, A. Halevy, and D. Maier, “From databases to dataspaces:A new abstraction for information management,” *SIGMOD Rec.*, vol. 34, no. 4, pp. 27–33, 2005.

[9] F. Li, B. Luo, P. Liu, D. Lee, P. Mitra,W. Lee, and C. Chu, “In-brokeraccess control: Towards efficient end-to-end performance ofinformationbrokerage systems,” in *Proc. IEEE SUTC*, Taichung, Taiwan,2006, pp. 252–259.

[10] F. Li, B. Luo, P. Liu, D. Lee, and C.-H. Chu, “Automaton segmentation:A new approach to preserve privacy in XML information brokering,”in *Proc. ACM CCS’07*, 2007, pp. 508–518.

[11] D. L. Chaum, “Untraceable electronic mail, return addresses, and digitalpseudonyms,” *Commun. ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[12] R. Agrawal, A. Evfimivski, and R. Srikant, “Information sharing acrossprivate databases,” in *Proc. 2003 ACM SIGMOD*, San Diego, CA,USA, 2003, pp. 86–97.

[13] M. Genesereth, A. Keller, and O.Duschka, “Informaster: An informationintegration system,” in *Proc. SIGMOD*, Tucson, AZ, USA, 1997.

[14] I. Manolescu, D. Florescu, and D. Kossmann, “Answering XMLqueries on heterogeneous data sources,” in *Proc. VLDB*, 2001, pp.241–250.

[15] J. Kang and J. F. Naughton, “On schemamatching with opaque columnnames and data values,” in *Proc. SIGMOD*, 2003, pp. 205–216.