Database Security—Concepts, Approaches, and Challenges

Elisa Bertino, *Fellow*, *IEEE*, and Ravi Sandhu, *Fellow*, *IEEE*

Abstract—As organizations increase their reliance on, possibly distributed, information systems for daily business, they become more vulnerable to security breaches even as they gain productivity and efficiency advantages. Though a number of techniques, such as encryption and electronic signatures, are currently available to protect data when transmitted across sites, a truly comprehensive approach for data protection must also include mechanisms for enforcing access control policies based on data contents, subject qualifications and characteristics, and other relevant contextual information, such as time. It is well understood today that the semantics of data must be taken into account in order to specify effective access control policies. Also, techniques for data integrity and availability specifically tailored to database systems must be adopted. In this respect, over the years the database security community has developed a number of different techniques and approaches to assure data confidentiality, integrity, and availability. However, despite such advances, the database security area faces several new challenges. Factors such as the evolution of security concerns, the "disintermediation" of access to data, new computing paradigms and applications, such as grid-based computing and on-demand business, have introduced both new security requirements and new contexts in which to apply and possibly extend current approaches. In this paper, we first survey the most relevant concepts underlying the notion of database security and summarize the most well-known techniques. We focus on access control systems, on which a large body of research has been devoted, and describe the key access control models, namely, the discretionary and mandatory access control models, and the role-based access control (RBAC) model. We also discuss security for advanced data management systems, and cover topics such as access control for XML. We then discuss current challenges for database security and some preliminary approaches that address some of these challenges.

Index Terms—Data confindentiality, data privacy, relational and object databases, XML.

1 Introduction

As organizations increase their adoption of database systems as the key data management technology for day-to-day operations and decision making, the security of data managed by these systems becomes crucial. Damage and misuse of data affect not only a single user or application, but may have disastrous consequences on the entire organization. The recent rapid proliferation of Web-based applications and information systems have further increased the risk exposure of databases and, thus, data protection is today more crucial than ever. It is also important to appreciate that data needs to be protected not only from external threats, but also from insider threats.

Security breaches are typically categorized as *unauthorized data observation*, *incorrect data modification*, and *data unavailability*. Unauthorized data observation results in the disclosure of information to users not entitled to gain access to such information. All organizations, ranging from commercial organizations to social organizations, in a variety of domains such as healthcare and homeland protection, may suffer heavy losses from both financial and human points of view as a consequence of unauthorized data observation. Incorrect modifications of data, either intentional or unintentional, result in an incorrect database state. Any use of incorrect data may result in heavy losses for the organization. When data is unavailable, information crucial for the proper functioning of the organization is not readily available when needed.

Thus, a complete solution to data security must meet the following three requirements: 1) *secrecy* or *confidentiality* refers to the protection of data against unauthorized disclosure, 2) *integrity* refers to the prevention of unauthorized and improper data modification, and 3) *availability* refers to the prevention and recovery from hardware and software errors and from malicious data access denials making the database system unavailable. These three requirements arise in practically all application environments. Consider a database that stores payroll information. It is important that salaries of individual employees not be released to unauthorized users, that salaries be modified only by the users that are properly authorized, and that paychecks be printed on time at the end of the pay period. Similarly, consider the Web site of an airline company. Here, it is important that customer reservations only be available to the customers they refer to, that reservations of a customer not be arbitrarily modified, and that information on flights and reservations always be available. In addition to these requirements, *privacy requirements* are of high relevance today. Though the term privacy is often used as a synonym for confidentiality, the two requirements are quite different. Techniques for information confidentiality

- *E. Bertino is with the Computer Science and Electric and Computer Engineering Department and CERIAS, Purdue University, West Lafayette, IN 47907. E-mail: bertino@cerias.purdue.edu.*
- *R. Sandhu is with the Information Science Engineering Department, George Mason University, Fairfax, VA 22030. E-mail: sandhu@ise.gmu.edu.*

Manuscript received 2 Sept. 2004; revised 11 Jan. 2005; accepted 1 Mar. 2005; published online 4 Apr. 2005.

For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-0130-0904.

may be used to implement privacy; however, assuring privacy requires additional techniques, such as mechanisms for obtaining and recording the consents of users. Also, confidentiality can be achieved be means of withholding data from access, whereas privacy is required even after the data has been disclosed. In other words, the data should be used only for the purposes sanctioned by the user and not misused for other purposes.

Data protection is ensured by different components of a database management system (DBMS). In particular, an *access control mechanism* ensures data confidentiality. Whenever a subject tries to access a data object, the access control mechanism checks the rights of the user against a set of authorizations, stated usually by some security administrator. An *authorization* states whether a subject can perform a particular action on an object. Authorizations are stated according to the access control policies of the organization. Data confidentiality is further enhanced by the use of encryption techniques, applied to data when being stored on secondary storage or transmitted on a network. Recently, the use of encryption techniques has gained a lot of interest in the context of outsourced data management; in such contexts, the main issue is how to perform operations, such as queries, on encrypted data [54]. Data integrity is jointly ensured by the access control mechanism and by semantic integrity constraints. Whenever a subject tries to modify some data, the access control mechanism verifies that the user has the right to modify the data, and the semantic integrity subsystem verifies that the updated data are semantically correct. *Semantic correctness* is verified by a set of conditions, or predicates, that must be verified against the database state. To detect tampering, data can be digitally signed. Finally, the recovery subsystem and the concurrency control mechanism ensure that data is available and correct despite hardware and software failures and accesses from concurrent application programs. Data availability, especially for data that are available on the Web, can be further strengthened by the use of techniques protecting against denial-of-service (DoS) attacks, such as the ones based on machine learning techniques [25].

In this paper, we focus mainly on the confidentiality requirement and we discuss access control models and techniques to provide high-assurance confidentiality. Because, however, access control deals with controlling accesses to the data, the discussion in this paper is also relevant to the access control aspect of integrity, that is, enforcing that no unauthorized modifications to data occur. We also discuss recent work focusing specifically on privacy-preserving database systems. We do not cover transaction management or semantic integrity. We refer the reader to [50] for an extensive discussion on transaction models, recovery and concurrency control, and to any database textbook for details on semantic integrity. It is also important to note that an access control mechanism must rely for its proper functioning on some authentication mechanism. Such a mechanism identifies users and confirms their identities. Moreover, data may be encrypted when transmitted over a network in the case of distributed systems. Both authentication and encryption techniques are widely discussed in the current literature on computer network security and we refer the reader to [62] for details on such topics. We will, however, discuss the use of encryption techniques in the context of secure outsourcing of data, as this is an application of cryptography which is specific to database management. We do not attempt to be exhaustive, but try to articulate the rationale for the approaches we believe to be promising.

1.1 A Short History

Early research efforts in the area of access control models and confidentiality for DBMSs focused on the development of two different classes of models, based on the *discretionary access control policy* and on the *mandatory access control policy*. This early research was cast in the framework of relational database systems. The relational data model, being a declarative high-level model specifying the logical structure of data, made the development of simple declarative languages for the specification of access control policies possible. These earlier models and the discretionary models in particular, introduced some important principles [45] that set apart access control models for database systems from access control models adopted by operating systems and file systems. The first principle was that access control models for databases should be expressed in terms of the logical data model; thus authorizations for a relational database should be expressed in terms of relations, relation attributes, and tuples. The second principle is that for databases, in addition to *name-based access control*, where the protected objects are specified by giving their names, *content-based access control* has to be supported. Content-based access control allows the system to determine whether to give or deny access to a data item based on the contents of the data item. The development of content-based access control models, which are, in general, based on the specification of conditions against data contents, was made easy in relational databases by the availability of declarative query languages, such as SQL.

In the area of discretionary access control models for relational database systems, an important early contribution was the development of the System R access control model [51], [42], which strongly influenced access control models of current commercial relational DBMSs. Some key features of this model included the notion of decentralized authorization administration, dynamic grant and revoke of authorizations, and the use of views for supporting content-based authorizations. Also, the initial format of well-known commands for grant and revoke of authorizations, that are today part of the SQL standard, were developed as part of this model. Later research proposals have extended this basic model with a variety of features, such as negative authorization [27], role-based and task-based authorization [80], [87], [47], temporal authorization [10], and context-aware authorization [74].

Discretionary access control models have, however, a weakness in that they do not impose any control on how

information is propagated and used once it has been accessed by subjects authorized to do so. This weakness makes discretionary access controls vulnerable to malicious attacks, such as Trojan Horses embedded in application programs. A Trojan Horse is a program with an apparent or actually useful function, which contains some *hidden* functions exploiting the legitimate authorizations of the invoking process. Sophisticated Trojan Horses may leak information by means of covert channels, enabling illegal access to data. A *covert channel* is any component or feature of a system that is misused to encode or represent information for unauthorized transmission, without violating the stated access control policy. A large variety of components or features can be exploited to establish covert channels, including the system clock, operating system interprocess communication primitives, error messages, the existence of particular file names, the concurrency control mechanism, and so forth. The area of mandatory access control and multilevel database systems tried to address such problems through the development of access control models based on information classification, some of which were also incorporated in commercial products. Early mandatory access control models were mainly developed for military applications and were very rigid and suited, at best, for closed and controlled environments. There was considerable debate among security researchers concerning how to eliminate covert channels while maintaining the essential properties of the relational model. In particular, the concept of polyinstantiation, that is, the presence of multiple copies with different security levels of the same tuple in a relation, was developed and articulated in this period [81], [55]. Because of the lack of applications and commercial success, companies developing multilevel DBMSs discontinued their production several years ago. Covert channels were also widely investigated with considerable focus on the concurrency control mechanisms that, by synchronizing transactions running at different security levels, would introduce an obvious covert channel. However, solutions developed in the research arena to the covert channel problem were not incorporated into commercial products. Interestingly, however, today we are witnessing a "multilevel security reprise" [82], driven by the strong security requirements arising in a number of civilian applications. Companies have thus recently reintroduced such systems. This is the case, for example, of the Labeled Oracle, a multilevel relational DBMS marketed by Oracle, which has much more flexibility in comparison to earlier multilevel secure DBMSs.

