Database security essay

– 1 – Database Security *) GUNTHER PERNUL Institut fur Angewandte Informatik und Informationssysteme Abteilung fur Information Engineering Universitat Wien Vienna, Austria 1.

Introduction 1. 1 The Relational Data Model Revisited 1. 2 The Vocabulary of Security and Major DB Security Threats 2. Database Security Models 2. 1 Discretionary Security Models 2. 2 Mandatory Security Models 2.

3 Adapted Mandatory Access Control Model 2. 4 Personal Knowledge Approach 2. 5 Clark and Wilson Model 2. 6 A Final Note on Database Security Models . Multilevel Secure Prototypes and Systems 3.

1 SeaView 3. 2 Lock Data Views 3. 3 ASD_Views 4. Conceptual Data Model for Multilevel Security 4.

1 Concepts of Security Semantics 4. 2 Classification Constraints 4. 3 Consistency and Conflict Management 4. 4 Modeling the Example Application 5. Standardization and Evaluation Efforts 6. Future Directions in Database Security Research 7.

Conclusions References 1. Introduction Information stored in databases is often considered as a valuable and important corporate resource. Many organizations have become so dependent on the proper functioning of their systems that a disruption of service or a leakage of stored information may cause outcomes ranging from inconvenience to catastrophe. Corporate data may relate to financial records, others may be essential for the successful operation of an organization, may represent trade *) Advances in Computers, Vol. 38.

M. C. Yovits (Ed.), Academic Press, 1994, pp. 1 – 74. – 2 – secrets, or may describe information about persons whose privacy must be protected.

Thus, the general concept of database security is very broad and entails such things as moral and ethical issues imposed by public and society, legal issues where control is legislated over the collection and disclosure of stored information, or more technical issues such as how to protect the stored information from loss or unauthorized access, destruction, use, modification, or disclosure. More generally speaking, database security is concerned with ensuring the secrecy, integrity, and availability of data stored in a database. To define the terms, secrecy denotes the protection of information from unauthorized disclosure either by direct retrieval or by indirect logical inference. In addition, secrecy must deal with the possibility that information may also be disclosed by legitimated users acting as an ‘ information channel’ by passing secret information to unauthorized users. This may be done intentionally or without knowledge of the authorized user. Integrity requires data to be protected from malicious or accidental modification, including the insertion of false data, the contamination of data, and the destruction of data.

Integrity constraints are rules that define the correct states of a database and thus can protect the correctness of the database during operation. Availability is the characteristic that ensures data being available to authorized users when they need them. Availability includes the ‘ denial of service’ of a system, i. e. a system is not functioning in accordance with its intended purpose. Availability is closely related to integrity because ‘ denial of service’ may be caused by unauthorized destruction, modification, or delay of service as well.

Database security cannot be seen as an isolated problem because it is effected by other components of a computerized system as well. The security requirements of a system are specified by means of a security policy which is then enforced by various security mechanisms. For databases, requirements on the security can be classified into the following categories: · Identification, Authentication Usually before getting access to a database each user has to identify himself to the computer system. Authentication is the way to verify the identity of a user at log-on time. Most common authentication methods are passwords but more advanced techniques like badge readers, biometric recognition techniques, or signature analysis devices are also available. · Authorization, Access Controls Authorization is the specification of a set of rules that specify who has which type of access to what information.

Authorization policies therefore govern the disclosure and modification of information. Access controls are – 3 – procedures that are designed to control authorizations. They are responsible to limit access to stored data to authorized users only. · Integrity, Consistency An integrity policy states a set of rules (i. e.

semantic integrity constraints) that define the correct states of the database during database operation and therefore can protect against malicious or accidental modification of information. Closely related issues to integrity and consistency are concurrency control and recovery. Concurrency control policies protect the integrity of the database in the presence of concurrent transactions. If these transactions do not terminate normally due to system crashes or security violations recovery techniques are used to reconstruct correct or valid database states. · Auditing The requirement to keep records of all security relevant actions issued by a user is called auditing.

Resulting audit records are the basis for further reviews and examinations in order to test the adequacy of system controls and to recommend any changes in the security policy. In this Chapter such a broad perspective of database security is not taken. Instead, main focus is directed towards aspects related to authorization and access controls. This is legitimate because identification, authentication, and auditing1 normally fall within the scope of the underlying operating system and integrity and consistency policies are subject to the closely related topic of ‘ semantic data modeling’ or are dependent on the physical design of the DBMS software (namely, the transaction and recovery manager). Because most of the research in database security has concentrated on the relational data model, the discussion in this Chapter mostly concerns the framework of relational databases. However, the results described may generally be applicable to other database models as well.

For an overall discussion on basic database security concepts consult the surveys by Jajodia and Sandhu (1990a), Lunt and Fernandez (1990), or Denning (1988). For references to further readings consult the annotated bibliography by Pernul and Luef (1992). The outline of this Chapter is as follows: In the remainder of the opening Section we shortly review the relational data model, we introduce a simple example that will be used throughout the Chapter, we present the basic terminology used in computer security, and we describe the most successful methods that might be used to penetrate a database. Because of the diversity of application domains for databases different security models and techniques 1.

However, audit records are often stored and examined by using the DBMS software. – 4 – have been proposed so far. In Section 2 we review, evaluate, and compare the most prominent representatives among them. Section 3 contains an investigation of secure (trusted) database management systems (DBMSs). These are special purpose systems that support a level-based security policy and were designed and implemented with main focus on the enforcement of high security requirements. Section 4 focuses on one of the major problems level-based security related database research has to deal with.

In this Section we address the problem of how to classify the data stored in the database with security classifications reflecting the security requirements of the application domain properly. What is necessary to counter this problem is to have a clear understanding of all the security semantics of the database application and a resulting clever database design. A semantic data/security model is proposed to arrive at a conceptualization and a clear understanding of the security semantics of the database application. Database security (and computer security in general) is subject to many national and international standardization efforts.

The efforts have the goal to develop metrics to evaluate the degree of trust that can be placed in computer products used for the processing of sensitive information. In Section 5 we will briefly review these proposals. In Section 6 we will point out research challenges in database security and we will give our opinion of the direction in which we expect the entire field to move within the next few years. Finally, Section 7 will conclude this Chapter.

