Database Security Models – A Case Study

Robert Dollinger
University of Wisconsin Stevens Point
Stevens Point, WI, 54481
USA
rdolling@uwsp.edu

Abstract – **Since information stored in databases is usually considered as a valuable and important corporate resource, security is a major issue in any database management system, particularly those that use sensitive information. Database security cannot be seen as an isolated problem because it is affected by other components of a computerized system as well, like the operating system. Therefore many of the security models developed for trusted OSs can be adapted to DBMSs. Security models are the basic theoretical tool to start with when developing a security system. This still seems to be an issue which is insufficiently understood and it may be an explanation for the actual "security crisis" of information systems. Most corporations try to address their security problems by simply "patching" their existing systems to eliminate identified vulnerabilities. In most cases this is already too late and on the long term such a strategy (or lack of strategy) is a sure looser. In this paper we advocate for the use of security models as an approach to systematically address the security issue starting from the early stages of the system design.**

I. INTRODUCTION

Since information stored in databases is usually considered as a valuable and important corporate resource, security is a major issue in any database management system, particularly those that use sensitive information. A database contains data of various degrees of importance and it is shared among a wide variety of users, so it needs to be protected and managed because any change to the database can affect it and affect other related databases.[1][9] Database security cannot be seen as an isolated problem because it is affected by other components of a computerized system as well, like the operating system. Therefore many of the security models developed for trusted OSs can be adapted to DBMSs. However there are several differences between OSs and DBMSs, from a security perspective like, **object granularity** (OSs typically act at file level, DBMSs at various lower levels), OSs deal with physical objects, DBMSs deal with logical objects, presence of semantic correlations among data in databases, DBMSs store and manage metadata etc. [10][11] Security models are the basic theoretical tool to start with when developing a security system. This still seems to be an issue which is insufficiently understood and it may be an explanation for the actual "security crisis" of information systems. Most corporations try to address their security problems by simply "patching" their existing systems to eliminate identified vulnerabilities. In most cases this is already too late and on the long term such a strategy (or lack of strategy) is a sure looser. In this paper we advocate for the use of security models as an approach to systematically address the security issue starting from the early stages of the system design. We briefly review the main security models proposed for database and operating systems, and conclude by a case study of how the Clark Wilson model maps to the Windows NT security system.

II. SECURITY MODELS

The role of any security system is to preserve *integrity of an operational system* by enforcing a *security policy* defined by a *security model*. System integrity refers both to data integrity, i.e. data is correct and accurate, and system integrity that is the system is in operation and works correctly. System integrity is achieved by rigorous control and management of subjects (users, processes) to objects (data, programs). This control is governed by a set of rules and objectives called a ***security policy***. *Security policies* are governing principles adopted by organizations. They capture the security requirements of an organization specify what security properties the system must provide and describe steps an organization must take to achieve security. *Security models* are formal descriptions of security policies. Security models are useful tools for evaluating and comparing security policies. Security models allow us to test security policies for completeness and consistency. They describe what mechanisms are necessary to implement a security policy.[1]
Security models are described in terms of the following elements:

- **Subjects** - entities that request access to objects.
- **Objects** - entities whose accesses are controlled by the security system.
- **Access modes** – type of operation performed by subject on object (read, write, create, etc.)
- **Policies** – enterprise wide accepted security rules.
- **Authorizations** – specification of access modes for each subject on each object.
- **Administrative rights** – who has rights in system administration and what responsibilities administrators have.
- **Axioms** – basic working assumptions.

III. CLASIFICATION OF DATABASE SECURITY MODELS

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:

A. Discretionary security models, DAC

Specify the rules under which subjects can, at their discretion, create and delete objects, and grant and revoke authorizations for accessing objects to others. That is:

- Govern the access of users to information on the basis of user's identity and predefined discretionary "rules" defined by the security administrator;
- The rules specify, for each user and object in the system, the types of access the user is allowed for the object.

