Security Enforced Mediation Systems for Data Integration

Li Yang[1], Raimund K. Ege[2]
[1]Department of Computer Science and Electrical Engineering
University of Tennessee at Chattanooga, Chattanooga, TN 37403
Li-Yang@utc.edu
[2]School of Computing and Information Sciences
Florida International University, Miami, FL 33199
ege@cis.fiu.edu

Abstract. Mediation techniques support integrated query processing among heterogeneous databases. While such techniques help data sharing among different distributed sources, they increase the risk for data security, such as violating access control rule. Successful protection of information by an effective access control mechanism is a basic requirement for data integration among heterogeneous sources. How to protect the object with fine-granularity, how to support the access control specification at different points, and how to support the semantic heterogeneity in the access control become the major challenges in the data protection of the mediation system. Currently few existing approaches address all three security challenges inmediation system. With the purpose of solving the aforementioned challenges faced by the access control in mediation system, we present an architectural and practical solution to assure the data security in the process of data integration from different sources.

Keywords: security, mediation, heterogeneous, integration
(Received June 07, 2005 / Accepted July 10, 2005)

1 Introduction

Enterprises tend to store and represent their data using a variety of data models and schemata, while users want to access data in an integrated and consistent fashion. Moreover, enterprises need to use data from the sources that do not follow well-structured scams, nor fit into object-oriented or relational database models. So, there is a pressing need to provide *integrated access* to information stored in heterogeneous databases and data sources. Mediators are typically employed in a situation where the client data model does not coincide with the data model of the potential data sources. A mediator provides a mapping of complex models to enable interoperability between clients and sources. Modern integration systems follow the mediation paradigm presented by Wiederhold [1].

Examples for mediators are TSIMMIS [2], Information Manifold [3],HERMES [4], DISCO [5], Garlic [6] and MMM [7]. The goal of such systems is to permit the exploitation of several independent data sources as if they were a single source, with a single global schema. We use global as view, because the use of a global view coincides with the desire of many to integrate information from a variety of sources and present it in an integrated view for a user. Since we map each local source to the global independently, changes in a particular source affect only the mapping of that source to the global. Thus, the approach is scalable and can be applied to an environment such as the Web where scalability is paramount. As far as integration of heterogeneous data sources is concerned, our work features an integrating of data sources based on semantic mapping.

The integration of heterogeneous data sources must also consider security concerns. It is important to consider whether the client has the necessary credentials to access the data before the integration can occurs. Dawson [8] associated wrappers with a mapping between the source's security lattice and other lattices, but failed to support object granularity. Damiani et al. [9] exploited XML's own capabilities, allowed the definition and enforcement of access restrictions directly on the structure and content of the documents. Several approaches haven been presented in the literature to enhance the RBAC model with additional features to allow extended policy specification frameworks. X-GTRBAC [10] is an XML-based GTRBAC [11] policy specification language that enforces enterprise-wide access control. X-GTRBAC supports the enforcement of enterprise policy across distributed domains, ensuring secure content-based access to enterprise resources at all user levels with the consideration of context conditions. The OASIS model for active security presented in Bacon et al. [12] and Strembeck et al. [13]. address the context-aware access control requirements within large scale systems.

Some recent work has been reported in the area of XML-based security and context-aware access control. Two representative security specifications emerging from the industrial community are the Security Assertions Markup Language (SAML) [14] and the XML Access Control Language (XACML) [15]. SAML is an authentication mechanism, and XACML is designed for Web-based authorization. XACML specification is based on an extension of XML to define an access control specification that supports user credentials and context-based privilege assignments. However, they disregard the pressing need for inter-operation and information sharing among databases, especially the semantic-related heterogeneous data sources.

In this paper we show how to support the secure data integration in the context of a distributed mediation system where the security policy specification at different data sources may differ. A reliable technology that enforces seamless data security into *integrated query processing* over heterogeneous information sources is presented.

The paper is organized as follows: Section 2 discusses a flexible mediation architecture designed to integrate information from heterogeneous data sources. Section 3 illustrated the secure data integration strategy in the mediation framework. Section 4 explains the implementation with several screen shots. Section 5 presents our concluding remarks.

2 A Mediation Architecture

Ege et al.[16] propose a mediator architecture for the information integration from heterogeneous databases. Such system can integrate semantically heterogeneous data with the knowledge of the capability from participating sources.

2.1 Data Representation

The primary motivation for mediation technology is to provide support for a broad spectrum of heterogeneous data which are available in different formats. A sound solution to the data integration task requires a clean abstraction of the different formats: any data must be mapped to an *exchange model* from which it is therefore accessible without the use of specific software.

We use XML, a standard for semi-structured data representation and transmission, as the exchange model to provide interoperability among heterogeneous data sources. Wrapper techniques are employed to transform the heterogeneous data models to homogeneous XML model. A W3C proposal is used to identify the internal components of an XML document, namely, the XPath language [17].

We keep at a simplified level the description of the language that expresses in XPath. The W3C [18] contains the complete specification of the language.