Early approaches to access control have since been extended in the context of advanced DBMSs, such as object-oriented DBMSs and object-relational DBMSs, and other advanced data management systems and applications, such as data made available through the Web and represented through XML, digital libraries and multimedia data, data warehousing systems, and workflow systems. Most of these systems are characterized by data models that are much richer than the relational model; typically, such extended models include semantic modeling notions such as inheritance hierarchies, aggregation, methods, and stored procedures. An important requirement arising from those applications is that it is not only the data that needs to be protected, but also the database schema may contain sensitive information and, thus, accesses to the schema need to be filtered according to some access control policies. Even though early relational DBMSs did not support authorizations with respect to schema information, today several products support such features. In such a context, access control policies may also need to be protected because they may reveal sensitive information. As such, one may need to define access control policies the objects of which are not user data, rather they are other access control policies. Another relevant characteristic of advanced applications is that they often deal with multimedia data, for which the automatic interpretation of contents is much more difficult, and they are in most cases accessed by a variety of users external to the system boundaries, such as through Web interfaces. As a consequence both discretionary and mandatory access control models developed for relational DBMSs had to be properly extended to deal with additional modeling concepts. Also, these models often need to rely on metadata information in order to support content-based access control for multimedia data and to support credential-based access control policies to deal with external users. Recent efforts in this direction include the development of comprehensive access control models for XML [14], [72].

1.2 Emerging Research in Database Security

Besides the historical research that has been conducted in database security, several new areas are emerging as active research topics. A first relevant recent research direction is motivated by the trend of considering databases as a service that can be outsourced to external companies [54]. An important issue is the development of query processing techniques for encrypted data. Several specialized encryption techniques have been proposed, such as the order-preserving encryption technique by Agrawal et al. [3]. A second research direction deals with privacy-preserving techniques for databases, an area recently investigated to a considerable extent. Research in this direction has been motivated, on one side, by increasing concerns with respect to user privacy and, on the other, by the need to support Web-based applications across organization boundaries. In particular privacy legislation, such as the early Federal Act of 1974 [43] and the more recent Health Insurance Portability and Accountability Act of 1996 (HIPAA) [53] and the Children's Online Privacy Protection Act (COPPA) [33], require organizations to put in place adequate privacy-preserving techniques for the management of data concerning individuals. The new Web-based applications are characterized by the requirement of supporting cooperative processes while ensuring the confidentiality of data. This research direction is characterized by a number of different approaches and techniques, including privacy-preserving data mining [92], privacy-preserving information retrieval, and databases systems specifically tailored toward enforcing privacy [2].

1.3 Organization of the Paper

The remainder of the paper is organized as follows: Section 2 discusses past and current developments for relational database systems. It discusses both discretionary and mandatory access control models and also briefly surveys other topics such as RBAC models. Section 3 presents an overview of relevant requirements for access control models for advanced data management systems and outlines the main approaches, including access control systems for XML. Section 4 summarizes privacy-preserving data management techniques, which are the focus of several research efforts today, and Section 5 discusses current factors and trends which make database security more challenging. Finally, Section 6 presents some concluding remarks.

2 Relational Database Systems

2.1 Discretionary Access Control for Relational Databases

Access control mechanisms of current DBMSs are based on discretionary policies governing the accesses of a subject to data based on the subject's identity and authorization rules. These mechanisms are discretionary in that they allow subjects to grant authorizations on the data to other subjects. Because of such flexibility, discretionary policies are adopted in many application environments and this is the reason that commercial DBMSs adopt such policies. An important aspect of discretionary access control is thus related to the *authorization administration policy*. Authorization administration refers to the function of granting and revoking authorizations. It is the function by which authorizations are entered into or removed from the access control mechanism. Common administration policies include *centralized administration*, by which only some privileged subjects may grant and revoke authorizations, and *ownership administration*, by which grant and revoke operations on data objects are entered by the creator (or owner) of the object. Ownership-based administration is often provided with features for *administration delegation*, allowing the owner of a data object to assign other subjects the right to grant and revoke authorizations. Delegation thus supports decentralized authorization administration. Most commercial DBMSs adopt ownership-based administration with administration delegation. More sophisticated administration mechanisms can be devised such as *joint administration*, by which several subjects are jointly responsible for authorization administration [17].

In this section, we review some discretionary models proposed for relational DBMSs. We start by describing the System R authorization model and then we survey some recently proposed extensions to it. We then discuss role-based access control (RBAC), a relevant extension to current authorization models, which finds application not only to database systems, but also to the more general context of enterprise security [60] and of multidomain systems [28].

2.1.1 The System R Authorization Model and Its Extensions

One of the first authorization models developed for relational DBMSs was defined by Griffiths and Wade [51], [42] in the framework of the System R DBMS [6]. Under this model, protection objects are tables and views, also referred to as virtual tables.[1] The possible access modes that subjects can exercise on tables correspond to SQL operations that can be executed on tables. Thus, relevant access modes include: *select* (to retrieve tuples from a table), *insert* (to add tuples to a table), *delete* (to remove tuples from a table), and *update* (to modify tuples in a table). The same access modes are defined for views with the difference that some access modes may not be applicable to a view depending on the view definition. For example, very often, delete, insert, and update operations are not allowed on views defined as joins or containing aggregate functions. In the remainder, we use the term table to refer to both base tables and views. It is important to point out that this basic model is still prevalent today in commercially available DBMSs. Of course, current DBMSs have extended the basic model by introducing new types of objects to be protected as a consequence of extensions to the data model, and the set of protection modes that one finds in such DBMSs is much larger than the set defined as part of the basic model. For example, the introduction of trigger mechanisms in relational DBMSs [93] has required the introduction of a specific access mode allowing a subject to create a trigger on a table. Similarly, the introduction of mechanisms for referential integrity through the use of foreign key has required the introduction of a related access mode allowing a subject to reference a table from another table.

Authorization administration in the System R model is based on the ownership approach coupled with administration delegation. Any database user authorized to do so can create a new table. When a user creates a table, he becomes the owner of the table and is solely and fully authorized to exercise all access modes on the table. The owner, however, can delegate privileges on the table to other subjects by granting these subjects authorizations with the so-called *grant option*. The possibility of delegating authorization administration introduces some interesting issues concerning the semantics of the revoke operations. A subject, to whom the administration right on a given table has been granted and then revoked, may have granted to another subject an authorization to access the table. The question is what happens to this authorization when the revokation takes place. The semantics of the revokation of an authorization from a subject (revokee) by another subject (revoker) is to consider as valid only the authorizations that would have been present had the revoker never granted the revokee the privilege. As a consequence, every time an authorization is revoked from a subject, a recursive revocation takes place to remove all authorizations for this

1. There are usually other objects to be protected in a database, such as application programs and stored procedures. We limit the discussion to tables and views to simplify the presentation.

table from the revokee. The revoke operation takes into account the temporal sequence according to which the grant operations were made. The temporal sequence is determined according to the timestamps that are associated with the granted authorizations.

A number of extensions to the basic model have been proposed with the goal of enriching the expressive power of the authorization languages in order to address a large variety of application requirements. A first extension deals with negative authorizations [27]. The System R authorization model, as the models of most DBMSs, uses the *closed world* policy. Under this policy, whenever a subject tries to access a table and no authorization is found in the system catalogs, the subject is denied access. Therefore, the lack of authorization is interpreted as no authorization. This approach has the major drawback that the lack of an authorization for a subject on a table does not prevent this subject from receiving this authorization some time in the future. Any subject holding the right to administer that table can grant any other subject the authorization to access the table. The introduction of *negative authorization* can overcome this drawback. An explicit negative authorization expresses a *denial* for a subject to access a table under a specified mode. Conflicts between positive and negative authorizations are resolved by applying the *denials-take-precedence* policy under which negative authorizations override positive authorizations. That is, whenever a subject has both a positive and a negative authorization for a given privilege on a table, the subject is prevented from exercising the privilege on the table. The subject is denied access even if a positive authorization is granted after a negative one has been granted. Negative authorizations can also be used to temporarily block possible positive authorizations of a subject and to specify exceptions. For example, it is possible to grant an authorization to all members of a group, but for one specific member, by granting the group a positive authorization for the privilege on the table and the given member the corresponding negative authorization. Such a model has been further extended with a more flexible conflict resolution policy, based on the concept of *more specific* authorization. Such a concept introduces a partial order relation among authorizations which is taken into account when dealing with conflicting authorizations. For example, the authorizations granted directly to a user are more specific than the authorizations granted to the groups of which the user is a member. Therefore, a negative authorization can be overridden by a positive authorization, if the latter is more specific than the former. If, however, two conflicting authorizations cannot be compared under the order relation, the negative authorization prevails. This line of work has been further extended by several other researchers and today we find a variety of approaches dealing with conflict resolution policies and with logical formalizations of access control policies. Such logical formalizations provide sound underlying semantics which is essential when dealing with complex access control models [16].

The notion of explicit denial has also been proposed in the context of the Sea View system [59]. In Sea View, authorizations can specify which users or groups are authorized to access particular tables and which users and groups are specifically denied for particular tables. Unlike positive authorizations, negative authorizations cannot specify an access mode. A special access mode, called "null," is used to denote a negative authorization. If a subject receives a null access mode on a table, the subject cannot exercise any access mode on the table. Conflicts between positive and negative authorizations are solved on the basis of the following policy: 1) authorizations directly granted to a user take precedence over authorizations specified for groups to which the user belongs and 2) a null mode authorization given to a subject overrides any other authorization granted to the same subject. Thus, negative authorizations always override positive authorizations. It is of interest to remark here that explicit denials have been also introduced in operating systems, e.g., Windows, as a mechanism for expressing exceptions. In such a context, specifying that a subject can access all the files in a directory, but one specific file can be concisely expressed by two authorizations, one giving the subject a positive authorization to the directory and all the files contained in it, and another one specifying an explicit denial on the specific file to which access from this subject has to be precluded.

A second major extension deals with a more articulated semantics for the revoke operation [95]. In the System R model, as in all DBMSs, whenever an authorization is revoked from a subject, a recursive revocation takes place. This approach can be very disruptive. In many organizations, the authorizations a user possesses are related to his particular task or function within the organization. If a user changes his task or function, it is desirable to remove only the authorizations of this user without triggering a recursive revocation of all the authorizations granted by this user. To support this requirement, a different kind of revoke operation called *noncascading revoke* has been proposed. Whenever a noncascading revoke operation is executed, the authorizations granted by the user from whom the authorization is being revoked are not revoked; instead, they are respecified as if they had been granted by the user requiring the revocation. Thus, all authorizations granted by the revokee to other users remain in place. By providing two different types of revoke operations, cascading and noncascading, the resulting access control system is able to better support a large variety of application requirements. A different approach to overcome the drawbacks of conventional revoke operations is represented the use of RBAC, which by introducing the notion of role and assigning authorizations to roles instead of directly to users, greatly simplifies administration management and reduces the need for recursive revoke operations (see Section 2.1.3).

