Institut Eurécom
Department of Corporate Communications
2229, route des Crêtes
B.P. 193
06904 Sophia-Antipolis
FRANCE

Research Report RR-09-227

Content-Driven Secure and Selective XML Dissemination

March 20th, 2009
Last update March 20th, 2009

Mohammad Ashiqur Rahaman, and Yves Roudier

Tel : (+33) 4 93 00 81 00
Fax : (+33) 4 93 00 82 00
Email : {mohammad.rahaman,yves.roudier}@eurecom.fr

[1] Institut Eurécom's research is partially supported by its industrial members: BMW Group Research & Technology - BMW Group Company, Bouygues Télécom, Cisco Systems, France Télécom, Hitachi Europe, SFR, Sharp, STMicroelectronics, Swisscom, Thales.

Content-Driven Secure and Selective XML Dissemination

Mohammad Ashiqur Rahaman, and Yves Roudier

Abstract

Collaborating on complex XML data structures is a non-trivial task in domains such as the public sector, healthcare or engineering. Specifically, providing scalable XML content dissemination services in a selective and secure fashion is a challenging task. This paper proposes a publish/subscribe infrastructure to disseminate enterprise XML content utilizing document semantics. Our approach relies on the dissemination of XML documents based on their content, as described by concepts that form the basis for an interoperable description of XML documents. This infrastructure leverages our earlier parsing [1] scheme for efficient processing of enterprise XML and at the same time for protecting the integrity and confidentiality of XML content during dissemination.

Index Terms

Enterprise XML, Ontology, Dissemination

Contents

List of Figures

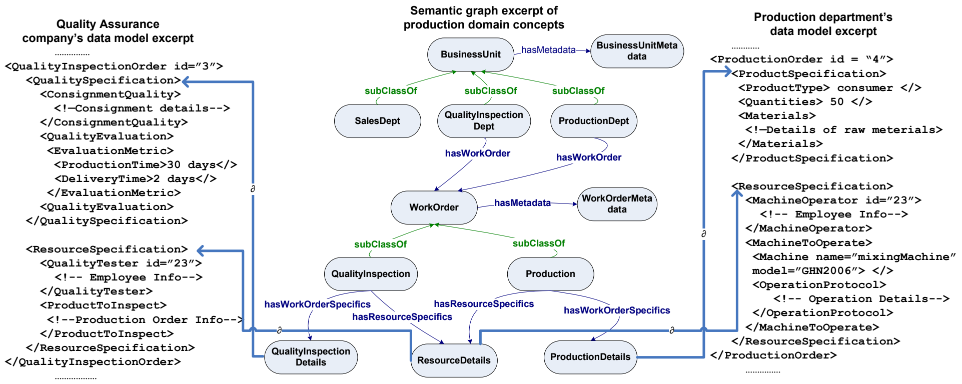
1 Introduction

Due to the rise of cross-organizational communication based on common XML processing standards such as XML schema, XSL, SOAP, WSDL or BPEL, an increasing number of business-related XML documents is exchanged through the internet. These documents may have a considerable size, a complex structure and rich semantics which we consider typical for enterprise applications such as enterprise resource planning (ERP) or supply chain management (SCM). We term such documents as 'Enterprise XML'. Today's cross-organizational communication mostly relies on a client-server interaction model which is not tailored for all business cases such as multiple government agencies (e.g. ministries) providing information to an anonymous audience (e.g. citizens). Certainly, a publish/subscribe interaction model is suitable for such business cases. Even though certain standards for publish/subscribe interaction exist (e.g. WS-Notification [2]), their adoption falls short.

Many organizations that participate in such processes develop proprietary XML schemas to address individual needs, for instance, a particular data model, business process or organizational structure. Such schemas may contain business critical information that needs to be protected. In addition, enterprise XML might be routed by untrusted intermediaries and through insecure communication channels which also asks for content confidentiality and integrity.

Regarding the actual service interface, communication parties need to agree on a certain data model (schema) which may evolve over time (e.g. due to changes in one party's organization, for instance after a merger); existing data exchanges with peers should however be maintained. We claim that, although data models may differ from one organization to another or vary with time, the underlying semantics (represented in the XML business document by XML fragments) constitute a more stable and interoperable interface between organizations. Semantic web languages like RDF [3] and OWL [4] make it possible to share an ontology describing a conceptual data model, independently from XML data structures yet can still be mapped to instances of XML schemas. To address security requirements, authorization policies on the semantic level, i.e. ontology, should be supported. Besides, with the disseminated enterprise XML being large, requires efficient XML processing to utilize memory and computation time. Such a secure exchange of documents can be achieved through the separate encryption of each document node with a secret that is computed in distributed fashion by the publishers and subscribers. In this approach, an authorization on a concept triggers a secret key computation resulting into granting authorizations to multiple XML documents or portions thereof.

Previous academic research effort [5–15] targets some of these mentioned issues, namely confidentiality and integrity of documents in a client-server environment. However, government or industry use cases that include multiple information providers and consumers which do not necessarily know each other a priori, require a different dissemination approach. We propose a publish/subscribe based document dissemination system where document producers publish documents and



```
     ...............
<QualityInspectionOrder id="3">
   <QualitySpecification>
     <ConsignmentQuality>
       <!—Consignment details-->
     </ConsignmentQuality>
     <QualityEvaluation>
      <EvaluationMetric>
       <ProductionTime>30 days</>
       <DeliveryTime>2 days</>
      </EvaluationMetric>
     <QualityEvaluation>
   </QualitySpecification>

   <ResourceSpecification>
     <QualityTester id="23">
       <!-- Employee Info-->
     </QualityTester>
     <ProductToInspect>
      <!--Production Order Info-->
     </ProductToInspect>
   </ResourceSpecification>
</QualityInspectionOrder>
     ..................
```

```
    ............
<ProductionOrder id = "4">
   <ProductSpecification>
    <ProductType> consumer </>
    <Quantities> 50 </>
    <Materials>
     <!—Details of raw meterials>
    </Materials>
   </ProductSpecification>

   <ResourceSpecification>
     <MachineOperator id="23">
       <!-- Employee Info-->
     </MachineOperator>
     <MachineToOperate>
      <Machine name="mixingMachine"
      model="GHN2006"> </>
      <OperationProtocol>
         <!-- Operation Details-->
      </OperationProtocol>
     </MachineToOperate>
   </ResourceSpecification>
</ProductionOrder>
     ...............
```

Figure 1: A semantic graph of work order document concepts in a production domain. The 'Production' and 'Quality-inspection' work order concepts are mapped to the corresponding XML data model excerpts using a mapping relation.