2.2 Architecture of Mediator

The proposed mediator architecture is to handle requests from a user (as in Figure 1). These mediators will play intermediate roles between users and data sources, and these mediators help the users to establish streams to and from the heterogeneous data sources.

The framework features three layers: presence, integration and homogenization/connector. The upper level is the presence layer that makes the data source seem ever-present to the user and communicates directly with the user. The presence layer is responsible to translate the heterogeneous request from user to XML format, extract the data type of request represented by XML schema, and translate the response from the XML format into the original user request format. The presence layer makes it feasible that the mediation architecture handles the request from any kind of devices or in any kind of formats by the two-way format translation. Therefore the work of underlying layers is encapsulated.

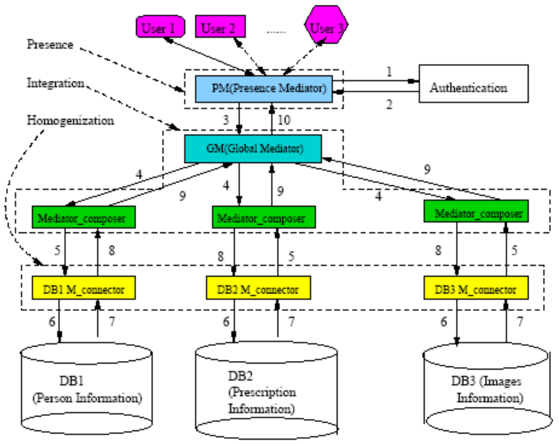


Figure 1. A three-layered architecture of mediation

The middle integration layer resolves the schema differences between the user needs and the source availability by schema mapping [19]. The entities in the integration layer are the "Mediator_composers" who are able to decompose the schema if necessary and locate the destination data source for a specific schema. Upon every request from a client, one global mediator will be elected from the "Mediator_composers" based on the availability. Before the election, the relationship between the mediator_composers is peer-to-peer. After the election, the global mediator distributes the query to the relevant mediator_composers and composed the retrieved information from the mediator_composers. This process of global mediator election dynamically determines the hierarchical structure in the integration layer for each request. This procedure makes the architecture more adaptive to both the network capability and mediator load, and then more efficient for the multimedia data operation, i.e. streaming, than a fixed architecture. The bottom level homogenization layer makes heterogeneous data sources appear to have a unifying XML schema.

With the assumption that the query to the databases happen more frequently than the update of the databases, the mediation systems include two phases as follows. This paper we put the main efforts on the second run-time query answering phase.

Initialization/Preparation phase

1. Each source has its own schema type and advertise to the global level. A global schema type was generated by the methods proposed in current related works [20].
2. Each source maps to the global schema type. By the mapping, each source generates it capability against global schema type. That is each source pushes its contribution to global mediator.
3. Global mediator maintains the sources capability lookup for all sources.

Run-time Query Answering phase

The following steps refer to the message passing in the architecture of mediation showed in Figure~\ref[16].

1. User log in to the system and messages are sent to the *Authentication* module.
2. *Authentication* module returns the secure global view (SGV) to the current user based on his/her credentials. The details for SGV will be discussed in section 3.2.
3. User poses query against the returned secure global view (or schema type).
4. Global mediator distributes the query to the relevant mediator_composers that can answer the queries.
5. Relevant mediator_composers translated queries represented by the global view (schema type), into queries represented by the source views (schema types).

6-7. The relevant sources execute the queries, and return the results to the corresponding mediator_composers.

8-9. The mediator_composers translate the results represented by the source views (schema type) into the ones represented by the global view (schema type).

10. .Global mediator integrates the results and returns to the user.

2.3 Mediator Specification

In order to achieve unified identification of mediation object in the above architecture, the transformation from heterogeneous data model to homogeneous data model is performed by wrappers. But the semantic heterogeneity still exists in both terminology and structural aspects. A *mediation system* is defined and how to resolve the semantic gap among heterogeneous sources is elucidated in this section. With the purpose of integrating data from heterogeneous databases, a mediation system includes three parts: a *global (mediated) view (schema type)*, a set of *source views (schemata type)* and *mapping relation*. A *global view (schema type)* is tree type whose labels are terms of a global vocabulary, distinct from source schema type used in the labels defining local data schema. The global schema vocabulary has been chosen to unify the local vocabulary and represent a specific domain of interest. A user query is formulated in terms of the global schema; to execute the query; the system translates it into sub-queries expressed in terms of the local schemata, sends the sub-queries to the local data sources, retrieves the results, and combines them into the final result provided to the user. Figure 2 shows a mapping example between the global view and source view.

