Secured Web Access

Mukesh Mohania* Vijay Kumar† Yahiko Kambayashi‡ Bharat Bhargava§

Abstract

In this paper we discuss various secured web access schemes using dynamic and static approaches. In a static approach the access environment, that is, set of authorized users, the mode of access, their access rights, etc., are predefined. This approach is suitable only for a static set up where the user requirements do not change frequently. In the dynamic approach, on the other hand, the authorized user set is defined when web pages access requests appear. An interested user to the web is authenticated by necessary information provided by the user. Once the information is verified, the user is either given conditional access, timed access, or full access only to the information relevant to the user.

1 Introduction

A web is a universal repository of information, which sits over the internet. It provides unlimited and instantaneous access to information and communication. In addition to these, web links nearly all information residing on the internet, thus creating an interconnected information space. Any user can access desired information on the web and the information can come from anywhere in the world.

Today the web is no longer a universal repository and a warehouse of information for organizations but it has become their showcase too. Such new trends in the use of web has created a conflicting scenario, which undermines its universality. The following examples illustrates some of the conflicting scenarios with today's web infrastructure.

Example 1 *Students of a university has secured access to their grades over the university web. However, when the student applies to other universities, then these universities have no permission to access student's grade on the web for verification. This is a classic example of inefficiency in legally sharing data over the web.*

Example 2 *A credit card company wants to verify credit history and personal information of one of its new applicants. If the data is not located within the scope of a predefined site, then the company cannot verify applicant's data. This is a commonly used verification scheme but the present systems do not provide this facility efficiently. Thus there is clearly a need for a scheme better than the existing authorization and verification schemes.*

Example 3 *Federal agencies compile and post scientific research data on to their web sites. The agencies may have set user permissions for a certain set of users to access this information. But there may be other legitimate users like students, researchers, etc., who would not be able to access these data.*

Example 4 *Universities post faculty information (evaluation, salary, etc.) on the web for authorized people. However, after authentication a student may be able to view this information in the absence of a reliable access management scheme.*

These examples indicate the nature of the problems that need to be investigated. In this paper we discuss a number of issues related to static and dynamic security. We discuss the authorization and access control mechanism for web documents that allows the definition and enforcement of access restrictions directly on the structure and content of the document so that the right users can access the right data.

2 Web Architecture

Several web database system architectures have been proposed LORE [17], WHOWEDA [18], FLORID [4], and W3QS [15]. These systems retrieve and manipulate semistructured data by supporting web query languages. For instance, in W3QS [15] a user can specify content and structure queries on the WWW and can maintain the results of queries as database *views* of the WWW. These web database systems allow users

*Dept. of Computer Science, Western Michigan University, Kalamazoo MI 49008, U.S.A. Email: mohania@cs.wmich.edu

†Computer Science Telecommunications, University of Missouri-Kansas City, U.S.A. Email: kumar@cstp.umkc.edu

‡Dept. of Social Informatics, Kyoto University, Kyoto, Japan Email: yahiko@kuis.kyoto-u.ac.jp

§Dept of Computer Science, Purdue University, Email: bb@cs.purdue.edu. This research is supported by CERIAS.

0-7695-1022-1/01 $10.00 © 2001 IEEE

to access and manipulate any part of the data, however, they do not address the issues of how to secure the web data on *views* and how to define access privileges for different users. In this paper, we discuss a security model that can sit on top of these systems so that web data can be secured from unauthorized users. There are several mechanisms to specify security requirements at the file level or with reference to the HTML constructs but these are very limited [24, 23]. Our work is motivated by these limitations. Figure 1 illustrates our