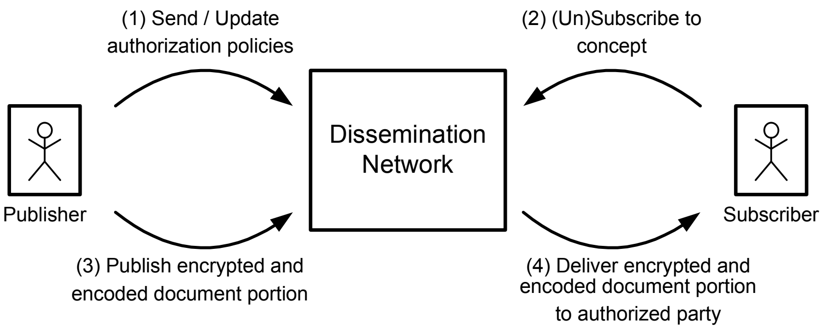

(1) Send / Update authorization policies
(2) (Un)Subscribe to concept
Dissemination Network
Publisher
Subscriber
(3) Publish encrypted and encoded document portion
(4) Deliver encrypted and encoded document portion to authorized party

Figure 2: A publish/subscribe model of content-driven XML dissemination.

subscribers consume those independently of each other. The dissemination system is an intermediate layer composed of disseminators and selectively routes XML content along the dissemination topology and performs selective delivery, i.e. filtering, to the authorized subscribers according to the authorization policy of the publishers. Important requirements we address in our systems are:

1. *Confidentiality of information:* Access to documents shall be limited to authorized communication partners, i.e. the respective publisher and authorized subscribers.

   This is addressed by (a) the encryption of published XML information, supported by a distributed key management and (b) an ontology-based authorization scheme that supports access control and dissemination on semantic level.

2. *Integrity of information:* Documents must not be altered during transit.

   This is achieved by an encoding method [1] that allows subscribers to verify integrity of the received documents.

3. *Confidentiality of schema information:* An information schema (e.g. XML schema) represents a valuable asset in itself (e.g. information about organizational structures or business processes can be derived from the schema) and as such needs to be confidential.

   This is fulfilled by using ontologies as the interface among organizations instead of concrete schemas.

4. *Scalability:* The system must scale for document producers, document subscribers, the number of documents and the size of documents.

   This is addressed by designing a publish/subscribe based dissemination system with (a) an efficient routing algorithm used by the disseminators and (b) an efficient XML processing minimizing memory consumption and processing time compared to typical XML processing.

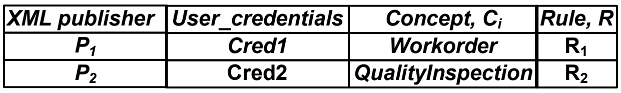
Some of these requirements have been addressed in our previous work: in [1], we developed an ontology-based XML content distribution system focusing on automation of policy evaluation, inference rules and granting access of selective XML content to authorized users. In [1], we also developed an efficient XML parsing technique called *encrypted breadth first order labeling* (EBOL) to protect the confidentiality and integrity of the XML content and its semantics. In [16], a distributed key management technique is described to enable users of different trust boundaries to compute a secret key independently and thus exchange confidential documents only among the peers.

This paper focuses on the publish/subscribe infrastructure of XML content disseminators. Section 2 describes a brief solution and preliminaries of the publish/subscribe model which is discussed in detail in section 3. This includes a subscription protocol, publishing of XML content, selective XML routing and unsubscription issues. A relative discussion of the solution is provided in section 4. A relative comparison with the related work is given in section 5 and section 6 finally concludes the paper with future work.

2 Background

System Overview. The dissemination system distinguishes three different actors (Fig 2): (a) document producers who publish encrypted and encoded XML document portions that represent ontology concepts, (b) document subscribers who receive these XML document portions and (c) disseminators that are part of a distributed dissemination network and who manage subscriptions, enforce authorization policies on behalf of the publishers and realize the actual content transmission from publishers to subscribers.

The EBOL-based encoding [1] (see Appendix for further information) of the original XML document portions ensures that the content is only readable to the respective publisher and authorized subscribers who do not need to know each other.



XML publisher	*User_credentials*	*Concept, C_i*	*Rule, R*
P_1	*Cred1*	*Workorder*	R_1
P_2	Cred2	*QualityInspection*	R_2

Figure 3: Ontology-based authorization policy.

In this context, we leverage our previous work [16] that allows a group of users to compute a common key independently of each other. The number of participants (both publishers and subscribers) as well as the number and size of the XML document portions depend on the actual use case.

The dissemination (thus publication/subscription) is based on a shared ontology that models all relevant business domain entities including their relationships and to which every system participant agrees to (Fig 1 sketches some concepts, e.g. Work Order, Production and Quality Inspection, of a production process domain). The definition (or nomination) of a shared ontology is the prerequisite for any interaction between the system actors. We assume a large scale system of large number of publishers and subscribers. As such the content publishers can neither serve content to each user nor authenticate each of them.

As soon as the common domain ontology has been agreed on, the system actors interact as follows (see Fig 2): (1) Prior to the first document publication, a publisher needs to provide authorization policies that determine user authorizations and which will be enforced by the dissemination network. These policies can be flexible and may evolve as described in our previous work [1]. (2) An end user sends a subscription request with valid credentials (e.g. public key certificate) to a disseminator which in turn evaluates associated policies (provided by the publishers) and trigger the computation of a secret key for every group of subscribers to the same concept [16]. Unsubscription might be done on user request or be forced by the disseminator (e.g. if the user credentials expired or if authorization policies changed). (3) The publisher of a given XML document encodes each XML document portion with its conceptual information [1], encrypts the nodes in a stipulated granularity with the secret key computed for the concept, and sends those to the disseminators. (4) Disseminators follow a dissemination protocol described in section 3 in order to route the encoded XML document portions selectively to all authorized subscribers. The recipient verifies the received XML content by decoding the EBOL-based encoding, both semantically and structurally, in a verification phase which is detailed in our previous work [1]. Eventually a subscriber may receive multiple document portions, possibly with different XML vocabularies, for a concept he is authorized to. Essentially, the user needs an adaption mechanism with its own XML data model (discussed in section 3).