Typical DAC models are: the Access Matrix Model [3][7] and Take-Grant Model [10]. The main characteristics of these models are:

- allow **the management of several types of discretionary privileges:**
  1. **The account level** - at this level the DBA specifies the particular privileges that each account holds independently of the relations in the database.
  2. **The relation (or table) level** -at this level we can control the privilege to access each individual relation or view in the database.
- **privileges can be specified using views** – views can be used to conveniently provide refined read access to selected columns and/or rows of one or more tables according to the view's definition specified as a query.
- **privileges can be revoked** – this may be a complex operation due to possible propagation of privileges from one user to the other.
- **privileges can be propagated using the grant option** – a user having the grant option on a privilege can propagate this privilege to other users.
- **one can specify limits on propagation of privileges -**
  1. **Horizontal propagation limits** - an account B given the GRANT OPTION can grant the privilege to at most i other accounts.
  2. **Vertical Propagation limits** - it limits the depth to which an account can pass on the privilege in terms of levels.

DAC models are flexible and suitable for various types of systems and applications like commercial and industrial environments. They have the disadvantage of not providing real assurance on the satisfaction of the protection requirements, and they are not able to impose any restriction on the usage of information once it is obtained by a user and makes systems vulnerable to attacks (like Trojan Horses). Information is "leaked" by getting around the set of discretionary access controls.

B. Mandatory security models, MAC

Not only control the access of subjects to objects, but regulate the flow of information between objects and subjects. That is, govern the access to the information by the individuals on the basis of the classification of subjects and objects in the system.

Most representative models of this class are: Bell-La Padula, Biba Multilevel Integrity Model, Sea View, Jajodia & Sandhu, Smith & Winslett models.

Research on such models produced models of multilevel databases as a way of limiting security risks by using multilevel security enforcement.[5][6][7]

For example **Bell-La Padula** is a multilevel security model using multiple security levels. Each **security level** is described by two components: **classification** and **categories**, where category may be an application or organization descriptor (e.g., Nato, Nuclear etc.). The model is based on the subject-object paradigm:

- subjects are active elements that can execute actions;
- objects are passive elements that can contain data.

The model considers 4 access modes executable by subjects on objects:

- Read-only or Read
- Append (writing without reading)
- Execute (executes an object /program)
- Read-write or Write

and uses 4 **Classification Levels** for subjects/programs and objects/resources: **(1) Top secret, (2) Secret, (3) Confidential, (4) Unclassified.**

Functionality is based on the following axioms:

(1) Simple security (ss) property: a subject may have read or write access to an object only if the clearance of the subject dominates the security level of the object.

(2) Star (*) property:

- a subject can only read objects **at or above** their level
- a subject can only write objects **at or below** their level

(3) Tranquility principle: no subject can modify the classification of an active object.

(4) Discretionary property (ds-property): every current access must be present in the access matrix: that is, a subject can exercise only accesses for which it has the **necessary authorization**.

(5) Non-accessibility of inactive objects: a subject cannot read the contents of an inactive object.

(6) Rewriting of inactive objects: a newly activated object is assigned an initial state independent of the previous activations of the object.

MAC models are suitable to certain kinds of environments where the users and objects can be classified and provide a high level of certification for security, being based on un-forgeable labels. Problems like Trojan Horse can be avoided. The main limitations of such systems are:

- **Rigidity** – mandatory models are too rigid and inapplicable to some environments;
- **Clearance Assignment** - it is not always possible to assign clearances to users of commercial information systems or to assign sensitivity levels to data.
- **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 persons (four-eyes-principle) having different clearances.

C. Adapted Mandatory Access Control, AMAC

Have been developed as an attempt to overcome certain limitations of mandatory security systems. AMAC is a security technique that focuses on the design aspect of secure databases. The main goals of the Adapted Mandatory Access Control (*AMAC*) model is provide a model based on mandatory access control, which (1) is better fit into general purpose data processing practice, and (2) offers a design framework for databases containing sensitive information. 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.

D. Personal Knowledge Approach

Is concentrating on enforcing the basic law of many countries for the informational self-determination of humans. The personal knowledge approach is focused on protecting the privacy of individuals by restricting access to personal information stored in a database or information system. The main goal of this security technique is to meet the right of humans for informational self-determination as requested in Constitutional Laws of many countries. In this context, privacy can be summarized as the basic right for an individual to choose which elements of his/her private life may be disclosed. In the model, all individuals, users as well as security objects, are represented by an encapsulated person object (in the sense of object-oriented technology). The data part of an object corresponds to the knowledge of the individual about himself/herself and his/her relationship to other persons. The operation part of an object corresponds to the possible actions the individual may perform. The approach is built on the assumption that a person represented in the database knows everything about himself/herself and if he/she wants to know something about someone else represented in the database that person must be asked. Knowledge about different persons cannot be stored permanently and therefore must be requested from the person whenever the information is needed. To achieve this high goal, the personal knowledge combines techniques of relational databases, object oriented programming, and capability based operating systems.