A third extension is related to the duration of authorizations. In all systems, an authorization is *valid* from the time it is entered into the system, by a grant operation, until it is explicitly removed by a revoke operation. In many

applications, however, permissions may hold only for specific time intervals. A further requirement concerns periodic authorizations. In many organizations, authorizations given to users must be tailored to the pattern of their activities within the organization. Therefore, users must be given access authorizations to data only for the time periods in which they are expected to need the data. We can consider this requirement as an instantiation of the well known "need-to-know" security principle. An example of policy with temporal requirements is that "all programmers can modify the project files every working day except Friday afternoons." In most current DBMSs, such a policy would have to be implemented as code in application programs. Such an approach makes it very difficult to verify and modify the access control policies and to provide assurance that these policies are actually enforced. An authorization model addressing such requirements has been recently proposed [10]. Under such a model, each authorization has a temporal interval of validity; an authorization is valid only in this interval. When the interval expires, the authorization is automatically revoked without requiring any explicit revoke operations from the security administrator. The interval associated with an authorization may also be periodic, thus consisting of several intervals which are repeated in time. In addition, the model provides deductive temporal rules supporting the automatic derivation of new authorizations based on the presence or absence of other authorizations in specific time periods. The resulting model provides a high degree of flexibility and is able to meet a large number of protection requirements that cannot be met by traditional access control models.

The previous temporal authorization model represents one of the earliest proposals recognizing the need for *context-based access control*; time can indeed be seen as a special contextual condition. A context-based access control model is able to incorporate into access control decision functions a large variety of context-dependent information, such as time and location. In addition to being investigated as part of research projects [8], context-based access control has been recently incorporated in the Oracle commercial DBMS [74], through the notion of a *virtual private database*. A virtual private database allows fine-grained access control down to the tuple level based on the use of predicates. The predicates, specified as part of an access control policy, identify the tuples, in a given table, to which the access control policy applies. Whenever a user, to whom the access control policy is granted, issues a query against the table, the DBMS transparently modifies the query by appending to it the predicates specified in the access control policies. Because such predicates can be expressed also against some special system variables, such as SYSDATE, such an approach allows one to take context-dependent information into account when specifying policies. Such a mechanism is complemented by the notion of *application context*. Each *application context* has a unique identifier and consists of a number of attributes, identifying security-relevant properties. The attributes that are part of a given context are specified by the application developer and can refer to any relevant information, such as the organizational position of or the geographical location of the user. Predicates against such attributes can be specified as part of access control policies and, thus, they concur to define a virtual private database. Notice that several contexts can be defined for the same table, each related to different application sectors from which the table is accessed.

2.1.2 Content-Based and Fine-Grained Access Control

Content-based access control is an important requirement that any access control mechanism for use in a data management system should satisfy. Essentially, content-based access control requires that access control decisions be based on data contents. Consider an example of a table recording information about employees of a company; a content-based access control policy would be the one "stating that a manager can only access the employees that work in the project that he manages." Whenever a manager issues a query, the system has to filter the query result by returning only the tuples related to the employees that verify the condition of working in the project managed by this manager. Support for this type of access control has been made possible by the fact that SQL is a language for which most operations for data management, such as queries, are based on declarative conditions against data contents. In particular, the most common mechanism, adopted by relational DBMSs to support content-based access control is based on the use of *views*; this important use of views was recognized by the differentiation of views into two categories [24]: *protection views* specifically tailored to support content-based access control and *shorthand views* specifically tailored to simplify query writing. A view can be considered as a dynamic window able to select subsets of column and rows; these subsets are specified by defining a query, referred to as a *view definition query*, which is associated with the name of the view. Whenever a query is issued against a view, the query is modified through an operation called *view composition* by replacing the view referenced in the query with its definition. An effect of this operation is that the "where clause"[2] in the original query is combined, through the AND Boolean connective, with the "where clause" of the view definition query. Thus, the query which is executed against the base table, that is, the table on which the view is defined, filters out the tuples that do not satisfy the predicates in the view. There are several advantages to such an approach. Content-based access control policies are expressed at a high level in a language consistent with the query language. Modifications to the data do not need modification to the access control policies; if new data are entered that satisfy a given policy, these data will be automatically included as part of the data returned by the corresponding view.

Recently, pushed by requirements for fine-grained mechanisms that are able to support access control at the

2. The "where clause" is the clause containing predicates against tables and is a component of several SQL commands, such as Select, Update, and Delete.

tuple level, new approaches have been investigated. The reason is that conventional view mechanisms, like the ones sketched above, have a number of shortcomings. A naive solution to enforce fine-grained authorization would require the specification of a view for each tuple or part of a tuple that is to be protected. Moreover, because access control policies are often different for different users, the number of views would further increase. Furthermore, as pointed out in [78], application programs would have to code different interfaces for each user, or group of users, as queries and other data management commands would need to use for each user, or group of users, the correct view. Modifications to access control policies would also require the creation of new views with consequent modifications to application programs. Alternative approaches that address some of these issues have been proposed, and these approaches are based on the idea that queries are written against the base tables and, then, automatically rewritten by the system against the view available to the user. The Oracle Virtual Private Database mechanism [74] and the Truman model [78] are examples of such approaches. These approaches do not require that we code different interfaces for different users and, thus, address one of the main problems in the use of conventional view mechanisms. However, they introduce other problems, such as inconsistencies between what the user expects to see and what the system returns; in some cases, they return incorrect results to queries rather than rejecting them as unauthorized. Approaches that address this problem, as the solutions proposed as part of the Truman model [78], have some decidability problems and, thus, do not appear to be applicable in practice. Thus, different solutions need to be investigated.

2.1.3 RBAC Models

RBAC models represent arguably the most important recent innovation in access control models. RBAC has been motivated by the need to simplify authorization administration and to directly represent access control policies of organizations. RBAC models are based on the notion of *role*. A role represents a specific function within an organization and can be seen as a set of actions or responsibilities associated with this function. Under an RBAC model, all authorizations needed to perform a given activity are granted to the role associated with that activity, rather than being granted directly to users. Users are then made members of roles, thereby acquiring the roles' authorizations. User access to objects is mediated by roles; each user is authorized to play certain roles and, on the basis of the roles, he can perform accesses to the objects. Because a role groups a number of related authorizations, authorization management is greatly simplified. Whenever a user needs to perform a certain activity, the user only needs to be granted the authorization of playing the proper role, rather than being directly assigned the required authorizations. Also, when a user changes his function within the organization, one only needs to revoke from the user the permission to play the role associated with the function. Complicated authorization revoke operations, such as the ones discussed in the previous sections, are no longer needed.

In addition, most RBAC models include *role hierarchies*, allowing one to represent role-subrole relationships, thus enabling authorization inheritance and *separation of duty (SoD) constraints* [5], [67]. SoD constraints typically prevent a subject from receiving too many authorizations. If a user that has a large number of authorizations is compromised —for example, by a malicious subject impersonating that user—the entire database would be compromised. It is thus preferable to spread authorizations among different subjects; in this case, the compromise of a subject would result in limited compromise of the database. Also, separation of conflicting permissions such as ability to cut checks and to issue purchase orders is crucial for reducing the potential for fraud in organizations. RBAC SoD constraints, represented in terms of constraints on the roles that users may take, are often classified into *static* and *dynamic* SoD. Static SoD typically impose restrictions on role intersections—two roles cannot have common users—and on the number of users that can be assigned to a role—a given role can only be assigned to two users. Dynamic SoD constraints are based on the history of role usage by users. Their enforcement is related to the notion of a *session*, which is another important notion underlying the RBAC model. A session represents a set of accesses performed by a user under one or more roles that can be considered as an atomic unit of work. A session could be a transaction execution in a conventional relational database system, or a task in a workflow. Dynamic SoD essentially restricts access to roles by a user based on the history of role usage by the user during the same session, or even, in some proposals, during previous sessions. As such roles can be considered as another type of "context sensitive" relation; an important research issue when dealing with SoD constraints is the verification of their consistency, especially when dealing with large constraint sets.

RBAC models have been widely investigated [48]. A standard has been developed [47] as well as an XML-based encoding of RBAC [28]. Relevant extensions include: the development of administration models [34], [63], [65]; the introduction of temporal constraints, resulting in the TRBAC model [11], [68]; and the development of security analysis techniques [56]. RBAC models are also supported by commercial DBMSs [76]. However, commercial implementations provided as part of DBMSs are very limited and only support a simple version of RBAC, referred to as *flat RBAC*, that does not include role hierarchies or constraints. Finally, it is worth mentioning that RBAC systems are also being developed for use in Web-service architectures, such as the Permis system [31], and as part of products for enterprise security management [61].

2.2 Mandatory Access Control and Multilevel Secure DBMSs

Mandatory access control (MAC) policies regulate accesses to data by subjects on the basis of predefined classifications

of subjects and objects in the system. Objects are the passive entities storing information, such as relations, tuples in a relation, or elements of a tuple. Subjects are active entities performing data accesses. The classification is based on a partially ordered set of *access classes*, often referred to as *labels*, that are associated with every subject and object in the system. A subject is granted access to a given object if and only if some order relationship, depending on the access mode, is satisfied by the access classes of the object and the subject. In a very well-known instantiation of this model [9], an access class consists of two components: a *security level* and a *set of categories*. The security level is an element of a totally ordered set. A well-known example of such set is the one that contains the levels Top Secret (TS), Secret (S), Confidential (C), and Unclassified (U), where $\mathrm{TS} > \mathrm{S} > \mathrm{C} > \mathrm{U}$. The set of categories is an unordered set (e.g., NATO, Nuclear, Army). Access classes are partially ordered as follows: An access class c_i dominates ($\geq$) an access class c_j if and only if the security level of c_i is greater than or equal to that of c_j and the categories of c_i include those of c_j. Two classes are said to be incomparable if neither $c_i \geq c_j$ nor $c_j \geq c_i$ holds. The security level of the access class associated with a data object reflects the sensitivity of the information contained in the object, that is, the potential damage that could result from unauthorized disclosure of the contents of the object. The security level of the access class associated with a subject reflects the user's trustworthiness not to disclose sensitive information to subjects not cleared to see it. Categories provide finer grained security classifications of subjects and objects than the classification provided by security levels alone, and are the basis for enforcing *need-to-know* restrictions. Denning [36] developed the mathematical theory that underlies such lattices and a comprehensive survey and discussion is given in [79].

Access control in MAC models is based on the following two principles, formulated by Bell and LaPadula in 1975 [9]:

No read-up. A subject can read only those objects whose access classes are dominated by the access class of the subject.

No write-down. A subject can write only those objects whose access classes dominate the access class of the subject.

The enforcement of these principles prevents information in a sensitive object from flowing, through either read or write operations, into objects at lower or incomparable access classes.