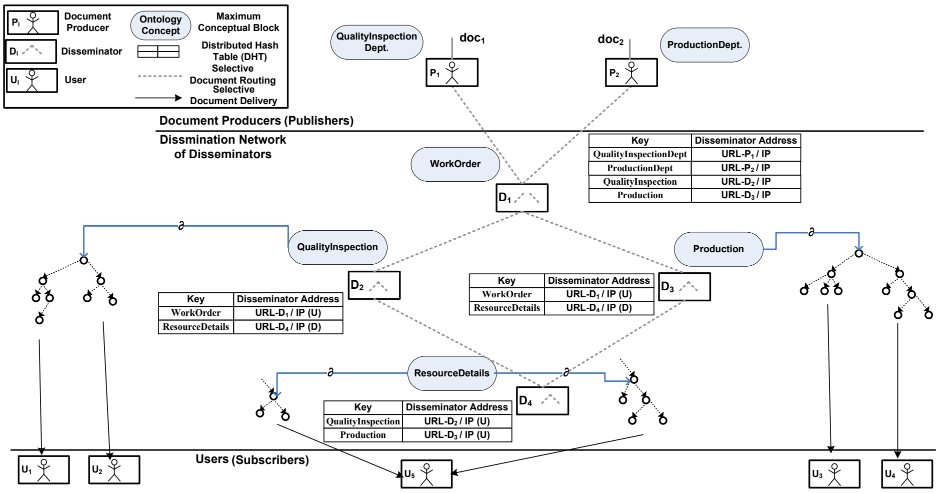
In the following an ontology concept and associated lemma is provided that we utilize for building an ontology-based dissemination topology and optimization thereof (section 3).

Preliminaries. A concept C_i is an abstraction of a physical or logical thing and can be communicated among peers. An ontology is a shared set of such concepts in a domain. The ontology is defined primarily by the notions of *class*, *subclass*, *properties* representing concepts and their relationship using OWL [4] as illustrated in Fig 1.

Definition 1 *Concept Containment: Let $\mathcal{C}$ be the collection of all concepts and $C_i, C_j \in \mathcal{C}$. If there exists a sub class hierarchy from C_i to C_j denoted as $C_i \Rightarrow, ..., \Rightarrow C_j$ then C_i contains C_j and noted as $C_i \preceq C_j$.*

Example: Fig 1 shows a collection of concepts $\mathcal{C} = \{BusinessUnit, BusinessUnitMetadata, SalesDept, QualityInspectionDept, ProductionDept, WorkOrder, WorkOrderMetadata, QualityInspection, Production, QualityInspectionDetails, ResourceDetails, ProductionDetails\}$in a production hall domain. $WorkOrder$ contains $QualityInspection$ and $Production$, i.e. $WorkOrder \preceq QualityInspection$, $WorkOrder \preceq Production$. □

Definition 2 *Maximum Conceptual Block: Let C_i be a concept. The maximum conceptual block for C_i is the set of all concepts that are reachable by following a succession of concept containment from C_i.*



Key	Disseminator Address
QualityInspectionDept	URL-P$_1$ / IP
ProductionDept	URL-P$_2$ / IP
QualityInspection	URL-D$_2$ / IP
Production	URL-D$_3$ / IP

Key	Disseminator Address
WorkOrder	URL-D$_1$ / IP (U)
ResourceDetails	URL-D$_4$ / IP (D)

Key	Disseminator Address
WorkOrder	URL-D$_1$ / IP (U)
ResourceDetails	URL-D$_4$ / IP (D)

Key	Disseminator Address
QualityInspection	URL-D$_2$ / IP (U)
Production	URL-D$_3$ / IP (U)

Figure 4: Publish/Subscribe infrastructure for XML content distribution.

Lemma 1 *Maximum conceptual blocks are always monotonically decreasing.*

Proof: *Let C_i and C_j be two concepts such that C_i contains C_j; let M_i^c and M_j^c be the maximum conceptual blocks for C_i and C_j respectively. As C_i contains C_j the number of classes reachable from C_i is always more than that of C_j. So $M_j^c \subset M_i^c$. Transitively for any concept C_k, if C_j contains C_k i.e. $C_j \preceq C_k$ then $M_k^c \subset M_j^c$.*

An ontology concept and its sub-class hierarchical path are mapped to disjoint document portions d_i. The mapping is illustrated by the following example.

Example: In Fig 1, the concepts $ProductionDetails$ and $ResourceDetails$, identified by the paths over the semantic graph $BusinessUnit.ProductionDept.hasWorkOrder.Workorder.Production.ProductionDetails$ and $...Production.ResourceDetails$ are mapped to the document portions rooted at `<ProductSpecification>` and `<ResourceSpecification>` of the production department's XML data model. In the quality assurance company's data model, the concepts $ResourceDetails$ and $QualityInspectionDetails$

are mapped to the document portions rooted at `<ResourceSpecification>` and `<QualitySpecification>` respectively. □

Ontology-based Authorization. We describe an ontology-based authorization policy as a set of explicit rules as illustrated in Fig 3 which shows an example of a policy specified by two XML content publishers P_1(i.e. Production department) and P_2 (i.e. Quality assurance company). R_1 and R_2 are inference rules which for instance, R_1 for the user with credential $Cred1$ is: if a user is allowed to access the concept $WorkOrder$ then he is also allowed to access to all the contained concepts of $WorkOrder$. R_2 for the user with credential $Cred2$ is: a user is allowed to access the concept $QualityInspection$ if he has access to the concept $ResourceDetails$.

Efficient XML Processing and XML security. XML documents being a tree structured data consume memory space not only for the XML nodes but also for their hierarchy and sibling relationship. For example, an empty element `<e/>` (4 bytes for the source file) could easily take 200 bytes of tree storage using Java [17]. Moreover the disseminated XML node must also carry its semantic information (e.g. concept, location, depth) for its later verification and thus make these content confidential and vulnerable to integrity violation. In [1], we developed a special XML parsing technique called *encrypted breadth first order labeling* (EBOL) that allows one level of XML nodes to be stored in a FIFO queue while parsing in breadth first order without storing any hierarchy information (e.g. child, parent, sibling) of nodes and thus saving considerable memory space. In [18] we showed that the required space of this parsing technique is proportional to $O(ms_x s_e)$ which would have been $2d$ times of this proportion if the full document trees and their normalized trees would be in memory. m and d are the number of nodes in a level and the number of levels (i.e. depth) of the document tree respectively. The average space required for a node x and its EBOL identifier are s_x and s_e respectively. The encoding method of [1] is such that it hides the XML content and its semantics from the disseminators yet they are able to perform selective routing and delivery of XML content.