1. 1 The Relational Data Model RevisitedThe relational data model was invented by Codd (1970) and is described in most database textbooks. A relational database supports the relational data model and must have three basic principles: a set of relations, integrity rules, and a set of relational operators. Each relation consists of a state-invariant relation schema RS(A1, …

, An), where each Ai is called attribute and defined over a domain dom(Ai). A relation R is a state-dependent instance of RS and consists of a set of distinct tuples of the form (a1,… , an), where each element ai must satisfy dom(Ai) (i.

e. aiIdom(Ai)). Integrity constraints restrict the set of theoretically possible tuples (i. e.

dom(A1) ? dom(A2) ? … ? dom(An)) to the set of practically meaningful. Let X and Y denote sets of one or more of the attributes Ai in a relation schema.

We say Y is functional dependent on X, written X®Y, if and only if it is not possible to have two tuples with the same value for X but different values for Y. Functional dependencies represent the basis for most integrity constraints in the relation model of data. As not all possible relations are meaningful in an application, only those that satisfy certain integrity constraints are considered. 5 – From the large set of proposed integrity constraints two are of major relevance for security: the key property and the referential integrity property.

The key property states that each tuple must be uniquely identified by a key and a key attribute must not have the null-value. As a consequence each event of reality may be represented in the database only once. Referential integrity states that tuples referenced in one relation must exist in others and is expressed by means of foreign keys. These two rules are application independent and must be valid in each relational database. In addition many application dependent semantic constraints may exist in different databases. Virtual view relations (or shortly views) are distinguished from base relations.

While the former are the result of relational operations and exists only virtually, the latter are actually present in the database and hold the stored data. Relational operations consist of the set operations, a select operation for selecting tuples from relations that satisfy a certain predicate, a project operation for projecting a relation on a subset of its attributes and a join operation for combining attributes and tuples from different relations. The relational data model was first implemented as System R by IBM and as INGRES at U. C. Berkeley.

These two projects have mainly started and also considerably advanced the field of database security research. Both systems are the basis of most commercially available products. A few words on designing a database are in order. The design of a relational database is a complicated and difficult task and involves several phases and activities. Before the final relation schemas can be determined a careful requirements analysis and a conceptualization of the database is necessary.

Usually this is done by using a conceptual data model which must be powerful enough to allow the modeling of all application relevant knowledge. The conceptual model is used as an intermediate representation of the database and finally transferred into corresponding relation schemas. It is very important to use a conceptual data model at this step because only such a high level data model allows to achieve a database that properly represents all of the application dependent data semantics. De facto standard for conceptual design is the Entity Relationship Approach (ER) (Chen, 1976) or one of its variants. In its graphical representation and in its simplest form the ER regards the world as consisting of a set of entity types (boxes), attributes (connected to boxes) and relationship types (diamonds).

Relationship types are defined between entity types and are either of degree <1: 1>, <1: n>, or . The degree describes the maximum number of participating entities. Following is a short example of a relational database. This example will be used throughout the Chapter.

It is very simple but sufficient to discuss many – 6 – ecurity relevant questions and to show the complexity of the field. Figure 1 contains the conceptualization of the database in form of an ER diagram and the corresponding relation schemas (key attributes are underlined, foreign keys are in italics). The database represents the fact that projects within an enterprise are carried out by employees. In this simple example we have to deal with the following three security objects: First, Employee represents a set of employees each of which is uniquely described by a characteristic SSN (i.

e. the social security number). Of further interest are the Name, the Department the employee is working for, and the Salary of the employee. Second, Project is a set of projects carried out by the enterprise.

Each project has an identifying Title, a Subject, and a Client. Finally, security object Assignment contains the assignments of employees to projects. Each assignment is characterized by the Date of the assignment and the Function the employee has to perform during the participation in the project. A single employee can be assigned to more than one project and a project may be carried out by more than one employee.

1. The Vocabulary of Security and Major DB Security Threats Before presenting the details of database security research it is necessary to define the terminology used and the potential threats to database security. As already has been pointed out, security requirements are stated by means of a security policy which consists of a set of laws, rules and practices that regulate how an organization manages, protects, and distributes sensitive information. In general, a security policy is stated in terms of a set of security objects and a set of security subjects. A security object is a passive entity that contains or receives information. This might be a structured concept like a whole database, Employee Project Assignment N M Date Function SSN Title Title Subject Client SSN Name Dep Salary Employee (SSN, Name, Dep, Salary) Project (Title, Subject, Client) Assignment (Title, SSN, Date, Function) FIG.

1. Representations of the Example DB – 7 – a relation, a view, a tuple, an attribute, an attribute value, or even a fact of reality which is represented in the database. A security object might also be unstructured like a physical memory segment, a byte, a bit, or even a physical device like a printer or a processor. Please note, the term object is used differently in other computer science disciplines. Within the framework presented here, security objects are the target of protection.

A security subject is an active entity, often in the form of a person (user) or process operating on behalf of a user. Security subjects are responsible for a change of a database state and cause information to flow within different objects and subjects. Most sources of threats to database security come from outside the computing system. If most emphasis is given to authorization, the users and processes operating on behalf of the users must be subject to security control.

An active database process may be operating on behalf of an authorized user who has legitimate access or may be active on behalf of a person who succeeded in penetrating the system. In addition, an authorized database user may act as an ‘ information channel’ by passing restricted information to unauthorized users. This may be intentionally or without knowledge of the authorized user. Some of the most successful database penetration methods are: · Misuses of authority Improper acquisition of resources, theft of programs or storage media, modification or destruction of data. Logical Inference and Aggregation Both deal with users authorized to use the database. Logical inference arises whenever sensitive information can be inferred from combining less sensitive data.

This may also involve certain knowledge from outside the database system. Tightly related to logical inference is the aggregation problem, wherein individual data items are not sensitive but a large enough collection of individual values taken together is considered sensitive. · Masquerade A penatrator may gain unauthorized access by masquerading as a different person. · Bypassing ControlsThis might be password attacks and exploitation of system trapdoors that avoid intended access control mechanisms.