The application of MAC policies to relational databases has been extensively investigated in the past. The introduction of such access control models requires addressing several difficult issues. Solutions to some of these issues have required extensions to the definition of the relational model itself, resulting in the so-called *multilevel relational model*, and to fundamental notions such as the notion of relational key. A multilevel relation is characterized by the fact that different tuples may have different access classes. The relation is thus partitioned into different security partitions, one for each access class. A partition associated with an access class c contains all tuples whose access class is c. A subject having access class c can read all tuples in partitions of access classes that are equal to or lower than c; such a set of tuples is referred to as a *view* of the multilevel relation at access class c. By contrast, a subject having access class c can write tuples at access classes that are equal or higher than c. In some implementations of the multilevel relational model, write operations at higher access classes are not allowed for integrity reasons. Such a restriction is usually known as a no write-up restriction. The multilevel relational model is further complicated if tuples are allowed to have attributes classified at different access classes. Each attribute of each tuple thus has an *attribute label*, denoting the access class of the attribute in the tuple, and a *tuple label*, which is the lowest element in the set of access classes associated with the attributes of the tuple. A consequence is that the same tuple may belong to several partitions of a multilevel relation, resulting in tuple *polyinstantiation* and, thus, in update anomalies. Handling polyinstantiation requires revisiting several classical notions of the relational model, such as the notion of a key. Because of such problems, commercial implementations of the multilevel relational model only support tuple-based labeling.

The development of multilevel secure (MLS) DBMSs entailed, however, extending not only the data model, but also the system architecture to make sure that covert channels would be closed [39]. A *covert channel* allows a transfer of information that violates the security policy. Covert channels are usually classified into two broad categories: timing channels, under which information is conveyed by the timing of events or processes; and storage channels that do not require any temporal synchronization in that information is conveyed by accessing system information. A well-known type of covert channel in a DBMS is represented by the 2-phase locking (2PL) protocol used for transaction synchronization [15]. Much academic research has been thus devoted to the development of concurrency control mechanisms that are secure against covert channels. Most of these approaches were based on the principle that transactions cannot be delayed or aborted due to a lock conflict with a higher-level transaction. Hence, low-level transactions have higher priority on low-level data than higher-level transactions. The consequence is that even though a transaction may have acquired a read lock on a lower-level data item, it may be forced to release this lock if a lower-level transaction requires a write lock on it. Due to such prioritization, transaction execution histories may not always be serializable. Several approaches have been proposed to address the issue of how to synchronize transactions so that timing channels do not occur and, at the same time, serializability is achieved. However, they suffer from several shortcomings, such as starvation of high-level transactions that can be repeatedly aborted, or require multiple versions of data, or force high-level transactions to read stale data. A different approach [14] was later defined based on application-level recovery and notification-based locking protocols combined with a nested transaction model [70].

We conclude this section by mentioning that multilevel access control models have also been applied to commercial relational DBMSs both in the past in products such as Trusted Oracle and Secure Informix and more recently. The most recent extension of a commercial product supporting MAC is the *label security* mechanism introduced in Oracle9i [74]. Such a mechanism allows the application developers to associate classification labels with both data and users, and to apply MAC access control policies. The labeling granularity supported by this mechanism is a row; thus, labels can only be associated with tuples and not with single attributes within tuples. Labels in Oracle have quite an articulated structure, as each label consists of three elements. In addition to the classical security level and category (referred in Oracle as *compartment*) set components, a label includes a third component, referred to as *group*. The group specifies one or more subjects that own or access the data. Furthermore, groups can be organized according to hierarchies. Labels and all their components can be defined by the applications and, thus, one can introduce levels, categories, and groups that are application-specific. Each user is associated with a label range, denoting a set of access classes, within which the user can read and write data. Finally, it is worth mentioning that, though secure concurrency control algorithms were widely investigated, most of the proposed concurrency control algorithms did not find their way into commercial DBMSs. The only concurrency control algorithm of a commercial DBMS which is documented by the scientific literature was based on a combination of 2PL protocol and multiversioning and was adopted in the Trusted Oracle product. Such an algorithm however was proven incorrect in that it would generate nonserializable transaction schedules.

3 Security for Advanced Data Management Systems

Though the relational database technology has today a central role to play in the data management arena, in the past 20 years, we have seen numerous extensions to this technology. These extensions have been driven on one hand by requirements from advanced applications, needing to manage complex, multimedia objects, and from decision-support systems, requiring data mining techniques and data warehousing systems, and on the other hand by the widespread use of Internet and Web-based applications, that have fueled the development of interoperability approaches, like XML and Web services. A key requirement underlying all those extended data management systems and tools is a demand for adequate security and, in particular, tailored access control systems. Relevant features of such systems include:

- **Fine-grained flexible authorization models for complex, multimedia objects**. Most innovative applications are characterized by objects whose structure is far more complex than the simple flat structure typical of relational data. This is the case, for example, of XML data [14] and object database systems, such object-oriented (OO) and object-relational (OR) database systems [75], [41].[3] Because applications may directly access data at various granularity levels from sets of data objects to specific portions of a single data object, mechanisms are needed to control access at varying granularity levels and to be able, at the same time, to support concise formulation of authorizations. Typical extensions that have been proposed to address such requirements include the notions of positive/negative authorizations, and implicit/explicit authorizations [44] that we discuss in the context of access control models for object-based systems. The presence of multimedia data makes content-based access control very difficult and, to date, the few proposed models are based on the use of metadata information [20], [66] rather than directly on the object contents.
- **Flexible user specification mechanisms based on user credentials and profiles**. Most Web-based applications are characterized by a user population which is far more heterogeneous and dynamic than the user population typical of conventional information systems. In such a scenario, traditional identity mechanisms, based on login or user names, for qualifying the subjects to which a policy applies are no longer appropriate in that they would require the specification and management of a large number of policies. There is thus the need for using other properties of subjects (e.g., age, nationality, job position) besides their login names, in the specification and enforcement of access control policies. Such properties that can be considered as a form of *partial identity* are often encoded into user profiles and certified by means of credentials and attribute certificates.
- **Access control mechanisms tailored to information dissemination strategies and third party publishing architectures**. An important requirement of today's Web-based information systems is to support a variety of *information dissemination strategies* [40]. A dissemination strategy regulates how a data source delivers data to subjects. In conventional database systems, data are delivered according to a strategy known as *pull strategy*. According to such a strategy, data are delivered to subjects upon an explicit request. However, in a Web environment, an alternative strategy can be adopted, which is more suitable when information has to be delivered to a large community of subjects. According to such strategy, referred to as *push strategy* or as *publish/subscribe*, the data source periodically (or when some

3. Object-oriented DBMSs, often referred to as pure object DBMSs, refer to systems developed by starting directly from object-oriented programming languages, such as GemStone and ObjectStore, as opposed to object-relational DBMSs which are essentially relational DBMSs extended with object modeling features. The term object-based DBMSs is used when it is not necessary to distinguish between the two types of systems.

predefined events happen) sends data to authorized subjects, without the need of an explicit access request by the subjects. In some cases, the data that are sent to subjects also depend on the specific subject interests, that are recorded in some special subject profiles managed by the data source [98]. Supporting different dissemination strategies may require the adoption of different access control techniques depending on the data dissemination strategy adopted. A comprehensive access control system should thus provide a large variety of access control techniques able to enforce a given policy under a variety of dissemination strategies.

Because of the relevance of efficient information dissemination in a large variety of environments, not only several dissemination strategies have been developed, but also approaches supporting third-party information publishing architectures have been proposed [13]. The main idea is that an organization producing and owning some data may outsource the publishing function to a third-party, which would typically be in charge of executing user queries; a well-known example is that of UDDI registries managing information concerning services provided by organizations on the Web. The main issue here is how to ensure the integrity and confidentialiy of data when their publication is outsourced to other parties.

- **Support for distributed cooperative data modifications and complex workflow-based activities**. The Web has enabled a new class of applications, including B2B and B2C, virtual organizations, e-contracting, and e-procurement, that are characterized by the need of collaborative processes across organization's boundaries. Such applications require not only data being securely exchanged, but also that *data flow policies* be specified, stating which party has to receive and/or modify data according to which order. Also, protocols are required allowing a party to verify that a given piece of data has been modified by subjects, that have accessed the data as part of a cooperative process, according to the stated access control policies.

In the remainder of this section, we elaborate on the above features and requirements by discussing solutions proposed by various systems and research proposals. We start by first discussing object-based DBMS, in the context of which several innovative solutions for access control had been developed. Though object-oriented DBMSs have not been very successfull from a commercial point of view, the development of access control models suitable for these systems required to address a large number of novel issues arising from the extended complexity of the data models characterizing such DBMSs. Several of these solutions can be directly applied to more recent ORDBMSs and to XML data, as we discuss in Section 3.2, and in general to complex data. It is important to notice that to date the potential application of these solutions to XML data has not been fully explored.

3.1 Access Control Systems for Object-Based Database Systems

As we mentioned in the introduction, today, access control systems are a basic component of every commercial DBMS. Existing access control models, defined for relational DBMSs, are not suitable for an object-based database system because of the wide differences in data models. These models, in particular the discretionary ones, consider the relation, or the attribute as the access control unit, in the sense that authorizations are granted on relations or, in some cases, on relation attributes. Moreover, an access control system for object-based database systems should take into account all semantic modeling constructs commonly found in object-oriented data models, such as composite objects, versions, and inheritance hierarchies. We can summarize these two observations by saying that the increased complexity in the data model corresponds to an increased articulation in the types and granularity of protection objects. In particular, as we will discuss in the remainder of this section, a key feature of both discretionary and mandatory access control models for object-based systems is to take into account all modeling aspects related to objects.

3.1.1 Discretionary Access Control Systems for Object-Based Database Systems

The first and most comprehensive discretionary access control model has been defined in the context of the Orion object-oriented DBMS [75]. Other systems implement less sophisticated models or have no access control at all. A key aspect of the Orion authorization model is the use of authorization implication rules supporting the derivation of additional authorizations, called *implicit authorizations*, from the ones explicitly specified by the application, called *explicit authorizations*. Implication rules are defined for all the three domains of authorizations, that is, objects, subjects, and modes. In particular, implication rules on objects support the derivation of authorizations from an object to all objects semantically related to it. For example, a read authorization on the root of a version hierarchy[4] implies read authorizations on all the versions in the hierarchy. However, it is also possible for an authorization to be granted on a single version of an object. The use of implication rules is instrumental in providing varying granularity levels of protection without performance penalties. The Orion model also supports negative authorizations; the main purpose of this type of authorization is the support for exceptions in derived authorizations. In particular, the combined use of derived and negative authorization allows one to concisely express a large number of access control policies. For example, consider a class with 1,000 instances; suppose that a subject has to be authorized to access all those instances except one. Under a

4. A version hierarchy consists of an object and all the version objects that have been derived directly or indirectly from it.

conventional authorization model one would have to enter 999 authorizations. Under the Orion model, one would need to enter only two authorizations, that is, a positive authorization on the class, which would automatically propagate to all instances, and a negative authorization on the instance to be excluded. It is important to notice that, in Orion, authorization implication is only possible among objects that are related by structural semantic relationships specified according to the data model. Recently, the notion of derived authorizations has been extended in the context of logic-based access control models to support arbitrary authorization derivation rules, not necessarily based only on the structural relationships among objects. The Orion authorization model also provides the notion of *authorization object schema (AOS)*, modeled as a graph, to represent all database granule types, modeled as nodes, and structural relationships among these granule types, modeled as edges. The notion of AOS, which can be considered as part of a metaschema for the authorization model, has been recently applied to the representation of an access control model for XML data [94].