E. Clark and Wilson Model

Tries to represent common commercial business practice in a computerized security model. We discuss this model in more detail in the following section.

IV. THE CLARK AND WILSON MODEL

This model was first summarized and compared to MAC by Clark and Wilson (1987) and is based on concepts that are already well established in the pencil-and-paper office world.[4] These are the notion of security subjects, (constraint) security objects, a set of well-formed transactions and the principle of separation of duty. If we transfer these principles to the database and security world we interpret them as follows:

The users of the system are restricted to execute only a certain set of transactions permitted to them and each transaction operates on an assigned set of data objects only.

More precisely, the Clark and Wilson approach can be interpreted in the
following way:

(1) *Security subjects* are assigned to roles. Based on their role in an organization users have to perform certain functions. Each business role is mapped into database functions and ideally at a given time a particular user is playing only one role. A database function corresponds to a set of (well formed) transactions that are necessary for the users acting in the role. In this model it is essential to state which user is acting in what role at what time and for each role what transactions are necessary to be carried out. To control the unauthorized disclosure and modification of data Clark and Wilson propose access to be permitted only through the execution of certain programs, well formed transactions, and that the rights of users to execute such code be restricted based on the role of each user.

(2) *Well-formed transactions*. A well-formed transaction operates on an assigned set of data and assurance is needed that all relevant security and integrity properties are satisfied. In addition it should provide logging and atomicity and serializability of resulting subtransactions in a way that concurrency and recovery mechanisms can be established. It is important to note, that in this model the data items referenced by the transactions are not specified by the user operating the transaction. Instead, data items are assigned depending on the role the user is acting in. Thus, the model does not allow ad-hoc database queries.

(3) *Separation of duty*. This principle requires that each set of users being assigned a specific set of responsibilities based on the role of the user in the organization. The only way to access the data in the database is through an assigned set of well-formed transactions specific to the role each of the users play. In those cases where a user requires additional information, another user

(which is cleared at a higher level) acting in a separate role has to use a well formed transaction from the transaction domain of the role he is acting in to grant the user temporary permission to execute a larger set of well-formed transactions. Moreover, the roles need to be defined in a way that makes it impossible for a single user to violate the integrity of the system.

The Clark and Wilson model can be summarized as a set of nine rules (5 Certification and 4 Enforcement rules):

Certification Rules

C1 (IVP Certification) - The system will have an IVP for validating the integrity of any CDI.

C2 (Validity) - The application of a TP to any CDI must maintain the integrity of that CDI. CDIs must be certified to ensure that they result in a valid CDI.

C3 - A CDI can only be changed by a TP. TPs must be certified to ensure they implement the principles of separation of duties & least privilege.

C4 (Journal Certification) - TPs must be certified to ensure that their actions are logged.

C5 - TPs which act on UDIs must be certified to ensure that they result in a valid CDI.

Enforcement Rules

E1 (Enforcement of Validity) - Only certified TPs can operate on CDIs.

E2 (Enforcement of Separation of Duty) - Users must only access CDIs through TPs for which they are authorized.

E3 (User Identity) - The system must authenticate the identity of each user attempting to execute a TP.

E4 (Initiation) - Only administrator can specify TP authorizations.

V. A CASE STUDY: WINDOWS NT AND MS SQL SERVER

The Clark and Wilson model partitions all data in a system into two: **constrained data items** (CDI) and **unconstrained items** (UDI), data items for which integrity must be ensured. The (CDI) are objects that the integrity model is applied to and (UDI) are objects that are not covered by the integrity policy (e.g. information typed by the user on the keyboard). Two procedures are then applied to these data items for protection. The first procedure integrity verification procedure (IVP), verifies that the data items are in a valid state (i.e., they are what the users or owners believe them to be because they have not been changed). The second procedure is the transformation procedure (TP) or well-formed transaction, which changes the data items from one valid state to another.