Trapdoors are security flaws that were built in the source code of a program by the original programmer. · Browsing A penetrator circumvents the protection and searches directory or – 8 – dictionary information, trying to locate privileged information. Unless strict need-to-know access controls are implemented the browsing problem is a major flaw of database security. · Trojan Horses A Trojan horse is hidden software that tricks a legitimate user without his knowledge to perform certain actions he is not aware of.

For example, a Trojan Horse may be hidden into a sort routine and be designed to release certain data to unauthorized users. Whenever a user activates the sort routine, for example for sorting the result of a database query, the Trojan horse will act with the users identity and thus will have all privileges of the user. · Covert Channels Usually information stored in a database is retrieved by means of legitimate information channels. In contrast to legitimate channels covert channels are paths that are not normally intended for information transfer. Such hidden paths may either be storage channels like shared memory or temporary files that could be used for communication purposes or timing channels like a degradation of overall system performance. · Hardware, Media Attacks Physical attacks on equipment and storage media.

The attack scenario described above is not restricted to occur in databases only. For example, the German Chaos Computer Club succeeded in attacking a NASA system masqueraded, by bypassing access controls (by means of an operating system flaw) and Trojan horses to capture passwords. As reported by Stoll (1988) some of these techniques were also used by the Wily Hacker. The Internet worm in 1988 exploited trapdoors in electronic mail handling systems and infected more than 5000 machines connected to the Internet network (Rochlis and Eichin, 1989). Thompson (1984), in his Turing Award Lecture, demonstrated a Trojan horse placed in the executable form of a compiler that permitted the insertion of a trapdoor in each program compiled with the compiler.

It is generally agreed that the number of the known cases of computer abuse is significantly smaller than the cases actually happened because in this topic a large number of dark figures exist. 2. Database Security Models – 9 – Because of the diversity of the application domains for databases different security models and techniques have been proposed to counter the various threats against the security. In this Section we will discuss the most prominent among them.

In a nutshell, Discretionary Security specifies the rules under which subjects can, at their discretion, create and delete objects, and grant and revoke authorizations for accessing objects to others. In addition to controlling the access Mandatory Security regulates the flow of information between objects and subjects. Mandatory security controls are very effective but suffer from several drawbacks. One attempt to overcome certain limitations of mandatory protection systems is the Adapted Mandatory Access Control (AMAC) model, a security technique that focuses on the design aspect of secure databases. The Personal Knowledge Approach is concentrating on enforcing the basic law of many countries for the informational selfdetermination of humans and the Clark and Wilson Model tries to represent common commercial business practice in a computerized security model.

First attempts to compare some of these techniques have been made by Biskup (1990) and Pernul and Tjoa (1992). Landwehr (1981) is a very good survey of formal policies for computer security in general and Millen (1989) focuses on various aspects of mandatory computer security. 2. 1 Discretionary Security Models Discretionary security models are fundamental to operating systems and DBMSs and have now been studied for a long time. From 1970 through 1975, there was a good deal of interest in the theoretical aspects of these models.

Then most of the relational database security research has turned to other security techniques. However, the appearance of more advanced data models has renewed interest in discretionary policies. 2. 1. 1 Discretionary Access Controls Discretionary access controls (DAC) are based on the concepts of a set of security objects O, a set of security subjects S, a set of access privileges T defining what kind of access a subject has to a certain object, and in order to represent content-based access rules a set of predicates P.

Applied to relational databases O is a finite set of values {o1,… , on} representing relation schemas, S is a finite set of potential subjects {s1,…

sm} representing users, groups of them, or transactions operating on behalf of users. Access types (privileges) are the set of database operations such as select, insert, delete, update, execute, grant, or – 10 – revoke and predicate pIP defines the access window of subject sIS on object oIO. The tuple is called access rule and a function f is defined to determine if an authorization f(o, s, t, p) is valid or not: : O ? S ? T ? P ® {True, False}. For any , if f(o, s, t, p) evaluates into True, subject s has authorization t to access object o within the range defined by predicate p. An important property of discretionary security models is the support of the principle of delegation of rights where a right is the (o, t, p)-portion of the access rule. A subject si who holds the right (o, t, p) may be allowed to delegate that right to another subject sj (i? j).

Most systems supporting DAC store access rules in an access control matrix. In its simplest form the rows of the matrix represent subjects, the columns represent the objects and the intersection of a row and a column contains the access type that subject has authorization for with respect to the object. The access matrix model as a basis for discretionary access controls was formulated by Lampson (1971) and subsequently refined by Graham and Denning (1972), and by Harrison et al. (1976). A more detailed discussion on discretionary controls in databases may be found in the book by Fernandez et al.

(1981). Discretionary security is enforced in most commercial DBMS products and is based on the concept of database views. Instead of authorizing a user to the base relations of a system the information of the access control matrix is used to restrict the user to a particular subset of the data available. Two main system architectures for view-based protection can be identified: query modification and view relations. Query modification is implemented in Ingres-style DBMSs (Stonebraker and Rubinstein 1976) and consists of appending additional security relevant qualifiers to a user supplied query.

View relations are unmaterialized queries which are based on physical base relations. Instead of authorizing the users to base relations they have access to the virtual view relations only. By means of qualifiers in the view definition security restrictions can be implemented. View relations are the underlying protection mechanism of System R-based DBMSs (Griffiths and Wade, 1976).

2. 1. 2 DAC-based Structural Limitations Although very common discretionary models suffer from major drawbacks when pplied to databases with security critical content. In particular we see the following limitations: · Enforcement of the security policy – 11 – DAC is based on the concept of ownership of information.

In contrast to enterprise models, where the whole enterprise is the ‘ owner’ of information and responsible for granting access to stored data, DAC systems assign the ownership of information to the creator of the data items in the database and allow the creator subject to grant access to other users. This has the disadvantage that the burden of enforcing the security requirements of the enterprise is in the responsibility of the users themselves and cannot be controlled by the enterprise without involving high costs. · Cascading authorization If two or more subjects have the privilege of granting or revoking certain access rules to other subjects this may lead to cascading revocation chains. As an example consider subjects s1, s2, s3, and access rule (s1, o, t, p).