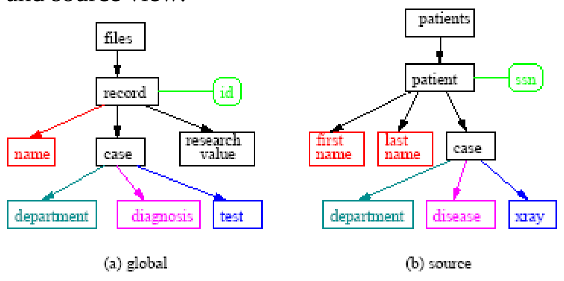


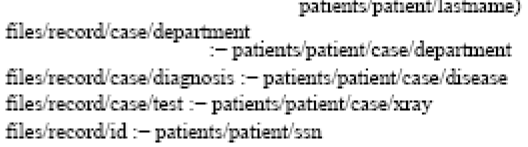
Figure 2. Mapping between global view and source view

Definition 2.1 (Mediation System) A *mediation system M* is a triple ($G, \{L_i\}, \{M_i\}$), where G is a global schema, L_i is a set of *n* source schemata, and M_i is a set of *n* source to global mappings, such that for each source schema L_i there is a mapping M_i from L_i to *G*, $1 \le i \le n$.

The above definition is inherited from [21] that defines the mapping between the path in global schema and the path in source schema. We enhanced their definition by the mapping function that includes complex operations [22], like *merge*, *concatenate*, thus improving the capacity for semantic gap resolution. We use tree to represent both global and source schema in *M*. A tree includes set of *leaf object set O* and a set of *path relationship R*. Therefore a *schema element* is either a leaf object set or a relationship set. For a schema *H*, which is either a source schema or a global schema, we let Σ_H denote the union of *O* and *R*.

A source-to-global mapping M_i for source schema S_i with respect to a global schema *G* is a function $f_i(\Sigma_{S_i}) \to \Sigma_G$. Intuitively, a source-to-global mapping M_i represents inter-schema correspondences between a source schema S_i and a global schema *G*. A source-to-global mapping between the two schemata includes a semantic correspondence.

For instance, mediator 1 is used to specify the mapping relationship between the global schema and the source information database (source). It declares that the concatenation of values in *firstname* and *lastname* in source semantically corresponds to the values of *name* in the global. And *patients/patient/case/disease* in source corresponds to *files/record/case/diagnosis* in the global.



```
files/record/name  :− merge(patients/patient/firstname,
                                   patients/patient/lastname)
files/record/case/department
                        :− patients/patient/case/department
files/record/case/diagnosis :− patients/patient/case/disease
files/record/case/test :− patients/patient/case/xray
files/record/id :− patients/patient/ssn
```

Figure 3. An example of mediator specification

User's queries are issued with the vocabulary defined in the global schema. To be executable, queries must be translated into queries expressed in terms of the labels used in the source data trees, which are labels appearing in the tree types defining their schema. Query translation depends on a *mapping relation* which connects paths of a global schema type to paths of local schema type. For instance, in order to retrieve *diagnosis* and *test* data from source by *name* the query against global schema is:

```
SELECT diagnosis, test
FROM global
WHERE name = "John Smith"
```

is translated into the query against the source schema as:

```
SELECT disease, xray
FROM source
WHERE firstname = "John"
    AND lastname = "Smith"
```

3 Security Enhanced Mediation Systems

In the mediation system, particular security specification requirements occur: Firstly, multiple heterogeneous data sources participate in the system, and the autonomy of security policy specification at each source level is allowed. That means security policy specification should support security policy specification at different points, i.e., global level, source level. Secondly, fine-grained security policy specification should be supported. Thirdly, semantic heterogeneity of the objects at the global level and source levels should be considered.

3.1 Security Policy Specification

In order to enforce the security policies, the policies should be interpreted and stored in the policy base. A W3C proposal is used to identify the internal components of an XML document, namely, the XPath language [17]. We assume that access control at each source and application is regulated by an Role-based Access Control (RBAC) model [23]. The consideration of RBAC models is justified because it simplifies the security management of permissions, provides the flexibility and meets the natural needs of the organizations and enterprise management. Moreover, DAC and MAC can be configured to RBAC [24]. Main components of RBAC96 model includes users, roles, role-hierarchy, permissions, user-role assignments and permission-role assignment [25]. The preliminary work for security policy specification in mediation systems has been done in [26]. This work proposes the security enforcement for the mediation framework with the knowledge of the security policy specification and the mediator specification. The view generation algorithm proposed in this work provides the seamless way to enforce the security in the mediation framework. This work is further solidified and verified by the Java-based implementation.

Definition 3.1 (Mediation Object (MO)) A *mediation object* is an XML schema or schema component(s), patterned by an XPath expression based on global schema at global level; and patterned by XPath based on source schema at source level, under the assumption that XML object (instance) is defined by XML schema.