A. Overview of Windows NT Security Model

Windows NT was built to incorporate networking, security and audit reporting as services within the operating system. The Windows NT Security Model was designed to monitor and regulate access to objects and it maintains security data for each user, group, and object. The basic components of the Windows NT security model are:

- Logon Process
- Local Security Authority (LSA)
- Security Account Manager (SAM)
- Security Reference Monitor (SRM)

Logon process, which accept logon request from users. It is the process that accepts the user's initial interactive logon, password, authenticates it, and grants entry into the system. The **LSA** is the heart of the security subsystem. It verifies the logon information from the SAM database and ensures that the user has permission to access the system. It generates access token, administers the local security policy defined in the system and is responsible for auditing and logging security events.

Security Account Manager (SAM) is the database that contains information for all user and group account information and validates users.

Security Reference Monitor provides real-time services to validate every object access and action made by a user to ensure that the access or action is authorized. This part enforces the access validation and audit generation policy defined by the Local Security Authority. Resources, such as processes, files, shares, and printers are represented in Windows NT as objects. Users never access these objects directly, but Windows NT acts as a proxy to these objects, controlling access to and usage of these objects. A subject in Windows NT is the combination of the user's access token plus the program acting on the user's behalf. Windows NT uses subjects to track and manage permission for the programs each user runs. This is the most basic object in Windows NT, Security Identifiers (SIDs), are internal numbers used with a Windows NT system to describe a user and a group uniquely amongst other Windows NT systems. Owners, users or groups are assigned permissions to an object and are identified by their SID. The security information for an object is encoded in a special data structure called the Security Descriptor (SD). The SD for an object contains the following components:

- **Owner Security ID,** this is the SID of the user or group who owns the object.
- **Group Security ID,** this is a primary group associated with the object. It is optional and is not used by the Windows NT file-system system security. It is included to simplify the implementation of a POSIX-compliant file system.
- **Discretionary Access Control List,** it identifies the user and group SIDs that are to be granted or denied access for the object.
- **System ACL,** this is what controls the auditing message that the system will generate.

Each user of Windows NT has a unique security ID (SID). When a user logs on, Windows NT creates a security

access token. The token contains information about the user account which includes a security ID for the user, as well as other security IDs for the groups to which the user belongs, and permissions assigned to the user. The security access token created for the logged in user is attached to all processes that are started by the user. When the process tries to access a particular object, the SRM checks to see whether any of the SIDs in the security access token attached to the process match a list called the access-control list (ACL) attached to that process. The ACL contains access-control entries (ACE) for each user authorized to access the object Windows NT includes an auditing mechanism that can be used to audit successful and unsuccessful attempts for operations on files and directories. This mechanism enables you monitor events related to system security, to identify any security breaches, and to determine the extent and location of any damage.

B. Windows NT Interpretation of the Clark-Wilson model

The table bellow shows the correspondence between the Clark-Wilson Security Model rules and their Windows NT counterpart:

Rule	Clark-Wilson Security Model	Windows NT Security Model
RULE 1. (C1)	The system will have an IVP for validating the integrity of any CDI.	In Windows NT there is a local security authority (LSA) which checks the security information in the subject's access token with the security information in the object's security descriptor.
RULE 2. (C2)	The application of a TP to any CDI must maintain the integrity of that CDI.	In Windows NT, most subjects cannot change the attribution of the objects, but some subjects have this privilege, such as administrator But this is only limited to some special users. So this rule is not applied to Windows NT strictly.
RULE 3. (C3)	A CDI can only be changed by a TP.	As mentioned above some special users can change attribution of the objects, and no other methods can be applied to change objects.
RULE 4. (C4)	Subjects can only initiate certain TPs on certain CDIs.	In Windows NT, the subject's access token includes what kinds of operations are permitted. Only when information of the access token is consistent with the information in the object's security descriptor, the operation is allowed.
RULE 5. (C5)	CW-triples must enforce some appropriate separation of duty policy on subjects.	In Windows NT, administrator can do anything. So this rule is not applied.
RULE 6. (E1)	Certain special TPs on UDIs can produce CDIs as output.	In Windows NT, users can change the object from without ACL state to with ACL state. Generally, this operation is performed by Administrator.
RULE 7. (E2)	Each TP application must cause information sufficient to reconstruct the application to be written to a special append-only CDI.	In Windows NT, audit services can collect information about how the system is being used.
RULE 8. (E3)	The system must authenticate subjects attempting to initiate a TP.	In Windows NT, any user has her or his SID, and any process in behalf of this user copies the same SID. By this way, Windows NT can authenticate subjects attempting to initial a TP.
RULE 9. (E4)	The system must only permit special subjects (i.e., security officers) to make any authorization-related lists.	In Windows NT, only administrator can do and view some high security events.