3 Publish/Subscribe Dissemination Model

Document semantics as represented by ontology concepts, being the only interface among multiple organizations, drive the XML content dissemination scheme. This section describes the setup of a publish/subscribe scheme, roles of the involved entities, subscription protocol, ontology-based dissemination topology (see Fig 4), selective routing and delivery and unsubscription of concepts. In the following, a document, d, identified by doc_{id} (e.g. URI, RDF) is a set of parsed XML nodes and a document portion d_i is a subtree rooted at node i of d.

3.1 Setup

Disseminators. A disseminator is a piece of software running either in intra or inter enterprise boundaries and thus is distributed. It is able to route documents through an ontology-based dissemination topology. As mentioned before, disseminators perform the ontology-based authorization check on behalf of publishers and are also trusted to manage the subscriptions. Disseminators, however, should not be able to read document content. Any malicious disseminator may violate content and structure integrity during routing.

Disseminator Initialization. The intermediate layer of disseminators of Fig 4 is introduced to ensure scalable and efficient document dissemination. Each disseminator (including publishers) maintains a distributed hash table where the key fields and the values are the concepts and references (i.e. URL/IP) of the disseminators respectively. The ordering of the key fields are determined using the $maximum\ conceptual\ block$ as follows: we assign each $maximum\ conceptual\ block$ in the key fields in monotonically decreasing fashion and assign the reference addresses of the next disseminators in the value fields for each such key.

Let D_i, D_j be two disseminators that disseminate two $maximum\ conceptual\ blok$s represented by concepts C_i, C_j respectively. If $C_i \preceq C_j$ holds then D_i is an uplink disseminator of D_j and D_j is a downlink disseminator of D_i. As such, D_i hosts XML document portions associated to C_i and all its contained concepts C_j. D_i puts C_j (i.e. downlink disseminators' D_j references) such that $C_i \preceq C_j$ and C_k (i.e. uplink disseminators' D_k references) such that $C_k \preceq C_i$ as its key and value fields in the hash table respectively.

3.2 Dissemination Topology

The disseminators form a topology of a directed acyclic graph (DAG) based on concept containment where document publishers comprise multiple starting points (roots) in the dissemination. Fig 4 shows such a dissemination topology.

Let C_i, C_j be two concepts identified by $O_1.C_i$ and $O_2.C_j$ and $C_i \preceq C_j$, then $O_{i\in[1,2]}$ are path expressions in the concept containment that leads to the disseminators D_i and D_j respectively. Let D_k be any disseminator reachable from D_i by following a dissemination path $D_i \rightarrow, ..., \rightarrow D_k$. C_i is the maximum conceptual block at D_i if and only if D_i or any disseminator D_k has registered only the users who have authorizations to the concepts C_i or any of its contained concept C_j. Consequently D_i can deliver the encoded and encrypted XML nodes to a set of subscribers such that none of them has access or has subscribed to a concept $C_m \in \mathcal{C}$, where $C_m \preceq C_i$. In effect, the disseminator D_i disseminates only the mapped XML nodes of C_i or any C_j such that $C_i \preceq C_j$. In Fig 4, the disseminator D_3 has 'Production' as the maximum conceptual block for which user 3 and user 4 have collectively registered.

3.3 Subscription Protocol

In this section, we elaborate on the subscription protocol which makes use of two functions and an encoding element 'content signature'. A content signature is comprised of XML node's structural and conceptual information (see appendix). The function $auth\_list(u)$ returns a set of content signatures which is used by a subscriber u as a means to verify the received XML content. The function $served\_list(d)$ returns the set of concepts represented by the $maximum$ $conceptual\ block$ in the distributed hash table of the disseminator d.

1. User u sends a subscription request (together with its credentials) for a set of concepts to a disseminator D_r. Upon receipt of a subscription request from a user u, D_r determines the authorizations of the user u as defined in section 2. The content signatures of all authorized concepts, i.e. $auth\_list(u)$, to which the user has access to if at all are returned.

2. If all authorized concepts of $auth\_list(u)$, are contained in the list of served concepts of the disseminator D_r, denoted as in the $served\_list(D_r)$, then D_r registers the user, u, successfully as an authorized user and the protocol ends.

3. Otherwise the content signatures received in the 1st step include at least one concept, $C_k \in auth\_list(u)$ such that $C_k \not\in served\_list(D_r)$. If the requested concept contains D_r's served concepts, i.e. $C_k \preceq \forall C_i \in served\_list(D_r)$, then D_r sends the request to the uplink disseminators. Otherwise, D_r sends the request to the downlink disseminators.

4. After receiving a request for C_k from D_r, a disseminator D_m checks if there exists either a $C_k \in served\_list(D_m)$ of step 3 or a concept containment relation $(C_m \in served\_list(D_m)) \preceq C_k$. If so, D_m returns the mapped encoded and encrypted XML nodes of C_k with success as a response to D_r, else D_m recursively performs the same step three for other disseminators in its hash table.

5. After receiving the responses possibly from several disseminators, the disseminator D_r selects a sending disseminator using a selection policy described below, updates its $served\_list(D_r)$ by adding the newly received content and notifies the disseminators accordingly. Now, the disseminator D_r is able to register the user u and sends a response stating a successful subscription back to it.

Selection policy: A selection policy is chosen based on the notion of concept distance aiming at optimizing the hops required to route the content: Let C_i, C_j be two concepts identified by $O_1.C_i, O_2.C_j$, where $O_{i\in[1,2]}$ are two path expressions and $|O_i|$ denotes the number of hops required as entailed by concept containment. Then concept distance between C_i and C_j is defined as $||O_i| - |O_j||$. The receiving disseminator chooses the sending disseminator with the smallest concept distance from itself.

Secret key computation: A disseminator determines groups of authorized subscribers for the same concept and sends their credentials to the publishers following the dissemination path. For each such group of subscribers, publishers trigger the independent secret key computation by sending the necessary cryptographic elements (details in [16]).

3.4 Publishing

Publishers take charge of individual XML document data models and policies over it. They also define a mapping relation of the ontology concepts into their individual data model as shown in Fig 1. For a new instance of a document, a publisher encodes and encrypts the mapped document portions and finally sends those to its downlink disseminators.