Subject s2 receives the privilege (o, t, p) from s1 and grants this access rule to s3. Later, s1 grants (o, t, p) again to s3 but s2 revokes (o, t, p) from s3 because of some reason. The effect of these operations is that s3 still has the authorization (from s1) to access object o by satisfying predicate p and using privilege t even if subject s2 has revoked it. This has the consequence that subject s2 is not aware of the fact that authorization (s3, o, t, p) is still in effect. · Trojan Horse attacks In systems supporting DAC the identity of the subjects is crucial, and if actions can be performed using another subject’s identity, then DAC can be subverted. A Trojan Horse can be used to grant a certain right (o, t, p) of subject si on to sj (i? j) without the knowledge of subject si.

Any program which runs on behalf of a subject acts with the identity of the subject and therefore has all of the DAC access rights of the subject’s processes. If a program contains a Trojan Horse with the functionality of granting access rules on to other users this cannot be restricted by discretionary access control methods. · Update problems View-based protection results in unmaterialized queries which have no explicit physical representation in the database. This has the advantage of being very flexible to support the subjects with different views and to automatically filter out data a subject is not authorized to access but has the disadvantage that not all data is updateable through certain views. This is due to integrity reasons that might be violated in data not contained in the view by updating data from the view.

2. 2 Mandatory Security Models – 12 – Mandatory policies address a higher level of threat than discretionary policies because in addition to controlling the access to data they control the flow of data as well. Moreover, mandatory security techniques overcome the structural limitations of DAC-based protection as described above. 2.

2. 1 Mandatory Access Controls While discretionary models are concerned with defining, modeling, and enforcing access to information mandatory security models are in addition concerned with the flow of information within a system. Mandatory security requires that security objects and subjects are assigned to certain security levels represented by a label. The label for an object o is called its classification (class(o)) and a label for a subject s is called its clearance (clear(s)). The classification represents the sensitivity of the labeled data while the clearance of a subject its trustworthiness to not disclose sensitive information to others.

A security label consists of two components: a level from a hierarchical list of sensitivity levels or access classes (for example: top_secret > secret > confidential > unclassified) and a member of a non hierarchical set of categories, representing classes of object types of the universe of discourse. Clearance and classification levels are totally ordered resulting security labels are only partially ordered – thus, the set of classifications forms a lattice. In this lattice security class c1 is comparable to and dominates (?) c2 if the sensitivity level of c1 is greater than or equal to that of c2 and the categories in c1 contain those in c2. Mandatory security grew out of the military environment where it is practice to label information. However, this custom is also common in many companies and organizations where labels termed like ‘ confidential’ or ‘ company confidential’ are used.

Mandatory access control (MAC) requirements are often stated based on Bell and LaPadula (1976) and formalized by two rules. The first (simple property) protects the information of the database from unauthorized disclosure, and the second (*-property) protects data from contamination or unauthorized modification by restricting the information flow from high to low. (1) Subject s is allowed to read data item d if clear(s) ? class(d). (2) Subject s is allowed to write data item d if clear(s) ? class(d).

Few final sentences on MAC policies are in order. In many discussions confusion has arisen about the fact that in mandatory systems it is not only sufficient to have strong controls over who can read which data. Why is it necessary to include strong controls over who can write which data in systems with high security requirements? The reason is that a system with high security – 13 – needs must protect itself against attacks from unauthorized as well as from authorized users. There are several ways authorized users may disclose sensitive information to others.

This can be done by mistake, as a deliberate illegal action, or the user may be tricked to do so by a Trojan horse attack. The simplest technique to disclose information by an authorized user is to retrieve it from the database, to copy it into an ‘ owned’ object, and to make the copy available to others. To prevent from doing so, it is necessary to control the ability of the authorized user to make a copy (which implies the writing of data). In particular, once a transaction has successfully completed a read attempt, the protection system must ensure that there is no write to a lower security level (write-down) that is caused by a user authorized to execute a read transaction. As the read and write checks are both mandatory controls, a MAC system successfully protects against the attempt to copy information and to grant the copy to unauthorized users. By not allowing higher classified subjects to ‘ write-down’ on lower classified data information flow among subjects with different clearances can efficiently be controlled.

As covert storage channels require writing to objects the *-property also helps to limit leakage of information by these hidden paths. Mandatory integrity policies have also been studied. Biba (1977) has formulated an exact mathematical dual of the Bell-LaPadula model, with integrity labels and two properties: no-write-up in integrity and no-read-down in integrity. That is, low integrity objects (including subjects) are not permitted to contaminate higher integrity objects, or in other words no resource is permitted to depend upon other resources unless the latter are at least as trustworthy as the former.

As an interesting optional feature mandatory security and the Bell- LaPadula (BLP) paradigm may lead to multilevel databases. These are databases containing relations which may appear different to users with different clearances. This is due to the following two reasons: Firstly, not all clearances may authorize all subjects to all data and secondly, the support of MAC may lead to polyinstantiation of attributes or tuples. We will discuss polyinstantiation and the mandatory relational data model in more detail in the next Subsection.

2. 2. 2 The Multilevel Secure Relational Data Model In this Subsection we will define the basic components of the multilevel secure (MLS) relational data model. We will consider the most general case in which an individual attribute value is subject to security label assignment. We will start by using the example database scenario from the Introduction.

– 14 – Throughout the text, whenever we refer to the example we assume the existence of four sensitivity levels, denoted by TS, S, Co, U (where TS> S> Co> U) and a single category only. In each relational schema TC is an additional attribute and contains the tuple classification. Consider the three different instances of relation Project as given in Figure 2. Fig. 2(a) corresponds to the view of a subject s with clear(s) = S. Because of the simple property of BLP (read access rule) users cleared at U would see the instances of Project as shown in Fig.

2(b). In this case the simple property of BLP would automatically filter out data that dominate U. Consider further a subject s with clear (s) = U and an insert operation where the user wishes to insert the tuple into the relation shown in Fig. 2(b). Because of the key integrity property a standard relational DBMS would not allow this operation (Although not seen by user s Alpha as a key already exists in relation Project.