Based on the information presented above, it is easy to see that the security mechanisms of Windows NT satisfy the axioms of the Clark-Wilson model and that the Clark-Wilson model could be implemented with security mechanisms of Windows NT.

VI. CONCLUSIONS

Database security is not an isolated problem - in its broadest sense it is a total system problem. Database security depends not only on the choice of a particular DBMS product or on the support of a certain security model, but also on the operating environment, and the people involved. Further database security issues include requirements on the operating system, network security, add-on security packages, data encryption, security in statistical databases, hardware protection, software verification, and others.

In general, it is important to recognize that by itself, a security model is not a panacea to information security issues. Security models have theoretical limits and do not establish security. Security models are generally used to evaluate existing secure system designs rather than a guide to developing secure systems. It is an effective method for verifying security. Security models are important and necessary, but focusing and relying only on a model can lead to a false sense of security. Confidentiality, integrity, availability are very important and much related aspects of security. To achieve any of these goals, the objective is to

strike a balance between applying generally accepted models and incorporating the latest security technologies and products, applying security patches, risk management, adhering to industry standards and guidelines, and implementing sound management principles to achieve secure systems.

There is a growing interest in database security and the approaches reported above show that there has been considerable success in developing solutions to the problems involved. Public interest has increased dramatically, but it is only recently that the issue of security outside the research community receives a priority properly reflecting its importance. Database security has been a subject of intensive research for almost two decades [3][4][5][6][7], but still remains one of the major and fascinating research areas of the day. It is expected that changing technology will introduce new vulnerabilities to database security. Together with problems not yet adequately solved database security promises to remain an important area of future research.

VII. REFERENCES

[1] Amoroso, E. - *Fundamentals of Computer Security Technology*, Prentice Hall,1994.

[2] Batini, C., Ceri, S., and Navathe, S. B. *Conceptual Database Design: An Entity-Relationship Approach*, The Benjamin/Cummings Company, Inc., 1992;

[3] Biskup, J. A., "General Framework for Database Security", *Proc. European Symp. On Research in Computer Security (ESORICS'90)*, Toulouse, France, 1990;

[4] Blake, S.Q., *The Clark-Wilson Security Model,* May 17, 2000

[5] Burns, R. K., "A Conceptual Model for Multilevel Database Design", *Proc. 5th Rome Laboratory Database Workshop*, Oct. 1992;

[6] Denning, D. E., Lunt, T. F., Schell, R. R., Heckman, M., and Shockley, W. R., "A multilevel relational data model", *Proc. 1987 Symp. on Research in Security and Privacy*, IEEE Computer Society Press, 1987;

[7] Denning, D. E., Lunt, T. F., Schell, R. R., Shockley, W. R., and Heckman, M., "The SeaView Security Model", *Proc. 1988 Symp. on Research in Security and Privacy*, IEEE Computer Society Press, 1988;

[7] Elmasri, R., and Navathe, S. B., *Fundamentals of Database Systems*, 3.ed., Addison-Wesley, 2000;

[8] Fernandez, E. B., Summers, R. C., and Wood, C, *Database Security and Integrity*, Addison-Wesley, Reading, MA, System Programing Series, 1981;

[9] Gollmann, D., *Computer Security - Database security*, John Wiley& Sons Ltd, 1999.

[10] Pernul, G., "Database Security", in: *Advances in Computers*, Vol. 38, M. C. Yovits (Ed.), Academic Press, 1994, pp. 1 - 74.

[11] Pfleeger, C.P., *Security in Computing – Database Security*, PTR, 1997.