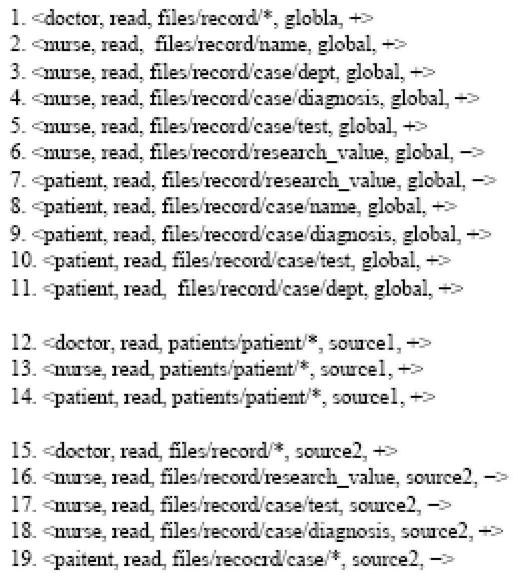
Definition 3.2 (Authorization request) <*subj, op, obj*> refers to a query as to whether or not the subject *subj* is permitted to perform the operation *op* on the object *obj*.

Definition 3.3 (Role assignment) conforms to user-to-role assignment in basic RBAC model, and is modeled as <*subj, role*> referring to subject-to-role assignment.

Definition 3.4 (Access policy) is a permission assignment defined as (<*role*>, <*op*>, <obj>, <*level*>, <*sign*>), where $role \in ROLES$, $op \in OPERATION$, $obj \in OBJECT$, $level \in \{global, source\}$ and $sign \in \{+, -\}$ denotes grant and deny respectively.

In a hospital medical records management system, we assume there are two data sources. Source 1 has the *source* schema type and Source 2 has the *global* schema type from figure 2 respectively. So the mapping between global view and source 1 are specified by mediator 1 in figure 3. The mapping between the global view and source 2 is trivial. The hospital system has access control policy specified at both global level and the source levels. An example of an access control policy specification is shown in Figure 4:

Rules 1 - 11 are specified at the global level, rules 12 - 14 are specified at source 1 and rules 15-19 are specified at source 2, which provide the flexibility of policy specification. Rule 1 stipulates that at global level *doctor* can see all the patient records and the sub-elements of the patients' records. Rule 2-5 stipulate that at global level *nurses* are allowed to view the *name*, *dept*, *diagnosis* and *test* nodes in patient records. Whereas rule 6 stipulates that at global level *nurses* are not allowed to view the *research_value* node in patient records. Rule 7 stipulates that at global level *patients* are not allowed to view the *research_value* node in patient records. Rule 8 - 11 stipulate that at global level *patients* are allowed to view the name, *diagnosis*, *test* node in patient records.



1. <doctor, read, files/record/*, globla, +>
2. <nurse, read, files/record/name, global, +>
3. <nurse, read, files/record/case/dept, global, +>
4. <nurse, read, files/record/case/diagnosis, global, +>
5. <nurse, read, files/record/case/test, global, +>
6. <nurse, read, files/record/research_value, global, −>
7. <patient, read, files/record/research_value, global, −>
8. <patient, read, files/record/case/name, global, +>
9. <patient, read, files/record/case/diagnosis, global, +>
10. <patient, read, files/record/case/test, global, +>
11. <patient, read, files/record/case/dept, global, +>

12. <doctor, read, patients/patient/*, source1, +>
13. <nurse, read, patients/patient/*, source1, +>
14. <patient, read, patients/patient/*, source1, +>

15. <doctor, read, files/record/*, source2, +>
16. <nurse, read, files/record/research_value, source2, −>
17. <nurse, read, files/record/case/test, source2, −>
18. <nurse, read, files/record/case/diagnosis, source2, +>
19. <paitent, read, files/recocrd/case/*, source2, −>

Figure 4. Security policy specification

At source 1, rule 12 stipulates that at source 1 *doctors* are allowed to view all the sub-nodes under the path *patients/patient* node in patient records. Rule 13 stipulates that at source 1 *nurses* are allowed to view all the sub-nodes under the path *patients/patient* node in patient records. So do the *patients* in source 1 denoted by rule 14. At source 2 that may contain sensitive data, rule 15 stipulates that at source 2 *doctors* can see all the patient records. Rule 16 -18 stipulate that at source 2, the *nurses* are not allowed to view the *research_value*, *test* and *diagnosis* nodes in the patient records. Rule 19 stipulate that at source 1, the *patients* are not allowed to view any sub-nodes under the path *files/record/case* in the patient record.

Suppose we are in the hospital system, the participating users can play the doctor, nurse, family member and patient roles. User-to-role assignment denoted by {<*Martha, doctor*>, <*Mary, nurse*>}. Example 1 and 2 intuitively illustrate the access decision with the observation of both global and local security policy specification.

Example 1: If a user *Martha* requests to view the *research_value* node of the patient record. According to role assignment, *Martha* is assigned the *doctor* role. The applicable rules are rule 1 from global level and rule 16 from source 2. Based on the rules the access are granted because both global and source 2 policies allow the access.

Example 2: If a user *Mary* requests to view the *diagnosis* node of the patient record. According to role assignment, *Mary* is assigned the nurse role. So the applicable rules are rule 4 from global level, rule 13 from source 1 and rule 18 from source 2 respectively. That means the access to the *diagnosis* node is allowed at global level and source 1, but denied from the source 2. For the security reason, we apply the "deny takes precedence" rule, and the access of nurse to the *diagnosis* node is allowed in the source 1 and denied in the source 2.