). However, from a security point of view the insert must not be rejected because otherwise a covert signalling channel occurs from which s may conclude that sensitive information he is not authorized to access may exist.

The outcome of the operation is shown in Fig. 2 (c) and consists of a polyinstantiated tuple in MLS relation Project. A similar situation may occur if a subject cleared for the U-level would update in Project as shown in Fig. 2(b) by replacing the null-values with certain data items.

Again, this would lead to polyinstantiation in relation Project. As another example of FIG. 2. Instances of MLS Relation ‘ Project’ (b) Project U Title Subject Client TC Beta, U -, U -, U U Celsius, U Production, U C, U U (a) Project S Title Subject Client TC Alpha, S Development, S A, S S Beta, U Research, S B, S S Celsius, U Production, U C, U U (c) Polyinstantiation at the tuple level Title Subject Client TC Alpha, S Development, S A, S S Beta, U Research, S B, S S Celsius, U Production, U C, U U Alpha, U Production, U D, U U – 15 – polyinstantiation consider that subject s with clear(s)= S wants to update . In systems supporting MAC such an update is not allowed because of the *-property of BLP.

This is necessary because an undesired information flow might occur between subjects cleared at the S-level to subjects cleared at the U-level. Thus, if a S-level subject wishes to update the tuple the update again must result into polyinstantiation. The problem of polyinstantiation arises because of the avoidance of a covert channel. Lampson (1973) has defined a covert channel as a means of downward information flow.

As example let us consider the situation described above once more. If an insert operation is rejected to a subject because of the presence of a tuple at a higher level, the subject might be able to infer the existence of that tuple, resulting in a downward information flow. With respect to security much more may happen than just inferring the presence of a tuple. The success or failure of the service request, for example, can be used repeatedly to communicate one bit of information (0: failure, 1: success) to the lower level.

Therefore, the problem is not only the inferring of a classified tuple, moreover, any information visible at the higher level can be sent through a covert channel to the lower level. The theory of most data models is built around the concept, that a fact of reality is represented in the database only once. Because of polyinstantiation this fundamental property is no longer true for MLS databases thus making the development of a new theory necessary. The state of development of a MLS relational theory has been considerably advanced by the researchers involved in the SeaView project.

For example, see Denning et al. (1988) or Lunt et al. (1990). The following discussion of the theoretical concepts behind the MLS relational data model is mainly based on the model developed by Jajodia and Sandhu (1991a). In the Jajodia-Sandhu model each MLS relation consists of a state-invariant multilevel relation schema RS (A1, C1, ..

. , An, Cn, TC), where each Ai is an attribute defined over a domain dom(Ai), each Ci is classification for Ai and TC is the tuple-class. The domain of Ci is defined by [Li, Hi] which is a sublattice of all security labels. The resulting domain of TC is [lub {Li, i= 1.. n}, lub {Hi, i= 1.

. n}], where lub denotes least upper bound operation in the sublattice of security labels. In the Jajodia-Sandhu model TC is included but is an unnecessary attribute. A multilevel relation schema corresponds to a collection of state-dependent relation instances R, one for each access class c. A relation instance is denoted by Rc (A1, C1, .

.. An, Cn, TC) and consists of a set of distinct tuples of the form (a1, c1, … , an, cn, tc) where each ai I dom (Ai), c ? ci, ci I [Li, Hi], and tc = lub – 16 – {ci, i= 1.

. n}. We use the notion t[Ai] to refer to the value of attribute Ai in tuple t while t[Ci] denotes the classification of Ai in tuple t. Because of the simpleproperty of BLP, t[Ai] is visible for subjects with clear(s) ? [Ci]; otherwise t[Ai] is replaced with the null-value.

The standard relational model is based on two core integrity properties: the key property and the referential integrity property. In order to meet the requirements for MLS databases both have been adapted and two further properties have been introduced. In the standard relational data model a key is derived by using the concept of functional dependencies. In the MLS relational model such a key is called apparent key. Its notion has been defined by Jajodia et al. (1990).

For the following we assume RS (A1, C1, … An, Cn, TC) being a MLS relation schema and A (AI{A1, …

, An}) the attribute set forming its apparent key. [MLS Integrity property 1]: Entity Integrity. A MLS relation R satisfies entity integrity if and only if for all instances Rc and t I Rc 1. Ai I A ? t[Ai] ? null 2. Ai, Aj I A ? t[Ci] = t[Cj] 3. Ai I A ? t[Ci] ? t[CA] (CA is classification of key A) Entity integrity states that the apparent key may not have the null value, must be uniformly classified and its classification must be dominated by all classifications of the other attributes.

[MLS Integrity property 2]: Null Integrity. R satisfies null integrity if and only if for each Rc of R the following conditions hold: 1. For every tIRc, t[Ai]= null ? t[Ci] = t[CA] 2. Rc is subsumtion free, i. e.

does not contain two distinct tuples such that one subsumes the other. A tuple t subsumes a tuple s, if for every attribute Ai, either t[Ai, Ci] = s[Ai, Ci] or t[Ai] ? null and s[Ai] = null. Null integrity states that null values must be classified at the level of the key and that for subjects cleared for the higher security classes, the null values visible for the lower clearances are replaced by the proper values automatically. The next property deals with consistency between the different instances Rc of R. The inter-instance property was first defined by Denning et al.

(1988) within the SeaView framework, later corrected by Jajodia and Sandhu (1990b) and later again included in SeaView by Lunt et al. (1990). [MLS Integrity property 3]: Inter-instance Integrity. R satisfies the interinstance integrity if for all instances Rc of R and all c’ < c a filter function s produces Rc’. In this case Rc’ = s(Rc, c’) must satisfy the following conditions: – 17 – 1. For every t I Rc such that t[CA] ? c’ there must be a tuple t’ I Rc’ 2.