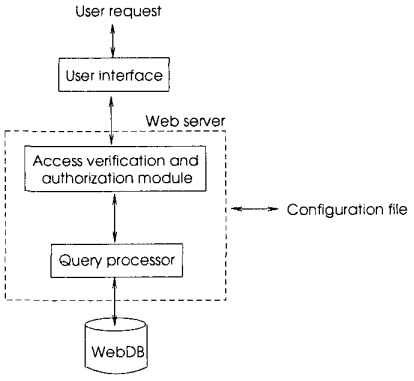

Figure 1: Web Database Architecture

platform, which has four main components (a) "user interface", (b) "access verification and authorization module", (c) "query processing", and (d) "WebDB". The user presents his/her access request through the "user interface". The "access verification" component authenticates the user using information provided by the user, such as username, password, etc. It uses the dictionary ("configuration file"), which holds the access information related to the user and defines the web elements (either on each page or part of it called object) for an authorized user. Once the user is authenticated the request is forwarded to the query processor. Depending on the materialization policy, either virtual or materialized, (discussed in detail in Section 3), the query processor executes the query and displays the results. One of the important property of our web database system (WebDB) is that the web pages in "WebDB" are stored in a multilevel hierarchy. Figure 2 illustrates the hierarchical organization of web pages in "WebDB". The web pages other than those on leaf nodes are derived from their children nodes. Such architecture is important for the environment where the web pages are queried on aggregate data for fast retrieval of information. In existing web database systems [17, 18, 4, 15] information is retrieved from querying underlying sources.

With the following example (Figure 2) we illustrate the process relevant web *view* access. In the example,

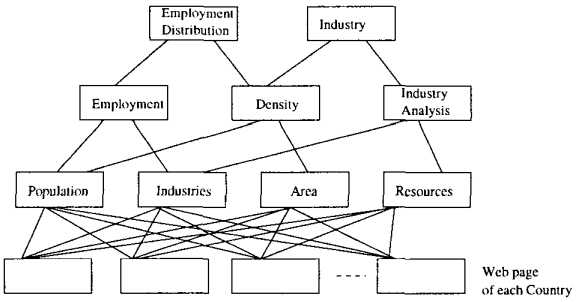

Figure 2: Hierarchy of Web Pages

at the leaf level there are web pages of each country. At the next higher level pages we aggregate the information about population, industries, area and resources, which is stored as separate pages as shown in Figure 2. Then at the next higher level, we aggregate information about employment from population and industry data, density from population and area, industry analysis from industry and resources. At the top level, we can have data about employment distribution derived from employment and density. Such hierarchy is necessary in answering queries like "How many employees are there in certain area?", "What kind of industries are there in a particular place?", etc. In this architecture, we can allow a set of pre-defined users to access certain web pages or part of these web pages. However this restricts information sharing among those users who are not pre-defined. Thus, it is important to define the security measures at each web page level or at each element level and to enable undefined but valid users to access a web database.

3 Related Work

"Username" and "password" are commonly used for user authentication. This simple approach has problems of their own. A password is only good if it's chosen carefully. The risk is that the password is vulnerable to interception as it is transmitted from browser to server. Public key cryptographic systems provide a more sophisticated form of authentication that uses non-forgettable electronic signature [8]. There are several standard access restriction techniques that are available for protecting the confidentiality of the documents at the required site. The commonly used among them are: (1) Restriction by IP address, subnet, or domain. Individual documents or whole directories are protected in such a way that only browsers connecting from certain IP (Internet) addresses, IP subnets, or domains can access them. (2) Restriction by user name and password

Documents or directories are protected so that the remote user has to provide a name and password in order to get access. (3) Encryption using public key cryptography where both the request for the document and the document itself are encrypted in such a way that the text cannot be read by anyone but the intended recipient. Public key cryptography can also be used for reliable user verification [8].

Most web servers that support authorizations and access control mechanism use a "configuration file" that contains the list of users, hosts (IP addresses), or user/host pairs, which allow or forbid connection to the server [23]. The configuration file indicates whether a single file (i.e. a web page) can be accessed by a given user. Such mechanism, however, fails to specify authorizations on different portions of web pages. There has been some work [24] that specify authorization at a fine granularity by considering a Dexter-like model for referencing portions of a file. Sophisticated access control mechanisms have been proposed for supporting security requirements and multiple policies in distributed and heterogeneous databases [13, 28]. However, these mechanisms cannot be applied in web databases because of flexible and volatile data organization and the semantic context is different.