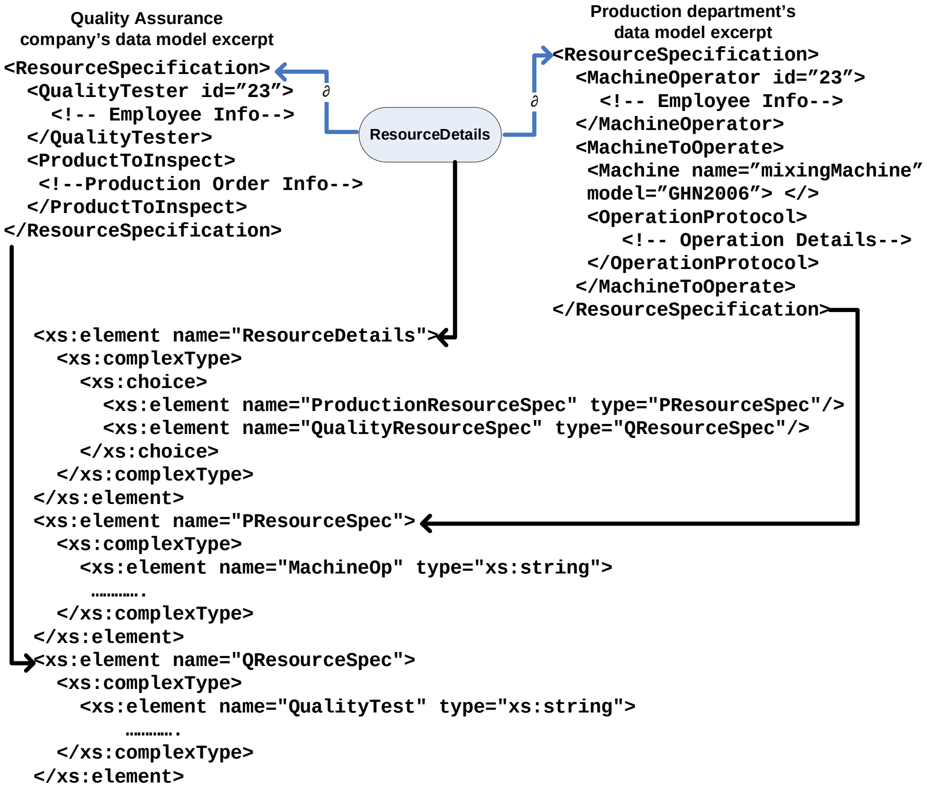
For selective routing and delivery of XML nodes, i.e. encoded and encrypted ($[C_i^x, E_p^x]$ of section A.2), a disseminator D_r follows specific process depending on the recipient (i.e. disseminator, user).

Routing to disseminators: In addition to the distributed hash table and served encoded and encrypted content, each disseminator D_r also maintains the list of concepts C_k requested by other disseminators. D_r performs the following steps for each such request from a disseminator D_k:

1. Determine requested concepts: find all $C_k \in served\_list(D_r)$.
2. Determine XML nodes: match concepts of step one with encoded concepts in C_i^x.
3. Forward the encoded and encrypted XML nodes of step two to the requested disseminator.

Delivery to users: For each subscribed user, u, the disseminator D_r performs the following steps in order.

1. Separate allowed concepts: find all $C_i \in auth\_list(u)$ such that $C_i \in served\_list(D_r)$.
2. Determine allowed nodes: match concepts of $auth\_list(u)$ with stored encoded content (i.e. C_i^x).
3. Extract associated encrypted and encoded XML nodes.
4. Finally, send the encoded and encrypted XML nodes extracted in step three to user u.


Quality Assurance company's data model excerpt
Production department's data model excerpt
ResourceDetails
∂
∂

```
<ResourceSpecification>
  <QualityTester id="23">
    <!-- Employee Info-->
  </QualityTester>
  <ProductToInspect>
   <!--Production Order Info-->
  </ProductToInspect>
</ResourceSpecification>
```

```
<ResourceSpecification>
  <MachineOperator id="23">
    <!-- Employee Info-->
  </MachineOperator>
  <MachineToOperate>
   <Machine name="mixingMachine"
   model="GHN2006"> </>
   <OperationProtocol>
      <!-- Operation Details-->
   </OperationProtocol>
  </MachineToOperate>
</ResourceSpecification>
```

```
<xs:element name="ResourceDetails">
  <xs:complexType>
    <xs:choice>
      <xs:element name="ProductionResourceSpec" type="PResourceSpec"/>
      <xs:element name="QualityResourceSpec" type="QResourceSpec"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="PResourceSpec">
  <xs:complexType>
    <xs:element name="MachineOp" type="xs:string">
     ………….
  </xs:complexType>
</xs:element>
<xs:element name="QResourceSpec">
  <xs:complexType>
    <xs:element name="QualityTest" type="xs:string">
       ………….
  </xs:complexType>
</xs:element>
```

Figure 5: Adapting XML structure. A built up XML schema for the concept ResourceDetails by user 5 of Fig 4

3.5 Adapting XML Structure

Upon receipt of various document portions, users need to adapt or match these nodes with their own XML structure. If a publisher and a subscriber share the same XML structural model (i.e. in the same business unit) then this adaption is easier than that of having different XML structural models (i.e. in different organizations). In the former case, the subscriber may know the complete mapping of ontology concepts to the XML structure defined by the publisher and thus adaption is straightforward. For the latter, the adaption can be addressed using schema matching solutions [19–21]. However, unlike these approaches, a specific algorithm needs to be devised as the adaption should be done over the encoded and encrypted XML nodes. We suggest the following techniques:

1. As users know the shared ontology of the domain and can decode the associated concepts of the received XML content they can build up an arbitrary schema or document using the concept and received content. As such these built up document portions can be associated with users' individual XML data models which essentially would have had complete mapping of the shared ontology concepts. Fig 5 illustrates such an adaption by the user

5 after receiving two document portions for the concept 'Resource Details' of Fig 4.

2. Publishers can send the partial mapping of the concepts to the document portions as part of encoded information which authorized users can utilize as straightforward mapping to their XML data models.

3.6 Unsubscription

As mentioned before we rely on a distributed key agreement scheme that is required to be executed by a group of subscribers in a subscription phase in order to compute the shared key and thus to protect the confidentiality of the XML content and its semantics between the publishers and the subscribers. While a new secret key should be computed by a group of subscribers in the event of a new subscriber for the same concept, the existing secret key can be used in case of a unsubscription of an existing user of the group. This is because for a successful unsubscription the responsible disseminator simply stops sending the associated XML content to that user. For an unsubscription of a concept C_i of a user u, the disseminator D_r performs the following steps:

1. Determines the authorized XML content based on the authorizations of u for the concept C_i.

2. Sends a response back to u stating that unsubscription is successful and stops sending encoded and encrypted XML content of step one to u. Then it checks whether any other authorized user has currently subscribed for the same concept C_i. If no then D_r also forwards the unsubscription request for the concept C_i to it's uplink disseminators in the distributed hash table as no user is subscribed for that concept.