3.2 View Generation Algorithm

After the user authentication, the mediation systems will return the *secure global view (SGV)* based on his/her role. And the user is safe to pose query against the SGV and retrieve the integrated data from the mediation systems. We introduce the term *secure global view* that denotes part of the global schema type and is visible or authorized to the current *subject* based on his/her profile and authorization policies that state a subject can (or cannot) access an element or (set of them). A user could login in to the authentication module with the his/her id or on behalf certain roles [25]. We call them *subject* in general. The idea of view computing [9] is to create and maintain a separate view for each subject (user, role) who is authorized to access a specific portion of global schema type (XML structured in nature). The view contains exactly the set of data nodes that the user is authorized to access. After the view is constructed, during the run time, users can simply run their queries against the views without worrying about security enforcements. The view of a subject on the global schema type depends on the access permissions and denials specified by the authorization and their priorities.

E. Damiani used *tree labeling process* [9] technique to compute the view under the assumption that the global and local levels employ the same set of vocabulary to represent the data. However, in our mediation systems the security policies specified by different vocabularies at the global level and the local levels. In order to generate the secure global view for the users playing different roles, the objects represented by different vocabularies in local security policies need to be unified towards global vocabulary based the semantic mappings. In the view generation algorithm, one node is allowed to be accessed by the certain subject if the access is authorized by global policy specification and by at least one available source. One node is not allowed to be access by the requesting subject if the access is denied either by the global policy specification or by all the available source policy specification.

The view generation algorithm in Algorithm 1 computes the secure global view for specific subject with the input of global schema type and security authorization rules. And Figure 5 demonstrates the secure global view for *nurse* after the application of the secure generation algorithm based on both the global security policy and local policies. The *research_value* node is removed because a *nurse* is not to access at the global level. The *department and diagnosis* nodes are allowed at global level, source 1 and source 2 for the *nurse*. The *test* node is allowed for the *nurse* at the global level and source 1 where it is represented as *xray*.

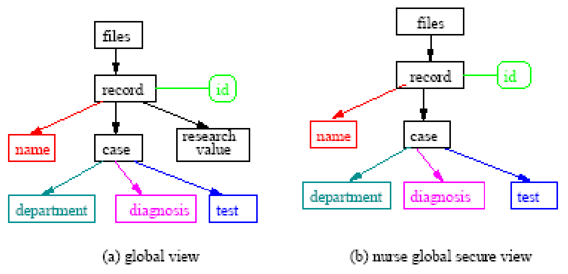


Figure 5. Global view and secure global view

```
input: a requester (role) and global XML document global.xml
output: the secure global view returned to the requester
repository: mediator specification M[i], global and local policy base, i.e., PA

/*parse and generate instance tree t rooted in root of global.xml*/
    1.  root = parse(global.xml)
    /*check the object's nodes permission based on policy base*/
    2.  function LabelTree(role, root)
    3.  {
    /*label the node based on the global policies*/
    4.  Boolean globalLabel = Label(role, root, "global", PA)
    /*obtain all the mapping objects in the available sources based on the mediation specification*/
    5.  Vector mappingObjectsVector = new Vector()
    6.  for each mediator M[i]
```

```
7.      mappingObject = mapping(root, M[i])
8.      mappingObject.sourceID = i
9.      mappingObjectVector.add(mappingObject)
10.  endfor
11.  for(Enumeration m = mappingObjectVector.elements(), m.hasMoreElements())
/*label the node based on the mapping objects and applicable local policies*/
12.     Boolean localLabel = false
13.     localLabel = localLabel || Label(role, m.nextElement.getNode(), m.nextElement.getSourceID(), PA)
14.  endfor
15.  if(globalLabel || localLabel)          // not allowed by either global or local policies
16.     mark root node "-"
17.  if(root not leaf)
18.     for each-subtree sub rooted in the root of global.xml
19.         LabelTree(role, sub)
20.     endfor
21.  endif
22.  boolean function Label(role, node, sourceID, PA)
23.  {
/*find the matched policies in policy base*/
24.  Vector policies = getAccessPolicy(role, sourceID, PA)
25.  for(Enumeration policy = policies.elements(), policy.hasMoreElements())
26.     String currentPolicy = (String)policy.nextElement()
27.     String sign = getSign(role, node, currentPolicy)
28.     if(sign.equals("-"))
29.         return false; break;}
30.     else   return true
31.  endfor
```

Algorithm 1. View generation algorithm for mediation systems

3.3 Secure Data Integration

This section explains how to protect data based on the *SGVs*, and how to evaluate the query based on the semantic mappings. After the user logs in to the mediation system, the system will return the SGV based on his/her credentials. Those secure views are calculated offline in section 3.2. Then the user's queries are issued with the vocabulary defined in the *secure global view*. The global