In [9], an access control model has been proposed for restricting access to web documents. They define *XML* markup for a set of security elements that describe the protection requirements of *XML* documents. The security markup is used to provide instance level and schema level authorizations with the granularity of *XML* elements. They maintain a dictionary that indicates which *XML* elements can be accessed by a user. For a valid user (after verifying the user name and password), the *XML* parser displays information associated with all those *XML* elements that can be accessed by this user. Their method seems to be in the right direction towards the construction of an access control mechanism when number of users are less and there are not many tags in *XML* documents. Moreover, when *XML* documents are changed, the entries in the dictionary needs to be changed. One way to solve this problem is to define views for each user and maintain a dictionary that indicates which user can access what views. As we can see the number of entries in this dictionary will be equal to the number of predefined users, thus it will be easy to maintain the dictionary.

4 Security Schemes

The web database system architecture discussed earlier could be viewed as generation of web pages at a higher level in the hierarchy from integrating data from the lower level pages. Security of these pages is a prime concern because there is a lot of information that could be associated with these pages, probably an entire database of a particular company or an institution. We envision two modes of security to cater for all types of secured access.

- **Static:** To manage security in the least fluid environment (i.e., where the contents of web and user requirements change rarely). Under this approach a set of users and their access rights are predefined. Security is provided for pages or part of these pages at different levels in the hierarchy as well as for the pages at the same level in the hierarchy.
- **Dynamic:** To manage security in frequently changing environment. Under this scheme user set is not predefined, rather users are asked to enter appropriate information that which is used to authenticate him/her. After obtaining the information, the system performs various checks on this information using a mobile agent that decides on the access restrictions to be imposed on the user.

4.1 Static Security Scheme

The "static" level security can be enforced by defining web *views* (a web *view* is an instance of web page) over web database. Defining *views* over web are more complicated than relational databases since web schemas [18] do not have rigid structures and they are not transparent to users. Thus, web *views* can be used to introduce some structure and they can group some arbitrary portion of a database into a logical unit. Another important advantage of *views* of the web is that it can be seen as personalized web pages. Now a days, most of the commercial web sites (e.g. yahoo.com, tdwaterhouse.com) offer users to define their own personalized web pages that can have information users may be interested. Each personalized web page can be seen as a web *view*. These *views* are dynamically generated from web data stored in *HTML* or *XML* documents based on the parameters in a query [26]. These personalized web *views* can be defined either virtually (i.e. views can be predefined using view specification language [1] in the web server [2] and they are computed dynamically usually by a CGI script), or they can be materialized (i.e. *views* are precomputed and stored) [6, 16, 19], or these *views* can be generated dynamically on-demand (this method is called "query-driven" method).

When the user logs on, the system checks user's validity. This can be done by creating a configuration file by "htpasswd" program. This file is stored in web

administrator's directory who enters user's names and passwords. The passwords in this file are encrypted using a simple algorithm and it is important that "htpasswd" is readable by the web server. It is also possible to create separate passwords files for various web directories but all the password files should be stored in "/passwd" directory. Any web access to a file in a private directory will cause the browser to prompt for a username and password. The owner of that file, is the only person that can change passwords and add new users to the file.

There can be two approaches for implementing the access control, namely (1) to materialize web *views*, and (2) to define virtual web *views*. Recall that a web *view* is an instance of web page that is automatically constructed from *XML* documents using a program.

- **Materialized Web Views:** Materialized web *views* are predefined web pages that are created for each set of users and these *views* reside on the web server (i.e. these web pages are not generated each time a user logs on or each time a page is requested). Web *view* materialization [16, 25] is different from the traditional web caching techniques [7, 5, 12]. The problem of web *view* materialization is similar to that of deciding which *views* to materialize in a data warehouse, is known as the "view selection" problem. However, there are significant differences between the two. First of all although both problems aim in decreasing query response times, "warehouse views" are materialized in order to speed up execution of OLAP queries whereas web *views* are materialized to avoid generating the same information again and again. Secondly, since web *views* are defined as a result of user requests, unlike "warehouse views", the resulting search space for decision problem is significantly smaller. Finally, web *view* materialization problem may not have any constraints, where as most *view* selection algorithms impose some resource constraints. This access control approach is shown in Figure 3.

  The main drawback of this approach is wastage of storage because of the redundancy involved. Secondly, the problem of update occurs as same information may exist in many pages, which requires that each page has to be updated each time a change occurs and this is a major maintenance overhead.