The above authorization model could be termed "structural authorization model" in that it does not exploit the *encapsulation* property typical of object systems. Encapsulation is, however, one of the most important features of the object-oriented paradigm. Encapsulation entails a separation between an object's status and interface. Such separation enables the *clients* of an object to use the services provided by the object without having to be aware of how the services are implemented (*information hiding*). Therefore, an object's implementation may change without impacting other objects or applications that use the services provided by the object. The information hiding capability has, in addition, a great potential for data protection. By "surrounding" an object with methods, it is possible to interpose an additional layer between the object and its users. Therefore, arbitrary complex content-based access policies can be supported. In particular, a relational DBMS typically allows a user to develop an application program and then grant the `run` authorization on this program to other users. The users receiving authorizations on a program do not usually need to have the authorizations on the data accessed by the program, as these authorizations are checked against the *program owner*. In this way, it is possible to support authorizations on an application basis. Methods in object-oriented databases could be used in the same way, thus providing an extensible authorization mechanism. However, the use of methods for authorization differs with respect to the use of application programs in the following aspect. When application programs are used, application-dependent access rules tend to be dispersed among the various application programs. Therefore, it is more difficult to verify that the correct authorization policies are applied and moreover modifications to these policies may require extensive changes in the application code. By contrast, methods are tightly coupled with data objects. Application-dependent access rules, of arbitrary complexity, are thus centralized and all redundancies eliminated. Therefore, since an object-oriented approach enforces the principle that changes to method implementation and object structures should not impact the clients of an object, it is possible to modify access rules without requiring changes to the clients. Of course, clients must be able to deal with exceptions arising from the lack of authorization. Note that the possibility of dynamically modifying access rules is a direct consequence of the fact that, in an object-oriented database, some of the high-level operations on data are moved into the data. Moving these operations into the database, by implementing them as methods, implies that access rules implement as part of these operations are also moved into the database. Thus, access rules are centralized and applied to all accesses made to objects. A number of authorization models have been developed based on the use of methods; among these the most notable are the models by Ahad et al. [4], exploiting the notions of *guard functions* and *proxy functions* to enforce content-based access control, and by Richardson et al. [77], providing the concepts of *method implementor*—the user who has written the method's code—and *method principal*—the user on whose behalf the method is executed.

A similar trend can also be observed in object-relational DBMSs which today provide functions for managing stored procedures that, very much like object methods, are stored and centralized in the database. Even though stored procedures are not usually associated with strong encapsulation principles, they can be very much used to provide an additional layer of access control and to implement arbitrarily complex access control. In particular, the use of stored procedures for improving database security is often recommended among best practices for protecting databases against various types of threats, such as SQL injection [12]. However, the use of stored procedures requires making sure that only those stored procedures are used whose origin and behavior are well-known.

3.1.2 Mandatory Access Control Systems for Object-Based Database Systems

The application of a typical MAC model to object-based systems in not straightforward, due to the semantic richness of object data models. Moreover, the differences both in theory and implementation among the various OODBMSs and ORDBMSs makes it very difficult to define sound and general principles upon which a suitable MAC model can be based. To date the problem of MAC models for object-based database systems has been investigated only in the context of object-oriented databases; no work has been reported dealing specifically with object-relational databases. However, despite such difficulties, the use of an object-oriented approach offers an important advantage with respect to mandatory policies. In particular, the fact that messages are the only means by which objects exchange information makes *information flow* [36] in object systems have a natural and direct representation in terms of message exchange among objects. By properly filtering messages among objects, according to the specified access

control policies, it is possible to develop effective approaches to access control enforcement.

MAC models can be classified in two main categories: *single-level models* and *multilevel models*. Models in the first category require that an object and all its features, e.g., attributes and methods, be classified at the same access class. Models in the second category do not impose such a restriction; however, they are rather difficult to implement in practice. Most proposed models are thus single-level. The main reason is the simplicity of such an approach and its compatibility with a security kernel. By using an underlying security kernel for the enforcement of MAC properties, the layer implementing the object data management system need not be trusted. The main drawback of single-level models, despite their simplicity of implementation, is that applications often need objects that are multilevel. In order to accommodate such applications, the most common approach is to use a single-level object system and map the multilevel application objects onto several single-level objects. This approach, first proposed by Thuraisingham in a seminal paper [89] and referred to as *multilevel object view approach*, has two variants depending on whether inheritance or aggregation is used to support the multilevel view. Real multilevel object models are more difficult to handle and no satisfactory approach has been proposed.

3.2 Access Control Systems for XML

XML [96] is today widely used in a large variety of applications and industry products as it has become the standard for describing data and documents circulated across the Web. The most important feature of XML that distinguishes it from other markup languages such as HTML is the notion of semantic tags, allowing one to mark different portions, called *elements*, of a given data item and to assign to them names that are semantically meaningful. XML can thus be seen as the "equivalent" for Web data of the notion of data models underlying modern DBMSs. Elements may in turn contain other elements, called subelements; thus, an XML data item or document is often characterized by a nested organization. An element may also have associated *attributes*, whose purpose is to provide additional information on the element. XML data can also be interlinked through some special attributes, e.g., IDREFs/URI attributes. Finally, some key features of XML are the notions of *Document Type Definition (DTD)* and XMLSchema, that are used for specifying document structures, very much like a relation schema is used for intensionally describing the structure of tuples in a relation. Note that, unlike relational data, an XML data or document does not necessarily have a DTD or XMLSchema of which it is an instance. A valid[5] XML data or document which is instance of some DTD (XMLSchema) is said to *conform* to the DTD (XMLSchema).

Because XML security is a key requirement, a large number of efforts have been reported dealing with various security standards for XML, such as encryption and signature standards. Access control models and mechanisms have also been widely investigated and several access control systems, specifically tailored to XML, have been developed [18], [49], [52], [71]. A standard access control model, known as XACML, has also been developed [72] which, however, has a limited set of features with respect to those of more advanced data models.

The main requirements toward an access control system for XML derive from the nested structure of XML data and from the main context of use for XML, that is, Web-based environments. The nested structure of XML data calls for a flexible protection object granularity. The system must be able to support a wide spectrum of protection granularity levels, identified on the basis of both the data structure and contents. Examples of protection granularity levels are a single document, a set of documents, an element of a document, and an attribute of a document. Moreover, it must be possible to exploit the intended description provided by a DTD or XMLSchema in the specification of protection objects. For example, it must be possible to specify access control policies at the DTD/XMLSchema level, which apply to all valid documents conforming to that DTD/XMLSchema. To address such requirement, the same techniques proposed for access control in object-based database systems that we discussed in the previous subsection have been adopted. Most of the proposed XML access control models thus provide positive/negative authorizations and explicit/implicit authorizations that can associated with a DTD, a single document, or to specific portions (elements, subelements, attributes) of a document. Authorization propagation, typical of implicit authorization mechanisms, can apply to various types of semantic relationships among protection objects (for instance, element-to-subelement and element-to-attribute/link relationships). With respect to protection objects, however, an important difference between object databases and XML data is that in the former each object is necessarily an instance of some class and, thus, if access control policies are specified at class level, each database object is "covered" by some access control policy. By contrast, in an XML data source, not necessarily each data is an instance of some DTD (or XMLSchema); it may happen, for example, that a source imports XML data for which no DTD (or XMLSchema) is specified. Thus, not every data in an XML source is necessarily covered by some access control policy. If the system uses a closed world access control policy,[6] users may unnecessarily be denied access to some data items. To date, this problem has not been investigated much and the only solutions that have been proposed are those that are part of the Author-X system [14].

The main context of use for XML data, that is, Web-based environments, introduces a number of requirements against both models and architectures of access control systems. Relevant requirements include flexible subject specifications in terms of credentials and profiles, support for dissemination strategies, and distributed and cooperative

5. A valid XML data (document) is a data (document) whose syntax is correct.

6. Under a closed world access control policy, a subject is denied access to a data item if there is no authorization for the subject.

updates. However, whereas most of the proposed systems address in some form the first of these requirements, solutions to the other two requirements are largely unexplored. To date, the only solutions that have been reported are those that are part of the Author-X system [14], which among other features provides flexible credential-based access control policies and different access control techniques for use under two different data dissemination strategies. In particular, it implements an encryption strategy, based on a hierarchical key management scheme [88]; this encryption strategy which requires the generation of a number of encryption keys linear in the number of access control policies is used by Author-X in combination with the push dissemination strategy. The Author-X approach to push-based information dissemination strategy is based on encrypting a given document with different keys [18]; the keys are determined according to the access control policies in such a way as to minimize the number of keys that have to be generated. Such an approach has been recently extended and combined with proxy reencryption schemes for use in content-based publish/subscribe systems [64]. Other notable features of Author-X include support for: distributed cooperative updates through a combination of hash functions, digital signature techniques and digital certificates [21], specification and enforcement of data flow policies, and third-party data publishing, through the use of the well-known Merkle hash trees [13]. An interesting research issue is to investigate how the above techniques could be extended in order to support applications related to content-data networks in peer-to-peer environments.

4 Privacy-Preserving Data Management Techniques

Data represent an important asset. We see an increasing number of organizations that collect data, often concerning individuals, and use them for various purposes, ranging from scientific research, as in the case of medical data, to demographic trend analysis and marketing purposes. Organizations may also give access to the data they own or even release such data to third parties. The number of increased data sets that are thus available poses serious threats against the privacy of individuals and organizations. Because privacy is an important concern, several research efforts have been devoted to address issues related to the development of privacy-preserving data management techniques.