There are no additional tuples in Rc’ other than those derived by the above rule. Rc’ is made subsumtion free. The inter-instance property is concerned with consistency between relation instances of a multilevel relation R. The filter function s maps R to different instances Rc (one for each c’

By using filtering a user may be restricted to that portion of the multilevel relation for which the user is cleared. If c’ dominates some security levels in a tuple but not others, then during query processing the filter function s replaces all attribute values the user is not cleared to see by null-values. Because of the use of this filter function a shortcoming in the Jajodia-Sandhu model has been pointed out by Smith and Winslett (1992). Smith and Winslett state that s introduces an additional semantics for nulls. In the Jajodia-Sandhu model a null value can now mean ‘ information available but hidden’ and this null value cannot be distinguished from a null-value representing the semantics ‘ value exists but not known’ or a null-value with the meaning ‘ this property will never have a value’. In a database all kinds of nulls may be present nd at a certain security level it may be hard for the subjects to say what should be believed at that level.

Let us now draw our attention to polyinstantiation. As we have seen in the example given above polyinstantiation may occur on several different occasions. For example, because of a user with low clearance trying to insert a tuple that already exists with higher classification, because of a user wanting to change values in a lower classified tuple, but it may also occur because of a deliberate action in form of a cover story, where lower cleared users should not be supported with the proper values of a certain fact. Some researchers state that using polyinstantiation for establishing cover stories is a bad idea and should not be permitted. However, if supported it may not occur within the same access class.

[MLS integrity property 4]: Polyinstantiation Integrity. R satisfies polyinstantiation integrity if for every Rc and each attribute Ai the functional dependency A Ci ® Ai (i= 1.. n) holds. Property 4 states that the apparent key A and the classification of an attribute correspond to one and only one value of the attribute, i. e.

polyinstantiation may not occur within one access class. In many DBMSs supporting a MLS relational data model multilevel relations exist only at the logical level. In such systems multilevel relations are with t'[A, CA] = t[A, CA] and for Ai I A t'[Ai, Ci] ={ t[Ai, Ci], if t[Ci] ? c' , otherwise. – 18 – decomposed into a collection of single-level base relations which are then physically stored in the database. Completely transparent multilevel relations are constructed from these base-relations on user demand. The reasons behind this approach are mostly practical.

Firstly, fragmentation of data based on its sensitivity is a natural and intuitive solution to security and secondly, available and well-accepted technology may be used for the implementation of MLS systems. In particular, the decomposition approach has the advantage that the underlying trusted computing base (TCB) needs not to be extended to include mandatory controls on multilevel relations and this helps to keep the code of the TCB small. Moreover, it allows the DBMS to run mostly as an untrusted application on top of the TCB. We will come back to this issue in Section 3 when discussing different implementations of Trusted DBMSs. 2.

2. 3 MAC-based Structural Limitations Although being more restrictive than DAC models MAC techniques need some extensions to be applied to databases efficiently. In particular, we see as limitations the following drawbacks in multilevel secure databases and mandatory access controls

based on BLP: · Granularity of security object It is not yet agreed about what should be the granularity of labeled data. Proposals range from protecting whole databases, to protecting files, protecting relations, attributes, or even certain attribute values. In any case, careful labeling is necessary because otherwise it could lead to inconsistent or incomplete label assignments.

· Lack of automated security labeling technique Databases usually contain a large collection of data, serve many users, and labeled data is not available in many civil applications. This is the reason manual security labeling is necessary which may result in an almost endless process for large databases. Therefore, supporting techniques are needed, namely guidelines and design aids for multilevel databases, tools that help in determining the relevant security objects, and tools that suggest clearances and classifications. · N-persons access rules Because of information flow policies higher cleared users are restricted from writing-down on lower classified data items.

However, organizational policies may require that certain tasks need to be carried out by two or more – 19 – persons (four-eyes-principle) having different clearances. As an example onsider subjects s1, s2 with clear(s1) > clear(s2), data item d with class(d) = clear(s2) and the business rule that writing of s2 on d needs the approval of s1. Following Bell-LaPadula's write-access rule would require the same level of clearance for s1 and s2. This may be inadequate for business applications of MLS database technology.

2. 3 The Adapted Mandatory Access Control Model Adapting mandatory access controls to better fit into general purpose data processing practice

and offering a design framework for databases containing sensitive information are the main goals of the Adapted Mandatory Access Control (AMAC) model. In order to overcome the MAC-based limitations stated above AMAC offers several features that assist a database designer in performing the different activities involved in the design of a database containing sensitive information. For AMAC as a security technique for databases we see the following advantages: · The technique supports all phases of the design of a database and can be used for the construction of discretionary protected as well as for the construction of mandatory protected databases.

· In the case mandatory protection is required a supporting policy to derive database fragments as the target of protection is provided. This overcomes the discussion about what should be the granularity of the security object in multilevel systems. · In the case mandatory protection is required automated security labeling for security objects and subjects is supported. Automated labeling leads to candidate security labels that can be refined by a human security administrator if necessary. This overcomes the limitation that labeled data often is not available.

· In AMAC security is enforced by using database triggers and thus can be fine-tuned to meet application dependent security requirements. For example, the n-eyes-principle may be supported in some applications and may not in others where information flow control is a major concern of the security policy. We will first give a general overview of the AMAC technique which is followed by a more formal discussion and an example. – 20 – 2.

3. 1 AMAC General Overview Adapted mandatory security belongs to the class of role-based security models which assume that each potential user of the system performs a certain role in the organization. Based on their role users are authorized to execute specific database operations on a predefined set of data. The AMAC model does not only cover access control issues but includes in addition a database design environment with main emphasis on the security of resulting databases. Resulting databases may be implemented in DBMSs supporting DAC only or supporting DAC and MAC. The technique combines well known and widely accepted concepts from the field of data modeling with concepts from the area of data security research.

By using AMAC the following design phases for security critical databases can be identified. (1) Requirements Analysis and Conceptual Design. Based on the role they perform in the organization the potential users of the database can be classified into different groups. For different roles data and security requirements may differ significantly. The Entity-Relationship (ER) model and its variants serve as an almost de facto standard for conceptual database design and have been extended in AMAC to model and describe security requirements.

The security and data requirements of each role performed in the organization are described by individual ER-schemas and form the view (perception) of each user group on the enterprise data. Please note, in this setting the notion of a view denotes all the information a user performing a certain role in the organization is aware of. This information includes data, security requirements, and functions. Thus, the notion of views appears different from that in a DAC environment.