- **Virtual Web Views:** Virtual web *views* are different from the materialized views because they

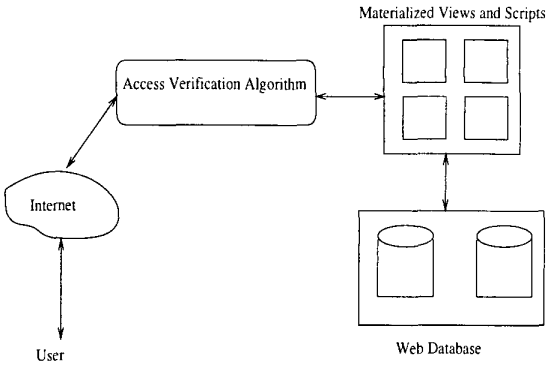

Figure 3: Materialized Web View Scheme

are computed on the fly. This is the most common way in existing database-backed web servers for automatically generating a hypertext for data presentation purpose and is based on the virtual *views* that refresh the data dynamically [27]. Recently, there has been some work on defining the *views* over a semistructured data [2, 1]. Virtual web *views* are computed dynamically on-demand, usually by a CGI script. The cost to compute the web *view* increases the query response time. Virtual web view approach is shown in Figure 4.

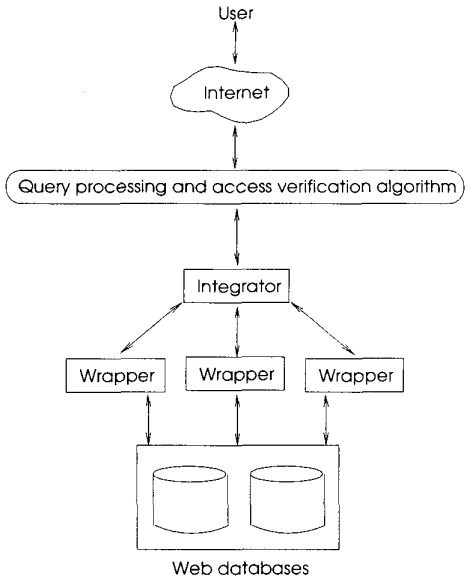

Figure 4: Virtual Web Scheme

4.1.1 Limitations of Static Security Scheme

These methods do provide ample security to HTML or *XML* documents (web pages), however, their main dis-

advantage is that the system has to maintain either a predefined set of user names or a predefined set of domain names or both and access restrictions for each. And by doing this, these systems restrict information from being shared over the web. That is, only a set of users who are predefined can access the information. A user who does not have any entry in the configuration file cannot access the information. The following explain illustrates the inefficiency of these approaches. The university's web site maintains a list of users, typically students and staff members. The students have access to their grade information and other academic data. In the event a student grade information needs to be verified by another university considering a transfer application by the student, the system will not share this information because of access restriction. However, the university cannot open up the system to public access due to matters related to information privacy.

4.2 Dynamic Security Scheme

The problem is to design a security system that interacts with the user cumulatively and thus provides authentication and personalized views based on analysis and verification of the interactive results. The server makes use of mobile agent technology to cumulatively interact with user and to move to related servers and verify data provided by the user. Thus a given server need not predefined a set of users and permissions. It would be the mobile agent's responsibility to provide the user with an initial set of questions and based on the answers provided by the user, the agent generates questions dynamically, thus interacting cumulatively with the user. Based on all the answers that the user provides, the mobile agent performs verification operations by traveling to different sites on the web and verify data provided by the user. Based on the findings of the mobile agent the system decides on the access restrictions to be imposed on the user.

There are three main problems that are eminent in dynamic level security approach. The first problem is to design a mobile agent. Mobile agents are agents that can physically travel across a network, and perform tasks on machines that provide agent hosting capability [21]. This allows processes to migrate from computer to computer, for processes to split into multiple instances that execute on different machines, and to return to their point of origin. Unlike remote procedure calls, where a process invokes procedures of a remote host, process migration allows executable code to travel and interact with databases, file systems, information services and other agents. Mobile agents should have the following properties:

- **Goal oriented** – they do not simply act in response to the environment;
- **Communicative** – they are able to communicate with other agents;
- **Mobile** – they are able to transport themselves from one host to another.

In addition to this will also have the capability to interact with the user cumulatively. The second problem is to define a method to identify the right users. In order to achieve this, the mobile agent should have the capability to dynamically generate questions based on data provided by the user and should be capable of deciding whether or not the user is a legitimate user.