A first important class of techniques deals with privacy-preservation when data are to be released to third parties. In this case, data once are released are no longer under the control of the organizations owning them. Therefore, the organizations that are owners of the data are not able to control the way data are used. The most common approach to address the privacy of released data is to modify the data by removing all information that can directly link data items with individuals; such a process is referred to as *data anonymization* [86]. It is important to note that simply removing identity information, such as names or social-security-numbers, from the released data may not be enough to anonymize the data. There are many examples that show that even when such information is removed from the released data, the remaining data combined with other information sources may link the information to the individuals it refers to. To overcome this problem, approaches based on generalization techniques have been proposed, the most well-known of which is based on the notion of k-anonymity [86].

A second class of techniques deals specifically with privacy-preservation in the context of data mining. Data mining techniques are very effective today. Thus, even though a database is sanitized by removing private information, the use of data mining techniques may allow one to recover the removed information. Several approaches have been proposed, some of which are specialized for specific data mining techniques, such as tools for association rule mining or classification systems, whereas others are independent from the specific data mining technique. In general, all approaches are based on modifying or perturbing the data in some way; for example, techniques specialized for privacy-preserving mining of association rules modify the data so to reduce the confidence of sensitive association rules. A problem common to most of these techniques is the quality of the resulting database; if data undergo too many modifications, they may not be useful any longer. To address these problems, techniques have been developed to estimate the errors introduced by the modifications [73]; such estimates can be used to drive the data modification process. A different technique in this context is based on data sampling [32]. The idea is to release a subset of the data, chosen in such a way that any inference that is made from the data has a low degree of confidence. Finally, in the area of data mining, techniques have been developed, mainly based on commutative encryption techniques, whose goals is to support distributed data mining processes on encrypted data [92]. In particular, the addressed problem deals with situations when the data to be mined is contained at multiple sites, but the sites are unable to release the data. The solutions involve algorithms that share some information to calculate correct results, where the shared information can be shown not to disclose private data.

Finally, some preliminary efforts have been reported dealing with database systems specifically tailored to support privacy policies, such as the policies that can be expressed by using the well-known P3P standard [97]. In particular, Agrawal et al. [2] have recently introduced the concept of Hippocratic databases, incorporating privacy protection in relational database systems. In their paper, Agrawal et al. introduce the fundamental principles underlying Hippocratic databases and then propose a reference architecture. An important feature of such an architecture is that it uses some *privacy metadata* consisting of privacy policies and privacy authorizations stored in privacy-policy tables and privacy-authorization tables, respectively. The privacy policy defines the intended use,

the external-recipients, and retention period for each attribute of a table, while the privacy authorization defines the authorized users. The proposed architecture also adds a special attribute, "purpose," to each table, which encodes the set of purposes users, to whom the data are referred, agree with during the data collection process. The Hippocratic database performs privacy checking during query processing. Every query is submitted to the database with its intended purpose. The system first checks if the user who issued the query is present in the set of authorized users for that purpose in the privacy-authorizations table. Next, the system ensures that the query accesses only the fields that are explicitly listed for the query purpose in the privacy-authorizations table. If the query is allowed to run, the system ensures that only records whose purpose attribute includes the query purpose are visible to the query during the execution. It is important to note that purposes are a very different notion with respect to the notion of role, in that purposes characterize data whereas roles characterize users. Moreover, though purposes may be considered a form of data labels and, thus, similar to labels used in MLS DBMSs, recent approaches [30] to purpose-management have some important differences with respect to label-based approaches developed as part of MLS. These approaches support the association of multiple purposes with the same data item and, thus, are not restricted to a single label, and the specification of negative purposes, specifying that certain data items should not be used for a given set of purposes. In their paper, Agrawal et al. also discuss various technical challenges and problems in designing Hippocratic databases, such as efficiency, disclosure, retention, and safety. To date, many of those problems have yet to be addressed.

5 Challenges—Why Protecting Databases is Even More Difficult Today

Despite the increased focus by research and industry toward improving security of our cyber infrastructures, today the protection of data, entrusted to enterprise information systems, is more challenging than ever. There are several factors underlying this trend.

Data security concerns are evolving. In addition to the traditional requirements of data confidentiality, integrity and availability, new requirements are emerging such as data quality [69], completeness, timeliness, and provenance [35]. In particular, it is important that data be complete, correct, and up-to-date with respect to the external world. The increasing quality of data will make data more valuable. Highly valuable data increases the potential to be gained from unauthorized access and the potential damage that can be done if the data is corrupted. The amount of data is increasingly large: "It is estimated that the amount of information in the world is doubling every 20 months, and the size and number of databases are increasing even faster" [1]. Therefore, protection mechanisms must be able to scale well.

We see increasing "disintermediation"[7] in data accesses. The intermediate information processing steps typically carried out by corporate employees such as typing an order received over the phone are removed. Users who are outside the traditional corporate boundary can have direct and immediate online access to business information which pertain to them. In a traditional environment, any access to sensitive information is through employees. Although employees are not always reliable, at least they are known, their access to sensitive data is limited by their function, and employees violating access policies may be subject to disciplinary action. When activities are moved to the Internet, the environment drastically changes. Today, due also to the offshoring of data management functions and the globalization of business enabled by the Internet, companies may know little or nothing about the users (including, in many cases, employees) accessing their systems and it is more difficult for companies to deter users from accessing information contrary to company policies. Finally, as a result of trends toward ubiquitous computing, data must be available to users anywhere anytime.

Because of these increased risks, the adequate protection of information systems, managing and making available large data volumes, is not an option any longer. Not only will damage to the data affects a company's businesses and operations, it could also have legal consequences on companies especially if, as discussed by Schneier [83], laws were to be promoted enforcing liability of software products and applications. As Schneier argues in his paper, in the very near future insurance companies will move into cyber-insurance and we can certainly expect that "they will start charging different premiums for different security levels." All the above motivations are thus strong drives for the systematic adoption of solutions that are more articulated and comprehensive than the ones available today. Not only must adequate solutions be developed and deployed, but organizations also need to show that they comply with security and privacy requirements. In particular, research efforts need to be devoted on a large number of topics including:

- *Data Quality and Completeness*. Users increasingly rely on information they find on the Web. This is the case for example of medical information. However, users do not, in general, have guarantees that the data is complete and of acceptable quality. We need techniques and organizational solutions to assess and attest the quality of data. Techniques in this respect may include simple mechanisms such quality stamps that are posted on Web sites. Other techniques include providing more effective integrity semantics verification and the use of tools for the assessment of data quality, based on techniques such as record linkage.

7. The term "disintermediation" means removing the middleman. It is today a popular buzzword used to describe many Internet-based businesses that use the WWW to sell products directly to customers rather than going through traditional retail channels.

Application-level recovery techniques are also needed for automatically repairing incorrect data.

- *Intellectual Property Rights (IPR)*. Data in many cases are the results of intellectual activities of individuals and organizations. Questions concerning IPR are thus becoming increasingly relevant. To address some of these concerns, watermarking techniques for relational data have been recently proposed [84], [85] which can be used to detect IPR violations. Research is however needed to assess the robustness of such techniques and to investigate different approaches aimed at preventing IPR violations.
- *Access control and privacy for mobile users*. Users will be increasingly mobile and will have a large variety of devices available to them. Moreover, the deployment of computing power and sensors in every-day environments will make it possible for users to be always connected, sometimes without even being aware of it. In such contexts, several issues are relevant. Users will execute many more activities online; information about user identities, profiles, credentials, and permissions will be more frequently required. Such information will need to be secure and reliable; reliable user identification will be increasingly crucial. It is thus important on one side to develop techniques for efficient storage of security relevant information on small devices; a relevant example in this respect is represented by the notion of portable access rights recently proposed by Bykova and Atallah [29]. On the other side, it is important that access control mechanisms be integrated with standards being developed for identity management [57] as well as with trust negotiation techniques [23]. Because large-sized streams of data are generated in such environments, efficient techniques for access control must be devised and integrated with processing techniques for continuous queries. Finally, the privacy of user location data, acquired from sensors and communication networks, must be assured.
- *Database survivability*. This is an important topic which has been largely unexplored, despite its relevance. Survivability refers to the ability of the database system to continue its functions, may be with reduced capabilities, despite disruptive events, such as information warfare attacks. To date, issues related to database survivability have not been investigated much. Liu [58] has proposed four database architectures for intrusion-tolerant database systems that focus on the containment of malicious transactions. Even though this is an important initial step, much more research needs to be devoted to techniques and methodologies assuring database system survivability.

6 Concluding Remarks

Data security and in particular protection of data from unauthorized accesses remain important goals of any data management system. In this paper, we have outlined research results and practical developments and we have discussed open research issues. The area of database security includes several other relevant topics, such as inference control and statistical database security, for which we refer the readers to [91] and [37], [38], respectively. Though these topics have been investigated several years ago, they are still relevant today especially in the context of privacy-preserving techniques. Other relevant issues that we have not covered here include security for GIS data, an increasingly important area for homeland security, for information-grid architectures and for sensor data as well as privacy and security for Web services and the semantic Web [46]. These applications all have interesting and novel security requirements that are still largely unexplored.

Acknowledgments

The authors would like to thank the anonymous referees and Mahesh Tripunitara of Purdue University for the many invaluable suggestions that lead to a much improved version of this paper. The work of Elisa Bertino is supported in part by the US National Science Foundation under the Project "Collaborative Research: A Comprehensive Policy —Driven Framework For Online Privacy Protection: Integrating IT, Human, Legal, and Economic Perspectives," by an IBM Fellowship, and by the sponsors of CERIAS.

References

[1] R. Agrawal, R. Srikant, and Y. Xu, "Database Technologies for Electronic Commerce," *Proc. Very Large Databases Conf. (VLDB),* 2002.

[2] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Hippocratic Databases," *Proc. 28th Int'l Conf. Very Large Databases (VLDB),* 2002.

[3] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order-Preserving Encryption for Numeric Data," *Proc. 2004 ACM Sigmod Conf.,* 2004.

[4] R. Ahad, J. Davis, S. Gower, P. Lyngbaek, A. Marynowski, and E. Onuegbe, "Supporting Access Control in an Object-Oriented Database Language," *Proc. Int'l Conf. Extending Database Technology (EDBT),* 1992.

[5] G.J. Ahn and R. Sandhu, "Role-Based Authorization Constraints Specification. " *ACM Trans. Information and System Security,* vol. 3, no. 4, pp. 207-226, 2000.

[6] M.M. Astrahan, M.W. Blasgen, D.D. Chamberlin, K.P. Eswaran, J. Gray, P.P. Griffiths, W.F. King III, R.A. Lorie, P.R. McJones, J.W. Mehl, G.R. Putzolu, I.L. Traiger, B.W. Wade, and V. Watson, "System R: A Relational Approach to Database Management," *ACM Trans. Database Systems,* vol. 1, no. 2, pp. 97-137, 1976.