3. Upon receipt of an unsubscription request for a concept C_i from a downlink disseminator, i.e. D_r, D_i sends a response back to D_r stating that unsubscription is successful and stops routing encoded and encrypted XML content associated to C_i to D_r. D_i further checks whether any other authorized user or disseminator has currently subscribed for the same concept C_i. If not, then it performs similar steps as in item 2.

4 Discussion

Selective and scalable dissemination. The ontology based dissemination ensures that a user is delivered only the nodes associated to the concepts it has access to. The notion of a $maximum\ conceptual\ block$ and lemma 1 ensure that a disseminator has only access to that many concepts that its subscribed users collectively are authorized to.

The topology formed by the dissemination scheme relying on the distributed hash tables is acyclic. This ensures the number of hops required for dissemination of a concept to be finite, in particular proportional to the number of successive concept containments. Lemma 1 also assures that any disseminator D_r in a dissemination path $D_i \rightarrow, ...D_r.., \rightarrow D_k$ only forwards concepts that are subscribed collectively at D_r as $maximum\ conceptual\ block$ which is monotonically decreasing along this path. In the worst case, all the subscribers may have access to all the concepts. But in reality the concept authorization is more fine grained. Such a topology certainly facilitates efficient network usage and speedy concept dissemination compare to star, broadcast and point to point topologies. However, efficient bandwidth usage is not guaranteed by this approach. Because, the number of mapped document portions may not decrease in the same fashion like maximum conceptual block decreases monotonically. However, as the concepts map to disjoint document portions this ensures that the same concept and its associated nodes are not published several times to a subscriber.

Policy evolution. Publishers may revise their existing policy, for instance the quality assurance company (i.e. P_2 of Fig 3) could add new rules: (1) dissemination of `<ResourceSpecification>` is allowed to users with $Cred2$ if `<ProductSpecification>` of production department has been disseminated already (i.e. temporal). (2) only one product, described by `<ProductSpecification>` can be tested by them (e.g. seperation of duty). As access control enforcers, disseminators need an automated system for policy evaluation which we developed in [1].

5 Related Work

There has been a remarkable progress in the recent years to address access control issues focusing on XML structure [5–7, 11–14]. The basic model of this work is a typical request response paradigm in a client server architecture. Instead, this paper proposes a publish/subscribe model for semantic based dissemination.

The work of [9,10] focuses on dissemination of XML data exploiting their hierarchical structural properties based on encrypted post order numbers (i.e. EPON). However, the proposed approach in our paper is fundamentally different as policy specification is assumed to be on domain concepts and selective dissemination is performed based on the semantics captured in the concepts as opposed to their structure based dissemination. Moreover, our enterprise XML processing is performed while parsing as opposed to their EPON for which they need to parse the complete XML documents into memory a priori. The routing model proposed in [10] is based on multi-casting of selected document portions from an intermediate router node to the subscribed users. Essentially the router may send the same document portion (i.e. subtree) multiple times to the subscriber as opposed to our approach where we forbid this multiple sending by the dissemination protocol.

Our previous work [16] allows authorized users to exchange document portions using a group key based approach that enables users to be independent of a central authority to which an interested user would have to send access request whenever it needs access. Instead of disseminating the allowed portions of the document to the requesters the authority simply initiates the collaboration by sending an initial version of the document portion among the same kind of authorized users and thus lets them exchange updated documents independently. This also proposes a delegation based authority hierarchy to handle the unavailability of the authorities. The delegation hierarchy is similar to our ontology-based dissemination topology in that each authority being a disseminator in the hierarchy is responsible for authorizing the allowable document portions.

The work in [8, 15] proposes an ontology based access control for XML documents having variant schemas and semantically related documents respectively. However, both consider issues related neither to dissemination of semantically related documents nor to integrity and confidentiality of documents at all.

6 Conclusions and Future Work

This paper describes how authorization policies can be defined on agreed domain concepts which can then be independently mapped to the individual document model. It introduces a publish/subscribe model for scalable and selective dissemination of semantically equivalent XML content to the authorized subscribers. This model describes an ontology-based dissemination topology and subscription management for efficient dissemination. This model also includes techniques of XML structure adaption for the subscribers. While this model relies on a set of disseminators to enforce access control, the confidentiality and integrity of the disseminated content is assured by a secret key computed in distributed fashion and special encoding method respectively.

We are currently implementing the dissemination method described above. We are also investigating how to extend semantic based selective document dissemination to a workflow context in which a document exchange may trigger processing of tasks and the generation of additional documents.

References

[1] M. A. Rahaman, Y. Roudier, P. Miseldine, and A. Schaad, “Ontology-based Secure XML Content Distribution,” in *IFIP SEC 2009, 24th International Information Security Conference, May 18-20, 2009, Pafos, Cyprus*, May 2009.

[2] “Web Services Notification, http://www.oasis-open.org/committees/tchome.php?wgabbrev=wsn.” [Online]. Available: http://www.oasis-open.org/committees/tchome.php?wgabbrev=wsn

[3] “Resource Description Framework (RDF), http://www.w3.org/rdf/.” [Online]. Available: http://www.w3.org/RDF/

[4] "OWL Web Ontology Language Overview, http://www.w3.org/tr/owl-features/." [Online]. Available: http://www.w3.org/TR/owl-features/

[5] W.-C. L. Bo Luo, Dongwon Lee and P. Liu, *A Flexible Framework for Architecting XML Access Control Enforcement Mechanisms*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, December 2004, vol. Volume 3178/2004.

[6] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati, "Fine Grained Access Control for Soap E-services," in *WWW '01: Proceedings of the 10th international conference on World Wide Web*. New York, NY, USA: ACM, 2001, pp. 504–513.

[7] ——, "A Fine-grained Access Control System for XML Documents," *ACM Trans. Inf. Syst. Secur.*, vol. 5, no. 2, pp. 169–202, 2002.

[8] A. Jain, D. Wijesekera, A. Singhal, and B. Thuraisingham, "Semantic-Aware Data Protection in Web Services, Proceedings of IEEE Workshop on Web Services Security held in Berkeley, CA, May 2006," 2006.

[9] A. Kundu and E. Bertino, "A new model for secure dissemination of xml content," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 38, no. 3, pp. 292–301, May 2008.

[10] A. Kundu and B. Elisa, "Secure Dissemination of XML Content Using Structure-based Routing," in *EDOC '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 153–164.

[11] W. Fan, C.-Y. Chan, and M. Garofalakis, "Secure XML Querying With Security Views," in *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM Press, 2004, pp. 587–598.