The third and final problem is to determine a method to decide on access privileges given to the users. Since user authorization is dynamic, the level of access rights varies for every user depending on the investigation of the mobile agent. If the mobile agent, decides that a user be given certain privileges, the problem lies on how to do this as they are determined dynamically.

4.2.1 Designing a Mobile Agent

The ability of mobile agents to fragment themselves into many pieces that travel to different points across the network sounds promising. It might enable new forms of interaction, such as negotiating agents that travel to vendors seeking the best deal, or meeting places where agents can "get together" and communicate [21]. The technology of mobile agents, where software pieces of active control and storage (called mobile agents) travel the network and perform tasks distributively, is of growing interest as an Internet technology. Such mobile processing can be employed in large scale census applications in statistics gathering, in surveys and tallying, in reading and collecting local control information, etc. This distributed computing paradigm where local pieces of data are getting accumulated in a mobile unit presents new information security challenges. Conversely, we plan to make use of this mobile agent technology to provide security for an information database. The basic problem involves the design of a mobile agent that is capable of traversing on untrusted (curious) network while gathering and securing data from the nodes that it visits. This kind of performance is imperative due to the functional nature of the desirable mobile agent in the proposed plan of approach.

There are two basic functions that need to be performed by the mobile agent. The first is to perform a user interactive session, and verify if the data provided by the user is valid. The next function is to authenticate the web user to access the local client and to provide a

set off access restrictions for the user, based on the interactive data or information.

The process of setting up an interactive session with the user and verifying if the user is a valid user involves three phases.

- Provide the user with a predefined set of questions and take user input for processing.
- Process the data provided by the user for generating queries dynamically in a predefined format, and take user input for processing to the next level.
- Perform a verification algorithm including both static and dynamic verification to assess the validity of the user. Static verification refers to verification of IP addresses and such. Dynamic verification involves the agent moving to the remote server and checking for validity of the information.

The next step is to authenticate the user on the local server. This involves, providing a session id for the user, identifying what data can be provided to the user and what cannot, changing access restrictions on the data for the active session id, deleting the session id after the user completes using the system, and resetting the access rights. We plan to make use of a "Framework" model for the mobile agent given in [10].

In order to facilitate the tasks of mobile agents, they must have (a) communication, (b) transportation, and (c) data storage and extraction capabilities. These capabilities provide the basic level of service under which they can be expected to operate above the code that provides the decision processes, which lead to the invocation of those services. In addition to the provision of these services, the "Framework" must also allow agents to execute their internal code, and maintain their state at all times. The "framework" should me able to increase the efficiency of individual agents, by reducing their complexity. A by-product of moving these services from the agent to the "framework" is a reduction in the overhead required by the "framework" to process an agent. This is analogous to a computer's operating system, such as Microsoft Windows(TM). In order to provide a rich graphical user interface (GUI), and an environment which supports a variety of services, Microsoft supply an Advanced Programming Interface (API). This API accepts function calls, which provide services to an application, thus reducing the complexity of that application.

There are also services, which are provided by the operating system and are called implicitly whenever they are required. These include notifying the application when it is the focus of the user's attention, and managing the memory required to store its resources, such as graphics and data. Without the API and the underlying services, a Windows application would be unnecessarily complex. It should be noted that this approach also leads to the standardization of applications, that is, applications can only use a certain number of function calls, which are well documented and supported. An "Agent Framework" provides the execution space in which an agent may choose to perform a task or series of tasks with minimum restriction, while maintaining efficiency and providing, security both for the legitimate Agents and against malicious intruders. The "framework" is the key to the success of the agents in providing the service for which they were designed. We plan to use the "Agent Framework" model to design an efficient mobile agent to perform its tasks.

4.2.2 Identifying the Right Users

The process of identifying the right users involves analysis of the user input. Consider the example of a student applying for a transfer from university "A" to another university "B"'. The enrollment office of this university attempts to verify the validity of the information provided by the student by accessing the other university's web site. The task of the mobile agent that resides on "A", is to verify the validity of the user. It is a three-tier process. The mobile agent first asks the user a predefined set of questions.

- What is the user's location?
- What is the name and URL of the University requesting information?
- What is the position of the user in this university?
- What is the name of the student?