[7] S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," Technical Report No. 99-15, Dept. of Computer Eng., Chalmers Univ. of Technology, Sweden, 2000.

[8] J. Bacon, K. Moody, and W. Yao, "A Model of OASIS Role-Based Access Control and its Support for Active Security," *ACM Trans. Information and System Security,* vol. 5, no. 4, pp. 492-540, 2002.

[9] D.E. Bell and L.J. LaPadula, "Secure Computer Systems: Unified Exposition and Multics Interpretation," Technical Report MTR-2997, The Mitre Corp., Bedford, Mass., 1976.

[10] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati, "An Access Control Model Supporting Periodicity Constraints and Temporal Reasoning," *ACM Trans. Database Systems,* vol. 23, no. 3, pp. 231-285, 1998.

[11] E. Bertino, P. Bonatti, and E. Ferrari, "TRBAC: A Temporal Role-Based Access Control," *ACM Trans. Information and System Security,* vol. 4, no. 3, pp. 191-233, 2001.

[12] E. Bertino, D. Bruschi, S. Franzoni, I. Nai-Fovino, and S. Valtolina, "Threat Modeling for SQL Server," *Proc. Eighth IFIP TC-6 and TC-11 Conf. Comm. and Multimedia Security (CMS 2004),* Sept. 2004.

[13] E. Bertino, B. Carminati, E. Ferrari, B. Thuraisingham, and A. Gupta, "Selective and Authentic Third-Party Distribution of XML Documents," *IEEE Trans. Knowledge and Data Eng.,* vol. 17, no. 1, pp. 4-23, 2004.

[14] E. Bertino, S. Castano, and E. Ferrari, "Securing XML Documents with Author-X," *IEEE Internet Computing,* vol. 5, no. 3, pp. 21-30, 2001.

[15] E. Bertino, B. Catania, and E. Ferrari, "A Nested Transaction Model for Multilevel Secure Database Management Systems," *ACM Trans. Information and System Security,* vol. 4, no. 4, pp. 321-370, 2001.

[16] E. Bertino, B. Catania, E. Ferrari, and P. Perlasca, "A Logical Framework for Reasoning About Access Control Models," *ACM Trans. Information and System Security,* vol. 6, no. 1, pp. 71-127, 2003.

[17] E. Bertino and E. Ferrari, "Administration Policies in a Multipolicy Authorization System," *Proc. 10th Ann. IFIP Working Conf. Database Security,* Aug. 1997.

[18] E. Bertino and E. Ferrari, "Secure and Selective Dissemination of XML Documents," *ACM Trans. Information and System Security,* vol. 5, no. 3, pp. 290-331, 2002.

[19] E. Bertino, E. Ferrari, and V. Atluri, "An Approach for the Specification and Enforcement of Authorization Constraints in Workflow Management Systems," *ACM Trans. Information and System Security,* vol. 2, no. 1, pp. 65-104, 1999.

[20] E. Bertino, J. Fan, E. Ferrari, M.S. Hacid, A. Elmagarmid, and X. Zhou, "A Hierarchical Access Control Model for Video Database Systems," *ACM Trans. Information Systems,* vol. 21, no. 2, pp. 155-191, 2003.

[21] E. Bertino, E. Ferrari, and G. Mella, "An Approach to Cooperative Updates of XML Documents in Distributed Systems," *J. Computer Security,* to appear.

[22] E. Bertino, E. Ferrari, and L. ParasilitiProvenza, "Signature and Access Control Policies," *Proc. 2003 European Symp. Research in Computer Security (ESORICS-03),* Oct. 2003.

[23] E. Bertino, E. Ferrari, and A. Squicciarini, "A Peer-to-Peer Framework for Trust Establishment," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 7, pp. 827-842, 2004.

[24] E. Bertino and L.M. Haas, "Views and Security in Distributed Database Management Systems," *Proc. Int'l Conf. Extending Database Technology,* Mar. 1988.

[25] E. Bertino, D. Leggieri, and E. Terzi, "Securing DBMS: Characterizing and Detecting Query Flood," *Proc. Ninth Information Security Conf. (ISC '04),* Sept. 2004.

[26] E. Bertino, S. Jajodia, and P. Samarati, "Database Security: Research and Practice," *Information Systems,* vol. 20, no. 7, pp. 537-556, 1995.

[27] E. Bertino, S. Jajodia, and P. Samarati, "An Extended Authorization Model," *IEEE Trans. Knowledge and Data Eng.,* vol. 9, no. 1, pp. 85-101, 1997.

[28] R. Bhatti, E. Bertino, A. Ghafoor, and J. Joshi, "XML-Based Specification for Web Services Document Security," *Computer,* vol. 37, no. 4, pp. 41-49, 2004.

[29] M. Bykova and M. Atallah, "Succint Specification of Portable Document Access Policies," *Proc. Ninth ACM Symp. Access Control Models and Technologies (SACMAT 2004),* June 2004.

[30] J.W. Byun, E. Bertino, and N. Lui, "Purpose-Based Access Control for Privacy Protection in Relational Database Systems," CERIAS Technical Report 2004-52, Purdue Univ., 2004.

[31] D.W. Chadwick, A. Otenko, and E. Ball, "Role-Based Access Control With X.509 Attribute Certificates," *IEEE Internet Computing,* vol. 7, no. 2, pp. 62-69, 2003.

[32] C. Clifton, "Using Sample Size to Limit Exposure to Data Mining," *J. Computer Security,* vol. 8, no. 4, Nov. 2000.

[33] COPPA, *Children's Online Privacy Protection Act of 1998,* Oct. 1998, available at www.cdt.org/legislation/105th/privacy/coppa.html.

[34] J. Crampton and G. Loizou, "Administrative Scope: A Foundation for Role-Based Administration," *ACM Trans. Information and System Security,* vol. 6, no. 2, pp. 201-231, 2003.

[35] Y. Cui and J. Widom, "Lineage Tracing for General Data Warehouse Transformations," *VLDB J.,* vol. 12, no. 1, pp. 41-58, 2003.

[36] D.E. Denning, "A Lattice Model of Secure Information Flow," *Comm. ACM,* vol. 19, no. 5, pp. 236-243, 1976.

[37] D.E. Denning, "Secure Statistical Databases with Random Sample Queries," *ACM Trans. Database Systems,* vol. 5, no. 3, pp. 291-315, 1980.

[38] D.E. Denning and J. Schlörer, "A Fast Procedure for Finding a Tracker in a Statistical Database," *ACM Trans. Database Systems,* vol. 5, no. 1, pp. 88-102, 1980.

[39] US Dept. of Defense, *Trusted Computer System Evaluation Criteria,* DOD 5200. 28-STD, Dept. of Defense, Washington, D.C., 1975.

[40] Y. Diao, S. Rivzi, and M. Franklin, "Toward an Internet-Scale XML Dissemination Service," *Proc. Very Large Databases Conf.,* 2004.

[41] A. Eisenberg and J. Melton, "SQL:1999, Formerly Known as SQL 3," *SIGMOD Record,* 1999.

[42] R. Fagin, "On an Authorization Mechanism," *ACM Trans. Database Systems,* vol. 3, no. 3, pp. 310-319, 1978.

[43] Federal Trade Commission, "FTC Announces Settlement with Bankrupt Website, Toysmart.com, Regarding Alleged Privacy Policy Violations," July 2000, available at www.ftc.gov/opa/2000/07/toysmart2.htm.

[44] E.B. Fernandez, R.C. Summers, and T. Lang, "Definition and Evaluation of Access Rules in Data Management Systems," *Proc. Very Large Databases Conf.,* 1975.

[45] E.B. Fernandez, R.C. Summers, and C. Wood, *Database Security and Integrity.* Addison-Wesley, Feb. 1981.

[46] E. Ferrari and B.M. Thuraisingham, "Security and Privacy for Web Databases and Services," *Advances in Database Technology—EDBT 2004, Proc. Ninth Int'l Conf. Extending Database Technology,* Mar. 2004.

[47] D. Ferraiolo, R. Sandhu, S. Gavrila, R. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-based Access Control," *ACM Trans. Information and System Security,* vol. 4, no. 3, pp. 224-274, 2001.

[48] D. Ferraiolo, R. Chandramouli, and R. Kuhn, *Role-Based Access Control.* Artech House, Apr. 2003.

[49] A. Gabillon and E. Bruno, "Regulating Access to XML Documents," *Proc. 15th Ann. IFIP WG 11.3 Working Conf. Database Security,* July 2001.

[50] J. Gray and A. Reuter, *Transaction Processing: Concepts and Techniques.* Morgan Kaufmann, 1993.

[51] P.G. Griffiths and B. Wade, "An Authorization Mechanism for a Relational Database," *ACM Trans. Database Systems,* vol. 1, no. 3, pp. 242-255, 1976.

[52] H. He and R.K. Wong, "A Role-Based Access Control for XML Repositories," *Proc. First Int'l Conf. Web Information Systems Eng. (WISE '00),* 2000.

[53] HIPAA, *Health Insurance Portability and Accountability Act of 1996,* available at http://www.hep-c-alert.org/links/hipaa.html, 1996.

[54] B. Iyer, S. Mehrotra, E. Mykletun, G. Tsudik, and Y. Wu, "A Framework for Efficient Storage Security in RDBMS," *Proc. Seventh Int'l Conf. Extending Database Technology (EDBT 2004),* Mar. 2004.

[55] S. Jajodia, R. Sandhu, and B. Blaustein, "Solutions to the Polyinstantiation Problem," *Information Security: An Integrated Collection of Essays,* vol. 1, M.A. Abrams et al. eds., IEEE CS Press, pp. 493-529, 1994.

[56] N. Li and M. Tripunitara, "Security Analysis in Role-Based Access Control," *Proc. Ninth ACM Symp. Access Control Models and Technologies (SACMAT 2004),* June 2004.

[57] Liberty Alliance Project (www.projectliberty.org), 2001.

[58] P. Liu, "Architectures for Intrusion Tolerant Database Systems," *Proc. 18th Ann. Computer Security Applications Conf. (ACSAC 2002),* Dec. 2002.

[59] D.E. Denning, T.F. Lunt, R.R. Schell, W.R. Shockley, and M. Heckman, "The Sea View Security Model," *IEEE Trans. Software Eng.,* vol. 16, no. 6, pp. 593-607, 1990.

[60] G. Karjoth, "Access Control with IBM Tivoli Access Manager," *ACM Trans. Information and System Security,* vol. 6, no. 2, pp. 232-257, 2003.