[12] G. Kuper, F. Massacci, and N. Rassadko, "Generalized XML Security Views," in *SACMAT '05: Proceedings of the tenth ACM symposium on Access control models and technologies*. New York, NY, USA: ACM Press, 2005, pp. 77–84.

[13] G. Miklau and D. Suciu, "Controlling Access to Published Data Using Cryptography," in *VLDB*, 2003, pp. 898–909.

[14] M. Murata, A. Tozawa, M. Kudo, and S. Hada, "XML Access Control Using Static Analysis," in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 2003, pp. 73–84.

[15] V. Parmar, H. Shi, and S.-S. Chen, “XML Access Control for Semantically Related XML Documents,” *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, pp. 10 pp.–, Jan. 2003.

[16] M. A. Rahaman, Y. Roudier, and A. Schaad, “Distributed Access Control For XML Document Centric Collaborations,” in *The 12th IEEE Enterprise Computing Conference (EDOC 2008)*, IEEE, Ed., September 2008. [Online]. Available: http://www.lrz-muenchen.de/~edoc2008/researchPaperProgram.html

[17] M. Kay, “An Anatomy of an XSLT Processor, http://www.ibm.com/developerworks/xml/library/x-xslt2/.” [Online]. Available: http://www.ibm.com/developerworks/xml/library/x-xslt2/

[18] M. A. Rahaman, Y. Roudier, and A. Schaad, “A Comparison Technique for Tree Structured Data,” in *ICIW 2009, 4th International Conference on Web Application and Services, May 24-28, 2009, Venice, Italy*, May 2009.

[19] Y. An, A. Borgida, and J. Mylopoulos, “Constructing Complex Semantic Mappings Between XML Data and Ontologies,” in *The Semantic Web ISWC 2005*. Springer Berlin / Heidelberg, 2005, pp. 563–574.

[20] M. Ferdinand, C. Zirpins, and D. Trastour, “Lifting XML Schema to OWL,” in *Web Engineering*. Springer Berlin / Heidelberg, 2004, pp. 563–574.

[21] E. Rahm and P. A. Bernstein, “A survey of approaches to automatic schema matching,” *VLDB Journal: Very Large Data Bases*, vol. 10, no. 4, pp. 334–350, 2001. [Online]. Available: citeseer.ist.psu.edu/rahm01survey.html

[22] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, “Order preserving encryption for numeric data,” in *SIGMOD ’04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2004, pp. 563–574.

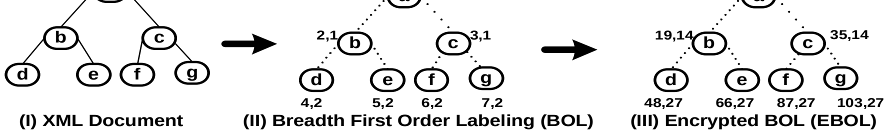


Figure 6: (I) XML document tree. (II) BOL labeling. (III) Encrypted BOL labeling. Solid and dotted lines respectively depict explicit (I) and implicit (II,III) hierarchy representations and storage.

A Processing Enterprise XML

A.1 Encrypted Breadth-First Order Labels for XML Parsing

Publishers parse the XML documents as follows: sibling nodes are stored into a FIFO queue and associated a BOL (an integer pair as defined below) capturing various structural relationships of the parsed XML node (i.e. parent-child, siblings, left/right child) with a minimal memory footprint.

Breadth First Order Labels (BOL): A *BOL* is a pair of integers associated to an XML node as it is parsed in breadth first order. The first integer in the pair is the order associated with a node whose left siblings and ancestors have already been parsed and thus have associated BOLs. The second integer is the depth of the node in the document which is increased by one as new depth level is reached. The BOL starts with (1,0) as illustrated in Fig. 6 (the example given is a binary tree, but BOLs can be defined on any type of tree)

Let a be the parent of two nodes $b, c \in d_i$. We denote its BOL as B_a. Let f_{order} and f_{level} be two functions operating on a BOL respectively returning the BOL order (first attribute of the BOL pair) and BOL depth (second attribute). Let us assume that b is the last child of a parsed and that c is to be parsed next. c will be associated a BOL with $f_{order}(B_c) = f_{order}(B_b) + 1$. $f_{level}(B_a)$ uniquely identifies the depth level of the node a in d. The order of the BOL exhibits the following structural properties:

1. $f_{order}(B_a)$ uniquely identifies node a in document d and the subtree d_a rooted at a.

2. Let $B^a_{Highest}$ be the largest BOL order of a parsed node in document portion d_a; then $B^a_{Highest} > f_{order}(B_z) > f_{order}(B_a)$, where $z \in d_a$.

3. $f_{order}(B_c) > f_{order}(B_b) > f_{order}(B_a)$.

The first property is used to identify and extract a specific document portion from a document. Combined with the depth level of a node, that property ensures that any unexpected move, copy or replace activity in the document is detected. The second property imposes an upper bound on the BOL of any queried node parsed in a document. In effect, it detects if a node is added or deleted and which one it is. The third property permits detecting any unintended swapping among the children in a received document portion (subtree).

A BOL is by definition plain text and thus may reveal important structure specific information (i.e. information leaking), such as number of nodes and thus the size of the document and even hierarchical relationship among the nodes to an adversary. Encryption over such BOL numbers protects this undesired information from leaking.

Encrypted BOL (EBOL): Let B_a be the BOL of an XML node a. Let f_e be an order preserving encryption function [22]. The EBOL of a, denoted as E_a is a pair of integers defined as: $(f_e(f_{order}(B_a)), f_e(f_{level}(B_a)))$. While $f_e(f_{order}(B_a))$ is performed for each node a, $f_e(f_{level}(B_a))$ is performed if a is the first node in a level.

The EBOL preserves exactly the same properties of BOL (see Fig 6). The EBOL order value hides the actual node number and its depth level as opposed to the BOL attributes and thus prevents information leaking.

A.2 Encoding Method

In the following, encoding elements are introduced to describe concepts that are mapped to data units (i.e. subtrees or nodes) as well as the properties of these data units and their encryption.

Node Identifier: Let x be a node in d_i. The node identifier of x denoted as N_x is a tuple formed by three elements $(doc_{id}, E_x, E^x_{Highest})$, where doc_{id} is the document identifier of d_i, E_x is the EBOL of x, $E^x_{Highest}$ is the highest EBOL in the document portion rooted at x.

A node identifier is unique for all documents in the system. The depth included in E_x uniquely determines the node's level. E_x and $E^x_{Highest}$ together determine the parsed document portion. Finally, doc_{id} resolves appropriate XML nodes of the associated document with respect to the same concept.