The mobile agent takes the input from the user and attempts to first verify this information. The agent travels to the remote location specified by the user and interacts with the server to verify if the user had requested information, if the user is currently awaiting authorization from the server, and if the user has the rights to verify such information. This might require the remote server to have agent hosting capability, which is necessary for the mobile agent to perform the said tasks on the remote server. Once this information is verified, the mobile agent determines that the user is a legitimate user requesting information from a trusted domain (in this case a .edu domain). The next functionality of the mobile agent is to verify if the said student had applied for a transfer to the university and thus determine if

it is necessary to give the requested information to the user. This is done by asking questions to the user like,

- What is the student's identification number?
- What is the student's date of birth?
- What is the enrollment date of the student?
- Which school did the student previously attend?

These kinds of questions will help determine if the user has an application from the student. Thus the mobile agent is now certain that the user is a valid user, and that the server can share information to the user.

4.2.3 Providing Access Restrictions

The next task of the mobile agent is to determine the access restrictions it can provide to this valid user. Upon determination of the user's validity, the agent creates a session id for this user, and assigns to this user access permissions to the data that resides on the server. The agent should be capable of this because. With the interactive question-answer session with the user, the agent now knows that the user is required to access information pertaining to the required student. Thus, the agent allows the user to retrieve information. But, the agent should also be careful not to give secure information to others. After the user has retrieved the required information the agent should end the session by expiring the session id and resetting the access permissions. Thus, the mobile agent should be capable of all these operations.

The above mentioned functions of the mobile agent explain its use in one case: the case of a student's information being verified by another university. However, on a general approach, we would like to put forward a few concepts.

The agent assumes that the data base has two different sets of data, one that can be shared and the other that has higher security parameters. The latter requires conventional security access methods like user based or domain based restrictions or a combination of both. The former is the data that can be shared, but yet has security concerns. This data can further be classified as:

- The data that is used by the agent to assist in determining the validity of the user.
- The data that is provided to the user only after validating the user.

This assumption does not impose changes to any existing data models. The mobile agent by itself should be capable of defining a set of rules on the data to satisfy the above mentioned criteria.

One of the problems is how quickly a mobile agent can authenticate each user without having any prior knowledge about each user. One of the approach is to define a set of rules that form some initial basis based on the classifications of users. Another approach can be based on referring other parties that have already established secure communication with the mobile agent. Another problem is how to maintain the initial authentication between the mobile agent and the user so that when this user logs on second time, the mobile agent does not have to establish the authentication again. One idea is to maintain a log file that contains all the information about user and the conversation between user and mobile agent. When the user logs on, the mobile agent should check the log file first. If the entry of this user is found, the agent should verify the user's identity by asking a set of questions related to the information stored in log file.

5 Conclusions

Authorized access of web organizations today need additional control over the use of their information on the web. Some of the important ones are (a) authorized users should be able to "see" the web views only relevant to them, (b) organizations should be able to define and efficiently authenticate (validate) new and old users respectively, (c) organizations should be able to use web views to structure and group arbitrary portion of the database into a logical unit, (d) organizations should be able to personalize and create web views, (e) organizations should be able to provide security at the "page level" and also at the "tag level", and so on.

When data is classified and the cost is associated with data, it becomes very important to provide security so that the right users can access the right data. In this paper we have discussed static and dynamic secured web access schemes using mobile agent. We have also discussed the design of mobile agent for dynamic authentication of incoming users.

References

[1] Abiteboul S., 'On Views and *XML*', In *Proc. of Principles of Database Systems*, 1999.

[2] Abiteboul S., Goldman R., McHugh J., Vassalos V., and Zhuge Y., 'Views for Semistructured Data', http://www-db.cs.stanford.edu/

[3] AlphaWorks, 'XML Security Suite', 1999, http://www.alphaworks.ibm.com/aw.nsf/html/XML_Security_Suite.

[4] Bertram L., Rainer H., Georg L., Wolfgang M, and Christian S., 'Managing Semistructured Data with FLORID: A Deductive Object-Oriented Perspective', *Information Systems*, 23(8), 1998.