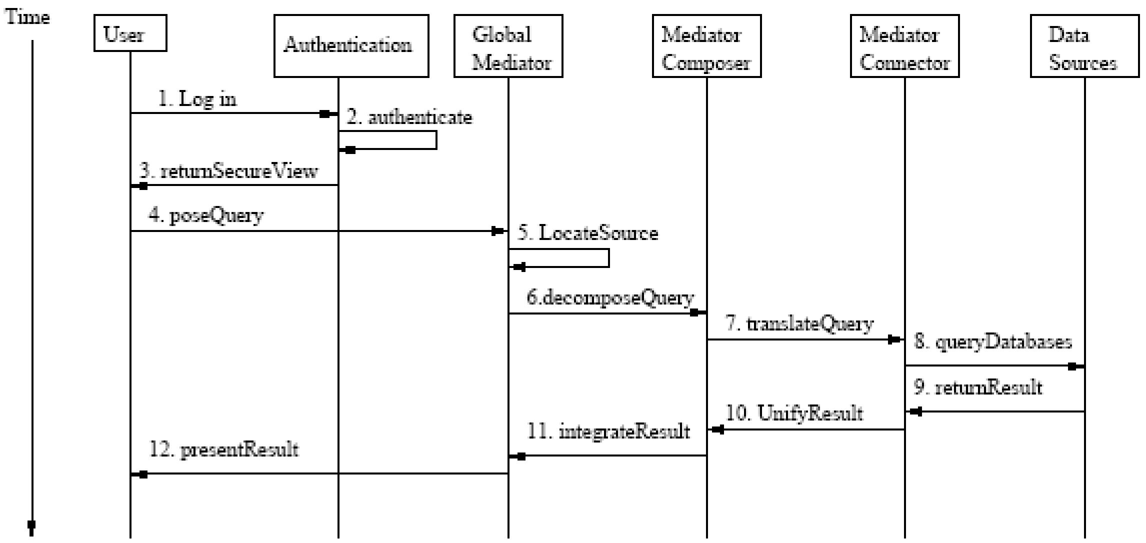


Figure 6. Runtime secure data integration

mediator locates and decomposes the queries to the relevant mediator_composers that connect to the relevant data sources. In our example if a doctor

logs in to the system and poses a query against the *doctor view*, the source s_1 and s_2 become the relevant sources and the mediator 1 and mediator 2 become the relevant mediator_composers because those two mediators provide the mapping specification from the *doctor view* to the *source views*. For instance, the *doctor* poses the query *Q* to retrieve the patient's *record* with the knowledge of patient's *name*. We call the paths either used as input condition, i.e. *files/record/name*, or used as output query result, i.e. *files/record* the essential paths in the global query tree. To be executable, queries must be translated into queries expressed in terms of vocabulary in the source schema type. Query translation relies on the semantic mapping relations between the global schema type and the local schema types. In the above example, after the global mediator receives the query Q, it detects that mediator 1 and mediator 2 are relevant mediator_composers. These two relevant mediator_composers will translate Q into the queries expressed in the local vocabulary based on the semantic mappings.

In the query translation process, the paths in the global tree query are searched in the mediator specifications and translated to the local tree queries. With the knowledge stored in global mediator that the information from source 1 and source 2 can join on patient's *id*, *id* becomes *essential path* also. So the essential paths in *Q* are: *files/record/id*, *files/record/name*, *files/record/case/department*, *files/record/research_value*, *files/record/case/diagnosis*, and *files/record/case/department*, and the corresponding in mediator 1 is *patients/patient/id*, *patients/patient/firstname*, *patients/patient/lastname*, *patients/patient/case/department*, *patients/patient/case/disease*, and *patients/patient/case/xray*. Correspondingly, the query essential paths set in source 2 are same as that in query *Q*. The complexity of the translation depends on the number of the essential paths in the global query tree and the number of semantic mappings in the mediator specifications. Finally, the answers from the source 1 and source 2 are integrated (joined) on patient *id* and presented to the user. Note, the patient *id* here can assure that the information from the same person in the real world is integrated. Figure 6 illustrates the above secure query processing step by step.

By the above strategy, the data security for heterogeneous sources is seamlessly enforced in our mediation framework.

4 Implementation

We implemented our approach to a security enhanced mediation system for data integration among the heterogeneous data sources in the context of a medical records management system. We employ heterogeneous XML documents in our implementation to simulate the heterogeneous data sources, and use the JDOM package to parse and process these documents. As different users log in to our system, different views are presented to him/her that corresponds to the current security situation. The users are safe to pose queries against his/her secure view. Figure 8 shows an example of such a view that applies to the doctor role.