In order to arrive at a conceptualization of the whole information system as seen from the viewpoint of the enterprise AMAC uses view integration techniques in a further design step. The resulting conceptual database model is described by a single ER-schema extended by security flags indicating ecurity requirements for certain user roles. (2) Logical Design. In order to implement the conceptual schema into a DBMS a transformation from the ER-schema into the data model supported by the DBMS in use is necessary. AMAC contains general rules and guidelines for the translation of ER-schemas into the relational data model.

Output of the transformation process is a set of relational schemas, global dependencies defined between schemas and necessary for database consistency during further design steps, and a set of views, now describing access requirements on relation schemas. If the DBMS that should hold the resulting database is only capable to support DAC the relational schemas are candidates for implementation and the view descriptors are used for discretionary access controls. In the case the DBMS under consideration supports MAC further design activities are – 21 – necessary. The Requirements Analysis, Conceptual and Logical Design phases in AMAC are described by Pernul and Tjoa (1991). (3) The AMAC security object.

In order to enforce mandatory security it is necessary to determine security objects and security subjects which are both subject to security label assignments. In AMAC a security object is a database fragment and a subject is a view. Fragments are derived by using structured database decomposition and views are derived by combining these fragments. A fragment is the largest area of the database to which two or more views have access in common.

Additionally, no view exists that has access to a subset of the fragment only. Pernul and Luef (1991) have developed the structured decomposition approach and the automated labeling policy. Their work includes techniques for a lossless decomposition into fragments and algorithms to keep fragmented databases consistent during database update. It should be noted that a database decomposition into disjoint fragments is a natural way to implement security controls in databases. (4) Support of automated security labeling. As in most IT applications labeled data is not available, AMAC offers a supporting policy for the automated security labeling of security objects and security subjects.

Automated labeling is based on the following assumption: The larger the number of users cleared to access a particular fragment, the lower is the sensitivity of the contained data and thus, the lower is the level of classification that needs to be provided for the fragment. This assumption seems to be valid because a fragment that is accessed by many users will not contain sensitive information and at the other side, a fragment that is accessible for few users only can be classified as being highly sensitive. Views (respectively the users having the view as their access window to the data) are ordered based on the number of fragments they may access (they are defined over) and additionally based on the assigned classifications for the fragments. In general, a view needs a clearance that allows the corresponding users to access all fragments the view is defined over.

The suggested classification class(F) applies to the whole fragmental schema F as well as to all attribute names and type definitions for the schema while the suggested clearance clear(V) to all transactions executing on behalf of a user V. It should be noted that classifications and clearances are only candidates for security labels and may be refined by a human database designer if necessary. (5) Security Enforcement. In AMAC the fragments are physically stored and access to a fragment may be controlled by a reference monitor. Security is enforced by using trigger mechanisms. Triggers are hidden rules that can be fired (activated) if a fragment is effected by certain database operations. In databases security critical operations are the select (read access), the insert, – 22 – elete, and update (write accesses) commands. In AMAC select-triggers are used to route queries to the proper fragments, insert-triggers are responsible to decompose tuples and to insert corresponding sub-tuples into proper fragments, and update- and delete-triggers are responsible for protecting against unauthorized modification by restricting information flow from high to low in cases that could lead to an undesired information transfer. The operational semantics of the AMAC database operations and the construction of the selectand insert-triggers are outlined by Pernul (1992a). . 3. 2 A More Technical View on AMAC and an Example In AMAC security constraints are handled during database design as well as during query processing. During database design they are expressed by the database decomposition while during query processing they are enforced by the trigger mechanisms. In the following we will give the technical details of the decomposition process, the decomposition itself, the automated security labeling process, and certain integrity constraints that need to be considered in order to arrive at a satisfactorily fragmentation. In

AMAC it is assumed that Requirements Analysis is performed on an individual user group basis and that the view on the database of each user group is represented by an Entity-Relationship (ER) model. The ER model has been extended to cover in addition to data semantics the access restrictions of the user group. The next design activity is view integration. View integration techniques are well established in conceptual database design and consist of integrating the views of the individual user groups into a single conceptual representation of the database. In AMAC the actual integration is based on a traditional approach and consists of two steps: integration of entity types and integration of relationship types (Pernul and Tjoa, 1991). During the integration correspondences between the modeling constructs in different views are established and based on the different possibilities of correspondences the integration is performed. After the integration the universe of discourse is represented by a single ER diagram extended by the access restrictions for each user group. The next step is the transformation of the conceptual model into a target data model. AMAC offers general rules for the translation into the relational data model. The translation is quite simple and results into three different types of modeling constructs: relation schemas (entity type relations or relationship type relations), interrelational dependencies defined between relation schemas, and a set of view descriptors defined on relation schemas and representing security requirements in the form of access restrictions for the different user groups. 23 – In the relational data model user views have no conceptual representation. The decomposition and labeling procedure in AMAC is build around the concept of a user view and this makes a simple extension of the relational data model necessary. Let RS(ATTR, LD) be a relation schema with