[61] G. Karjoth, M. Schunter, E. VanHerreweghen, "Translating Privacy Practices into Privacy Promises—How to Promise What You Can Keep," *Proc. IEEE POLICY Workshop,* 2003.

[62] C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World,* second ed. Prentice-Hall, 2002.

[63] A. Kern, M. Kuhlmann, R. Kuropka, and A. Ruthert, "A Meta Model for Authorisations in Application Security Systems and their Integration into RBAC Administration," *Proc. Ninth ACM Symp. Access Control Models and Technologies (SACMAT 2004),* June 2004.

[64] H. Khurana, "Scalable Security and Accounting Services for Content-Based Publish/Subscribe Systems," *Proc. Symp. Applied Computing (SAC05),* Mar. 2005.

[65] M. Koch, L. Mancini, and F. Parisi-Presicce, "Administrative Scope in the Graph-based Framework," *Proc. Ninth ACM Symp. Access Control Models and Technologies (SACMAT 2004),* June 2004.

[66] N. Kodali, C. Farkas, and D. Wijesekera, "An Authorization Model for Digital Libraries," *Int'l J. Digital Libraries,* vol. 4, no. 3, pp. 156-170, 2004.

[67] R. Kuhn, "Mutual Exclusion of Roles as a Means of Implementing Separation of Duty in Role-Based Access Control Systems," *Proc. Second ACM Workshop Role-Based Access Control,* June 1997.

[68] J.B. Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A Generalized Temporal Role Based Access Control Model," *IEEE Trans. Knowledge and Data Eng.,* vol. 17, no. 1, pp. 4-23, 2005.

[69] P. Missier, G. Lalk, V.S. Verykios, F. Grillo, T. Lorusso, and P. Angeletti, "Improving Data Quality in Practice: A Case Study in the Italian Public Administration," *Distributed and Parallel Databases,* vol. 13, no. 2, pp. 135-160, 2003.

[70] J.E. Moss, *Nested Transactions: An Approach to Reliable Distributed Computing.* MIT Press, 1985.

[71] M. Murata, A. Tozawa, M. Kudo, and S. Hada, "XML Access Control Using Static Analysis," *Proc. 10th ACM Conf. Computer and Comm. Security,* Nov. 2003.

[72] OASIS Consortium, eXtensible Access Control Markup Language (XACML) Committee Specification, Version 1.1, available at: http://www.oasis-open.org/committees/xacml/, 2000.

[73] S.R.M. Oliveira and O.R. Zaiane, "Privacy Preserving Frequent Itemset Mining," *Proc. IEEE ICDM Workshop Privacy, Security and Data Mining,* 2002.

[74] Oracle, The Virtual Private Database in Oracle9iR2, available at http://otn.oracle.com/deploy/security/oracle9iR2/pdf/VPD9ir2twp.pdf, 2000.

[75] F. Rabitti, E. Bertino, W. Kim, and D. Woelk, "A Model of Authorization for Next-Generation Database Systems," *ACM Trans. Database Systems,* vol. 16, no. 1, pp. 88-131, 1991.

[76] C. Ramaswamy and R. Sandhu, "Role-Based Access Control Features in Commercial Database Management Systems," *Proc. 21st Nat'l Information Systems Security Conf.,* pp. 503-511, Oct. 1998.

[77] J. Richardson, P. Schwarz, and L.F. Cabrera, "CACL: Efficient Fine-Grained Protection for Objects," *Proc. Int'l Conf. Object-Oriented Programming Systems, Languages, and Applications (OOPSLA),* 1992.

[78] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy, "Extending Query Rewriting Techniques for Fine-Grained Access Control," *Proc. ACM Sigmod Conf.,* June 2004.

[79] R. Sandhu, "Lattice-Based Access Control Models. " *Computer,* vol. 26, no. 11, pp. 9-19, 1993.

[80] R. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman, "Role-Based Access Control Models," *Computer,* vol. 29, no. 2, pp. 38-47, 1996.

[81] R. Sandhu and F. Chen, "The Multilevel Relational Data Model," *ACM Trans. Information and System Security,* vol. 1, no. 1, pp. 93-132, 1998.

[82] O. SamySayadjari, "Multilevel Security: Reprise," *IEEE Security and Privacy,* vol. 3, no. 5, 2004.

[83] B. Schneier, "Hacking the Business Climate for Network Security," *Computer,* vol. 37, no. 4, pp. 87-89, 2004.

[84] R. Sion, M. Atallah, and S. Prabhakar, "Resilient Rights Proofs for Sensor Streams," *Proc. Conf. Very Large Databases,* Sept. 2004.

[85] R. Sion, M. Atallah, and S. Prabhakar, "Protecting Rights Proofs for Relational Data using Watermarking," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 12, pp. 1509-1525, 2004.

[86] L. Sweeney, "Achieving k-Anonymity Privacy Protection Using Generalization and Suppression," *Int'l J. Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, no. 5, 2002.

[87] R. Thomas and R. Sandhu, "Task-Based Authorization Controls (TBAC) Models for Active and Enterprise-Oriented Authorization Management," *Database Security XI: Status and Prospects,* T.Y. Lin and S. Qian, eds., pp. 262-275, 1998.

[88] W.G. Tzeng, "A Time-Bound Cryptographic Key Assignment Scheme for Access Control in a Hierarchy," *IEEE Trans. Knowledge and Data Eng.,* vol. 14, no. 1, pp. 182-188, 2002.

[89] B. Thuraisingham, "Mandatory Security in Object-Oriented Database Systems," *Proc. Int'l Conf. Object-Oriented Programming Systems, Languages, and Applications (OOPSLA),* 1989.

[90] B. Thuraisingham, *Database and Applications Security: Integrating Databases and Applications Security.* CRC Press, Dec. 2004.

[91] B.M. Thuraisingham, W. Ford, M. Collins, and J. O'Keeffe, "Design and Implementation of a Database Inference Controller," *Data Knowledge Eng.,* vol. 11, no. 3, pp. 271-285, 1993.

[92] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* July 2002.

[93] J. Widom and S. Ceri, *Active Database Systems: Triggers and Rules For Advanced Database Processing.* Morgan Kaufmann, 1996.

[94] J. Wang and S. Osborn, "A Role-Based Approach to Access Control for XML Databases," *Proc. Ninth ACM Symp. Access Control Models and Technologies (SACMAT 2004),* June 2004.

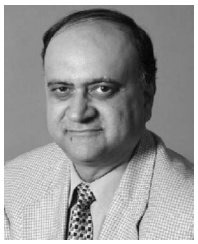
[95] C. Wood and E.B. Fernandez, "Decentralized Authorization in a Database System," *Proc. Conf. Very Large Databases,* 1979.

[96] World Wide Web Consortium, Extensible Markup Language (XML), 1.0, 1998, available at: http://www.w3.org/TR/REC-xml.

[97] World Wide Web Consortium, Platform for Privacy Preferences (P3P), available at www.w3.org/P3P, 1994.

[98] T.W. Yan and H. Garcia-Molina, "The SIFT Information Dissemination System," *ACM Trans. Database Systems,* vol. 24, no. 4, pp. 529-565, 1999.

Elisa Bertino is a professor of computer science and of electrical and computer engineering at Purdue University and serves as the research director of CERIAS. She is also a faculty member in the Department of Computer Science and Communication of the University of Milan where she is the director of the DB&SEC laboratory. She has been a visiting researcher at the IBM Research Laboratory (now Almaden) in San Jose, at the Microelectronics and Computer Technology Corporation, at Telcordia Technologies. Her main research interests include security, privacy, database systems, object-oriented technology, and multimedia systems. In those areas, she has published more than 250 papers in all major refereed journals and in international conferences and symposia proceedings. Her research has been funded by several entities and organizations both in the USA and Europe, including the US National Science Foundation, the European Union under the Fifth and Sixth Research Program Framework, IBM, Telcordia, Microsoft, and the Italian Telecom. She is a coauthor of the books *Object-Oriented Database Systems—Concepts and Architectures* (Addison-Wesley, 1993), *Indexing Techniques for Advanced Database Systems* (Kluwer Academic, 1997), and *Intelligent Database Systems* (Addison-Wesley, 2001). She is a coeditor-in-chief of the *Very Large Database Systems (VLDB) Journal* and a member of the advisory board of the *IEEE Transactions on Knowledge and Data Engineering*. She serves also on the editorial boards of several scientific journals, incuding *IEEE Internet Computing*, *ACM Transactions on Information and System Security*, *IEEE Transactions on Secure and Dependable Computing*, the *Journal of Computer Security*, *Data & Knowledge Engineering*, the *International Journal of Cooperative Information Systems*, and *Science of Computer Programming*. She has served as program committee members of several international conferences, such as ACM SIGMOD, VLDB, ACM OOPSLA, as program cochair of the 1998 IEEE International Conference on Data Engineering (ICDE), as program chair of 2000 European Conference on Object-Oriented Programming (ECOOP 2000), as program chair of the Seventh ACM Symposium on Access Control Models and Technologies (SACMAT 2002), and as program chair of the 2004 Extending Database Technology (EDBT 2004) Conference. She is a fellow of the IEEE and a fellow of the ACM. She received the 2002 IEEE Computer Society Technical Achievement Award for "For outstanding contributions to database systems and database security and advanced data management systems."



Ravi Sandhu received the BTech and MTech degrees in electrical engineering from the Indian Institutes of Technology at Bombay and Delhi, respectively, and the MS and PhD degrees in computer science from Rutgers University. He is a professor of information security and assurance and the director of the Laboratory for Information Security Technology (www.list.gmu.edu) at George Mason University, and chief scientist and cofounder of TriCipher, Inc. He is a leading authority on access control, authorization, and authentication models and protocols, and is especially known for his seminal and highly influential work in role-based access control. He is a fellow of the ACM and a fellow of the IEEE. He has published more than 150 technical papers on computer security in refereed journals, conference proceedings and books. He founded the *ACM Transactions on Information and Systems Security (TISSEC)* in 1997 and served as editor-in-chief until 2004. He served as the chairman of ACM's Special Interest Group on Security Audit and Control (SIGSAC) from 1995 to 2003, and founded and led the ACM Conference on Computer and Communications Security (CCS) and the ACM Symposium on Access Control Models and Technologies (SACMAT) to high reputation and prestige. Most recently, he founded the IEEE Workshop on Pervasive Computing Security (PERSEC) in 2004. His research has been sponsored by numerous public and private organizations currently including Intel, the US National Science Foundation, and ARDA. He has provided high-level security consulting services to several private and government organizations. Dr. Sandhu has also served as the principal designer and security architect of TriCipher's identity management appliance which earned the coveted FIPS 140 level 2 rating from NIST.