The semantic mapping relationship between the global view and the local views are represented by the XML path mapping and stored in the relational databases. The views visible to the users are patterned by the global view and the mediators will locate the relevant data sources that can answer the query and translate the user query into the ones recognized by the local sources. The left hand side window in Figure 7 shows the visible view for a doctor, and the right hand side window demonstrates a doctor's secure query result from the heterogeneous data sources when he/she wants to read the medical records of patients where "*//record*" is an XPath patterned query. The query results, i.e., *doctor name*, *nurse name*, *diagnosis* are separated by double stars in the right hand side window: it means that those results are retrieved from different heterogeneous data sources and that the result is presented to the user according to the global schema type.

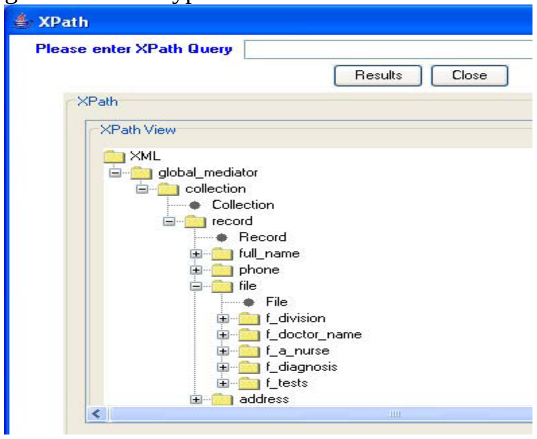
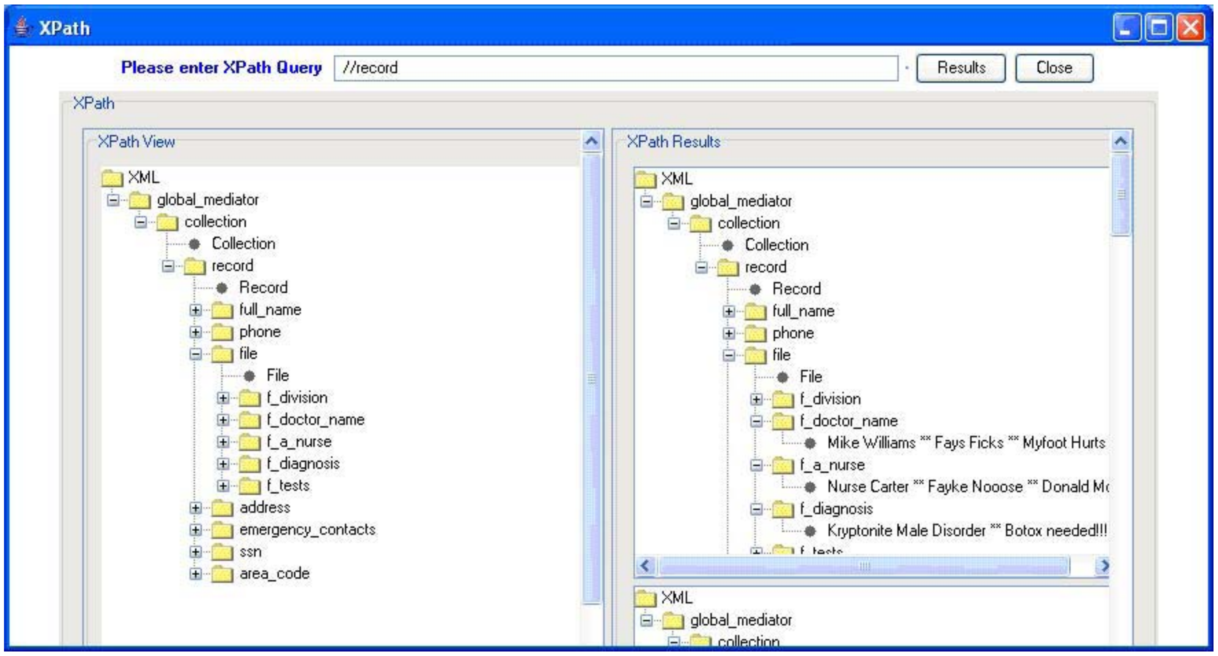


Figure 7. Secure view for doctor role

5 Conclusion

In this paper we presented how to integrate information from heterogeneous sources based on the semantic mapping between the global schema type and the source schema types. Moreover, the flexible security enforcement in mediation systems is supported by the off-line view of computing for different users/roles. Our implementation incorporates advanced security capabilities into our dynamic mediation framework.



```
//record
```

```
XML
  global_mediator
    collection
      Collection
      record
        Record
        full_name
        phone
        file
          File
          f_division
          f_doctor_name
          f_a_nurse
          f_diagnosis
          f_tests
        address
        emergency_contacts
        ssn
        area_code
```

```
XML
  global_mediator
    collection
      Collection
      record
        Record
        full_name
        phone
        file
          File
          f_division
          f_doctor_name
            Mike Williams ** Fays Ficks ** Myfoot Hurts
          f_a_nurse
            Nurse Carter ** Fayke Nooose ** Donald Mc
          f_diagnosis
            Kryptonite Male Disorder ** Botox needed!!!
          f_tests
XML
  global_mediator
    collection
```

Figure 8. Secure integration results for doctor role

Reference

[1] Wiederhold, G., *Mediators in architecture of future information systems.* IEEE Computer, v.25, p. 38-49, 1992.