ATTR a set of attributes {A1,… , An}. Each AiIATTR has a domain dom(Ai). LD is a set of functional dependencies (FDs) restricting the set of theoretically possible instances of a relation R with schema RS (i. e. ? i dom(Ai)) to the set of semantically meaningful. A relation R with schema RS is a set of distinct instances (tuples) {t1,… , tm} of the form where ai is a value within dom(Ai). Let RS1(ATTR1, LD1) and RS2(ATTR2, LD2) be two relation schemas with corresponding relations R1 and R2. Let X and Y denote two attribute sets with XIATTR1 and YIATTR2. The interrelational inclusion dependency (ID) RS1[X]IRS2[Y] holds if for each tuple tIR1 exists at least one tuple t'IR2 and t[X]= t'[Y]. If Y is key in RS2 the ID is called key-based and Y is a foreign key in RS1. Let V={V1,… , Vp} be a set of views. A view Vi (ViIV, i= 1.. p) consists of a set of descriptors specified in terms of attributes and a set of conditions on these attributes. The set of attributes spanned by the view can belong to one or more relation schemas. View conditions represent the access restrictions of a particular user group on the underlying base relations. For each user group there must be at least one view. The concepts defined above serve as the basis of an AMAC conceptual start schema SS. SS may be defined by a triple SS(A, GD, V), where: A = {RS1(ATTR1, LD1),… , RSn(ATTRn, LDn)} is a set of relation schemas, GD = {ID1,… , IDk} is a set of key-based IDs, and V = {V1,… , Vm} is the set of views. In the case discretionary protection is sufficient, the relational schemas are candidates for implementation in a DBMS, the views may be used to implement content-based access controls and the set GD of global dependencies may be associated with an insert-rule, a delete-rule, and a modification-rule in order to ensure referential integrity during database peration. In the case DAC is not sufficient and MAC should be supported it is necessary to determine the security objects and subjects and to assign appropriate classifications and clearances. In order to express the security requirements defined by means of the views a decomposition of SS into single level fragments is necessary. The decomposition is based on the derived view structure and results in a set of fragmental schemas in a way, that no view is defined over a subset of a resulting schema only. A single classification is – 24 – ssigned to each fragmental schema and the decomposition is performed by using a vertical, horizontal, or derived horizontal fragmentation policy. A vertical fragmentation (vf) results into a set of vertical fragments (F1,… , Fr) and is the projection of a relation schema RS onto a subset of its attributes. In order to make the decomposition lossless the key of RS must be included in each vertical fragment. A vertical fragmentation (vf) R=(F1,… , Fr) of a relation R is correct, if for every tuple tIR, t is the concatenation of (v1,… vr) with vi tuple in Fi (i= 1.. r). The (vf) is used to express ‘ simple’ security constraints that restrict users from accessing certain attributes. The effects of (vf) on an existing set of FDs have been studied by Pernul and Luef (1991) and the authors show that if R is not in 3NF (third normal form) some FDs might get lost during a decomposition. In order to produce a dependency preserving decomposition in AMAC they have suggested to include virtual attributes (not visible for any user) and update clusters in vertical fragments in the case a schema is not in 3NF. A horizontal fragmentation (hf) is a subdivision of a relation R with schema RS(ATTR, LD) into a subset of its tuples based on the evaluation of a predicate defined on RS. The predicate is expressed as a boolean combination of terms, each term being a simple comparison that can be established as true or false. An attribute on which a (hf) is defined is called selection attribute. A (hf) is correct, if every tuple of R is mapped into exactly one resulting fragment. Appending one horizontal fragment to another leads to a further horizontal fragment or to R again. A (hf) is used to express access restrictions based on the content of certain tuples. A derived horizontal fragmentation (dhf) of a relation Ri with schema RSi(ATTRi, LDi) is partitioning RSi by applying a partitioning criterion that is defined on RSj (i? j). A (dhf) is correct if there exists a key-based ID of the form Ri[X]IRj[Y] and each tuple tIRi is mapped into exactly one of the resulting horizontal fragments. A (dhf) may be used to express access restrictions that span several relations. A view Vi (Vi IV) defined on A represents the area of the database to which a corresponding user group has access. Let F (F= ViCVj) be a database fragment then F represents the area of the database to which two groups of users have access in common. If F= Vi Vj, then F is only accessible by users having view Vi as their interface to the database. In this case, F represents data which is not contained in Vj and must therefore not be accessible for the corresponding user set. From the point of view of a mandatory security policy a certain level of assurance must be given that users Vj are restricted from accessing F. In AMAC this is given by separation. For example, fragment (Vi – 25 – Vj) is separated from fragment (VjVi) and fragment (Vi CVj) even if all fragments belong to the same relation. The construction of the fragments makes a structured database decomposition necessary and in order to support mandatory access controls, the access windows for the users is constructed in a multilevel fashion such that only the necessary fragments are combined to form a particular view. Let Attr(V) be the attribute set spanned by view V and let the subdomain SD(V[A]) be the domain of attribute A valid in view V (SD(V[A])IDom(A)). Two particular views Vi and Vj are said to be overlapping, if: $Ao(AIAttr(ViCVj) and

SD(Vi[A])CSD(Vj[A]) ? ?, otherwise, Vi and Vj are called isolated. The process of decomposing A (A={RS1(ATTR1, LD1),… , RSn(ATTRn, LDn)}) is performed for any two overlapping views and for each isolated view by using the (vf), (hf), and (dhf) decomposition operations. It results in a fragmentation schema FS={FS1(attr1, ld1),… , FSm(attrm, ldm)} and a corresponding set of fragments F (F={F1,… , Fm}). If Ei ATTRi = Ej attrj (i= 1.. n, j= 1.. m) the decomposition is called lossless and if Ei LDi I Ej ldj (i= 1.. , j= 1.. m) it is called dependency preserving. Please note that (hf) or (dhf) may result in additional FDs. A fragmental schema FSjIFS is not valid if for any view V ($Fj'IFj) (V? Fj', VUFj). Here, V? F denotes that users with view V have access to fragment F while VUF means that F is not included in view V. To illustrate the concepts defined above we will apply the fragmentation policy to the example given in the Introduction of this Chapter. We assume, that the Requirements Analysis has been performed and that the resulted ER model has been translated into the following start schema: SS = (A={ Employee ({SSN, Name, Dep, Salary}, {SSN ® Name, Dep, Salary}), Project ({Title, Subject, Client}, {Title ® Subject, Client}), Assignment ({Title, SSN, Date, Function}, {Title, SSN ® Date, Function})}, GD ={AssignmentDatabase SecurityIProjectDatabase Security, Assignment[SSN]IEmployee[SSN]}, V = {V1, V2, V3, V4, V5}) The security policy of the organization requires to represent the following conditions on the security: · View V1 represents the access window for the management of the organization under consideration. Users with view V1 should have access to – 26 – the whole database. Views V2 and V3 represent users of the pay-office department. Their requirements include access to Employee and Assignment. For V2 access to Employee is not restricted. However, access to attribute Function should only be provided in the case the employees' Salary ? 100. Users V3 should only have access to employees and their assignments in the case the attribute Salary ? 80. · View V4 has access to Project. However, access to attribute Client should not be supported in the case the subject of a project is ' research'. · View V5 represents the view of the users of the quality-control department. For them to perform their work it is necessary t