Node Integrity: The node content consists of attributes, their values and text content inside the tag but not any descendants of the node. The node integrity code is a hash computed out of the concatenation of a node identifier and content, denoted as $I_x = H(N_x, Ct_x)$, where N_x is the node identifier, Ct_x is the content of x, and H is a one way collision resistant hash function.

Content Signature: Let C_i and x be a concept and an XML node respectively. The content signature, denoted as C^x_i, is a pair (N_x, C_i), where N_x is the node identifier of x and C_i is a concept mapped to x. The $content\ signature$ incorporates semantic information such as conceptual and structural information attached to an XML nodes.

Content Encoding: An encoding information CE_x of a node x is $CE_x = (C^x_i, I_x)$, where C^x_i is the $content\ signature$ and I_x is the node integrity respectively. Each XML node x is encoded as a pair $[CE_x, C^z_i]$, where CE_x is the encoding information of node x and C^z_i is the $content\ signature$ of the parent node z of x. For the root node of a document the encoded node is $[CE_x]$.

Document Encryption: Each encoded node is encrypted using the common key computed [16] by a group of subscribers and the publishers for an authorized concept. After encryption, an XML node x is represented as $[C^x_i, E^x_p]$, where C^x_i is the $content\ signature$ of x and E^x_p is the encrypted value of the content encoding pair $[CE_x, C^z_i]$ of the node x.

An algorithm for enterprise XML processing using above encoding elements is provided in Fig 7. Furthermore, we also provided an algorithm (Fig 8) that illustrates the selective routing and delivery process of XML content by the disseminators.

```
1. Input: $C$, a collection of concepts $\{C_i\}$; a set of documents identified by $\{doc_{id}\}$;
   Output: Encrypted and encoded document.
2. Let $B \in \mathbb{N}$ be an integer for BOL, $l \in \mathbb{N}$ be the depth level of a node, $Q$ be a FIFO queue.
3. FOR all documents $\{doc_{id}\}$ do
   (a) **Initialize:** set $B = 0; l = 1; Q[0] = l; Q[1] =$ root node of $doc_{id}$.
   (b) WHILE $Q$ is not empty
       Let $x$ be the current node.
         i. IF $x$ is a level delimiter.
            set $l = l + 1$; Add $l$ into $Q$.
        ii. ELSE
            A. **BOL Generation:**
               POP $x$ from $Q$; set B=B+1; Associate $(B, l)$ to $x$, $B_x = (B, l)$.
            B. Add all the children nodes of $x$ into $Q$.
            C. **EBOL computation:**
               Compute $(f_e(f_{order}(B_x)), f_e(f_{level}(B_x)))$.
            D. **Document Encoding:**
               Determine node identifier of $x$: $N_x = (doc_{id}, E_x, E^x_{Highest})$.
               Determine mapping: $\vartheta(C_i, x)$.
               Determine content signature of $x$: $C^x_i = (N_x, C_i)$.
               Compute node integrity of $x$: $I_x = (N_x, Ct_x)$.
               Encode node of $x$ as $CE_x = (C^x_i, I_x)$.
            E. **Document Encryption:**
               Encrypt the document node as $E^x_p(CE_x, C^z_i)$; where $C^z_i$ is the content
               signature of the parent of $x$.
               Generate encoded content as $(C^x_i, E^x_p)$.
```

Figure 7: An illustrative algorithm for enterprise XML processing.

```
1. Input: A collection of encoded and encrypted XML content, i.e. $(C_i^x, E_p^x)$.
   Output: Selective routing and delivery of enterprise XML.
2. Disseminators Initialization:
   FOR all disseminators $D_i$ do
   (a) Fill in the distributed hash tables for uplink and downlink disseminators.
   (b) Retrieve the authorization policies of all the document publishers.
   (c) Initialize served concepts list of $D_i$ as null, i.e. $Served\_list(D_i) = null$.
       FOR all downlink disseminators $D_j$ of individual distributed hash table do
         i. Send encoded and encrypted XML nodes associated to the $maximum$
            $conceptual\ block$ of $D_j$.
3. User Subscription:
   FOR each subscription request from a user $u$ do
   (a) Authorization Determination:
        i. Determine the set of authorized concepts based on the authorization policies.
       ii. Send the content signatures of the authorized concepts, i.e. $Auth\_list(u)$, to $u$.
       Let $C \in Auth\_list(u)$.

       IF $\forall C \in Served\_list(D_i)$ then
       Registers the user $u$ for $C$.

       ELSE
        i. Determine $Auth\_list(u) - Served\_list(D_i)$, i.e. $\{C_k | C_k \in Auth\_list(u) \setminus$
           $Served\_list(D_i)\}$.
       ii. FOR each $C_k$ do
           IF $C_k \preceq maximum\ conceptual\ block$ served by itself, i.e. $D_i$ then
           Send requests to uplink disseminators of its distributed hash table.
           ELSE
           Send requests to downlink disseminators (if there is) of its distributed hash table.

   (b) FOR each request from a disseminator $D_i$ for a concept $C_k$, $D_j$ do
        i. Checks whether $C_k$ is served by itself, i.e. $D_j$.
       ii. IF $C_k \in Served\_list(D_j)$ then
           Send encoded and encrypted XML nodes associated to the concept $C_k$, i.e.
           $(C_i^x, E_p^x)$.
   (c) FOR a set of responses i.e. $(C_i^x, E_p^x)$ from other disseminators, $D_j$ for a requested
       concept $C_k$ do
        i. Applies selection policy to choose the disseminator.
       ii. Perform document verification.
4. Document Verification:
   $D_i$ performs ontology-based verification and structure-based verification in order.
   IF Verification is successful then
   Adds all the XML content i.e. $(C_i^x, E_p^x)$ into its $Served\_list(D_i)$.
   Registers the user $u$ for $C_k$.
5. Selective delivery to subscribers:
   FOR each subscribed user, $u$ do
   (a) Determines $u$'s concept authorizations $(auth\_list(u))$.
   (b) Separates the allowed concepts from its served concepts by finding $(\forall C_i \in$
       $auth\_list(u)) \in served\_list(D_i)$.
   (c) Determines the allowed XML nodes by simply matching the concept of $auth\_list(u)$
       with corresponding concept in the served $C_i^x$ and thus extracting the associated encrypted
       and encoded XML nodes, $(C_i^x, E_p^x)$.
   (d) Sends the encoded and encrypted XML nodes i.e. $(C_i^x, E_p^x)$ to the subscribed user $u$.
```

Figure 8: An illustrative algorithm for enterprise XML routing of disseminators.