[2] Papakonstantinou, Y., H. Garcia-Molina, and J. Widom. *Object exchange across heterogeneous information sources*. Proceedings of ICDE Conference, 1995.

[3] Kirk, T., Levy, A. Y., Sagiv, Y., Srivastava, D. *The Information Manifold*, Information Gathering from Heterogeneous, Distributed Environments, Stanford University, Stanford, CA, 1995.

[4] Adali, S. and R. Emery. *A uniform framework for integrating knowledge in heterogeneous knowledge systems*, Proceedings of ICDE conference, 1995.

[5] Tomasic, A., L. Raschid, and P. Valduriez. *Scaling heterogeneous databases and the design of Disco*, International Conference on Distributed Computing Systems, 1996.

[6] Roth, M.T. and P. Schwarz. *Don't scrap it, wrap it! A wrapper architecture for legacy sources*, Proceedings of *23th VLDB Conference*, Athens, Greece, 1997.

[7] Altenschmidt, C., Biskup, J. Freitag, J. Sprick, B. *Weakly constraining multimedia types based on a type embedding ordering*, Proceedings of 4th International Workshop on Multimedia Information Systems, Berlin: Springer-Verlag, 1998.

[8] Dawson, S., S. Qian, and P. Samarati, *Providing security and interoperation of heterogeneous systems*, Distributed and Parallel Databases, v. 8(1), p. 119-145, 2000.

[9] Damiani, E., Vimercati, S. D. C. di Paraboschi, S. Samarati, P., *A fine-grained access control system for XML documents*, ACM Transaction Information System Security, v.5(2), p. 169-202, 2002

[10] Bhatti, R., Ghafoor, A., Bertino, E., Joshi, J.B.D. *X-GTRBAC: an XML-based policy specification framework and architecture for enterprise-wide access control,* ACM Transactions Information System Security, p. 187-227, 2005.

[11] Joshi, J.B.D., Bertino, E., Latif, U., Ghafoor, A., A *Generalized Temporal Role-Based Access Control Model*, IEEE Transactions on Knowledge and Data Engineering, v.17(1), p. 4-23, 2005.

[12] Bacon, J., K. Moody, and W. Yao, *A model of OASIS role-based access control and its support for active security,* ACM Transactions on Information System and Security, p. 492-540, 2002.

[13] Strembeck, M. and G. Neumann, *An integrated approach to engineer and enforce context constraints in RBAC environments*, ACM Transactions on Information System and Security, p. 392-427, 2004.

[14] *XML Coverages: SAML 1.0 specification.* http://xml.coverpages.org/ni2003-05-27-b.html, 2003.

[15] *XML Coverages: XACML 1.0 specification.* http://xml.coverpages.org/ni2003-02-11-a.html, 2003.

[16] Ege, R.K., Yang, L., Kharma, Q. ,Ni, X.. *Three-layered mediator architecture based on DHT*, Proceedings of International Symposium on Parallel Architecture, Algorithm and Networks. Hong Kong, China: IEEE press, 2004.

[17] Clark, J. *XML path language (XPath) Version 1.0*, World Wide Web Consortium (W3C), 1999.

[18] *World Wide Web Consortium (W3C),* XML path language (XPath), 2002.

[19] Altenschmidt, C. and J. Biskup. *Explicit representation of constrained schema mapping for mediated data integration*, Proceedings of *2nd Workshop on Databases in Networked Information Systems (DNIS)*, 2002.

[20] Bergamaschi, S., Castano, S., Beneventano, D., Vincini, M., *Semantic integration of heterogeneous information sources,* Special Issue on Intelligent Information Integration, Data & Knowledge Engineering, v.36(1), p. 215-249, 2001.

[21] Delobel, C. and M.C. Rousset. *A uniform approach for querying large tree-structured data through a mediated schema*, Proceedings of Foundations of Models for Information Integration Workshop (FMII), 2001.

[22] Xu, L. and D.W. Embley, *Using schema mapping to facilitate data integration.* 2003.

[23] Sandhu, R.S. Coyne, E. J., Feinstein, H. L., Youman, C. E., *Role-based access control models.* IEEE Computer, v.29(2): p. 38--47, 1996.

[24] Osborn, S., R. Sandhu, and Q. Munawer, *Configuring role-based access control to enforce mandatory and discretionary access control policies,* ACM Transaction. Information System Security, v.3(2): p. 85--106, 2000.

[25] Sandhu, R., D. Ferraiolo, and R. Kuhn. *The NIST model for Role-based Access Control: Towards a Unified Standard*, Proceedings of 5th ACM Workshop on Role Based Access Control, 2000.

[26] Yang, L., R.K. Ege, and H. Yu. *Security specification and enforcement in heterogeneous databases*, Proceedings of 20th Annual ACM Symposium on Applied Computing, Computer Security Track, 2005.