[5] Cao Pei, Zhang J., and Beach K., 'Active Cache: Caching Dynamic Contents on the Web', In *Proc. of IFIP International Conference on Distributed Systems Plateforms and Open Distributed Processing*, pages 373-388, 1998.

[6] Catarci T., Iocchi L., Nardi D., and Santucci G., 'Conceptual Views over the Web', In *Proc. of the 4^{th} KRDB Workshop*, Greece, 1997.

[7] Challenger J., Iyenger A., and Dantzig P., 'A Scalable System for Consistency Caching Dynamic Web Data', In *Proc. of IEEE INFOCOM*, USA, 1999.

[8] Cormack A., 'Web Security' http://www.jisc.ac.uk/acn/authent/cormack.html

[9] Damiani E., Vimercati S., Paraboschi S., and Samarati P., 'Securing XML Documents', *To appear in EDBT*, 2000.

[10] Guy W. and Lecky T., 'Mobile Agents and Framework Models', 1997, http://www.geocities.com/CapeCarnival/Lab/1888/fw.htm.

[11] How do browsers (IE4.01) trap XML streams coming from server? http://www.lists.ic.ac.uk/hypermail/XML-dev-Jul-1998/0493.html

[12] Iyenger A., and Challenger J., 'Improving Web Server Performance by Caching Dynamic Data', In *Proc. of International Symposium on Internet Technologies and Systems*, USA, 1997.

[13] Jajodia S., Samarati P., and V.S¿ Subramanian, ' A Logical Language for Expressing Authorizations', In *Proc. of IEEE Symposium on Security and Privacy*, pages 31-42, 1997.

[14] King C., 'Web Access Authentication Using RADIUS', CISSP, *Web Techniques Magazine*. August 1996.

[15] Konopnicki D., Shmueli O., 'W3QS: A query System for the World Wide Web, In *Proc. of 21^{st} International Conference Very Large Databases*, September 1995, pages 54-65.

[16] Labrinidis A.. and Roussopoulos N., 'WebView Materialization', To appear in *Proc. of International Conference on Management of Data (SIGMOD)*, 2000.

[17] McHugh J., Abiteboul S., Goldman R., Dallan Q., Widom J., 'Lore: A Database Management System for Semistructured Data', http://www-db.stanford.edu/lore

[18] Ng Wee-Keong, Lim Ee-Peng, Bhowmick S., Madria S., 'Web Warehousing: Design and Issues', Proc. of *International Workshop on Data Warehousing and Data Mining*, LNCS 1552, Singapore, 1998.

[19] Rainsford C.P. and Mohania M.K., 'Maintenance and Analysis of Web Space Abstractions', In *Proc of International Symposium on Database Applications in Non-Traditional Environments*, Japan, (Proc. by IEEE Computer Society Press), 1999.

[20] Reagle J. and Cranor L.F., 'The Plateform for Privacy Preferences', *Communications of the ACM*, 42(2):48-55, February 1999.

[21] Reilly D., 'Mobile Agnets Process Migration and its Implications', http://www.davidreilly.com/topics/software_agents/mobile_agents/.

[22] Restricting Web Access by Username and Password http://www.columbia.edu/cu/help/htpasswd.html.

[23] Rutgers Security Team, 'WWW Security: A Survey', 1999, http://www-ns.rutgers.edu/www-security

[24] Samarati P., Bertino E., and Jajodia S., 'An Authorization Model for a Distributed Hypertext System', *IEEE Transactions on Knowledge and Data Engineering*, 8(4):555-562, 1996.

[25] Samtani Sunil, Kumar Vijay, and Mohania Mukesh, 'Self Maintenance of Multiple Views in Data Warehousing', 8^{th} ACM International Conference on Information and Knowledge Management (CIKM), U.S.A., 1999.

[26] Selena S., 'Introduction to Databases for the Web', http: www.extropia.com/tutorials/sql/toc.html

[27] Sindoni G., 'Incremental Maintenance of Hypertext Views', In *Proc. of WebDB*, 1998.

[28] Thomas Y.C. and Simon L.M., 'Authorizations in Distributed Systems: A New Approach', *Journal of Computer Security*, 2(2,3):107-136, 1993.

[29] Wong H.Chi and Sycara Katia, 'Adding Security and Trust to Multi-Agent Systems', http://www.cs.cmu.edu