Aspects of data security in general-purpose data base management systems

Serge M. Miranda

IRIA

CENTRE DE RENNES

IRISA

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tel. 954 90 20

Rapports de Recherche

N° 100

ASPECTS OF DATA SECURITY IN GENERAL-PURPOSE DATA BASE MANAGEMENT SYSTEMS

Serge M. MIRANDA

Novembre 1981

ASPECTS OF DATA SECURITY IN GENERAL-PURPOSE DATA BASE MANAGEMENT SYSTEMS

S.M. MIRANDA

ABSTRACT

With the ever-increasing use of computers for storing large volumes of vital data, the problems involved in providing data security have been receiving very great attention from researchers.

In this paper,we try to investigate the various aspects of data security in a general-purpose Data Base Management System (DBMS)

Data security concerns data manipulation which encompasses two phases : retrieval and access (read/write).

Privacy protection is associated with the retrieval phase and integrity is attached to the modification phase.

In this paper, we detail both parts of data security and give an overview of the potential solutions to each feature.

ASPECTS DE LA SECURITE DANS LES BASES DE DONNEES DU TYPE "GENERAL"

S. M. MIRANDA

RESUME

Dû à l'utilisation toujours croissante des ordinateurs pour stocker de grandes quantités de données vitales, les problèmes concernant la sécurité des données ont reçu une attention particulière de la part des chercheurs.

Dans cet article nous essayons d'approfondir les différents aspects de la sécurité dans les bases de données du type "général".

La sécurité des données est associée aux deux aspects de la manipulation des données : la recherche et la modification.

La confidentialité est liée à la phase de recherche alors que l'intégrité est liée à la phase de modification.

Nous détaillons ces deux composants de la sécurité des données et faisons un survol des solutions potentielles/existantes.

ASPECTS OF DATA SECURITY IN GENERAL-PURPOSE DATA BASE MANAGEMENT SYSTEMS

S.M. Miranda
CERISS-INRIA
Univ. des Sciences Sociales
31070 TOULOUSE Cedex FRANCE

Introduction

In the history of data processing to date, we can clearly distinguish two particular trends, the trends toward :

- data independence
- data usability

In every area involving user/system interaction we can trace a steady improvement in the quality and level of the human interface. The trend is always away from the irrelevant complexities of the underlying machine and physical data organization and toward a higher and more casual user-oriented level of expression. An obvious example is the progression from machine code to simple assemblers, to macro assemblers, to high level procedural languages, to high level non-procedural languages, to object-oriented languages (using the abstract data type concept), and to program generators. At the same time the computation power kept on increasing in a wide range as a result of the introduction of computer networks which offer a number of advantages in providing efficient and economical computing to the users. The time is ripe for the next step in this progression which leads to a cross-section namely distributed data bases.

These evolutions have had a huge impact on the design of security systems in the DBMS. Initially, the security system was not an independent module of the DBMS. It relied heavily on the security system of the underlying operating system (it was also a parallel evolution with the access-methods of the information systems). Then the need for more powerful controls, for a finer grain of protection, and the inadequacy of identification/authentication schemes, led to the design of security systems as a component of a DBMS. An intermediate step was associated with the advent of logical data independence in information systems. This feature has led to the concept of subschema or view used in DBMS to give the users some restricted pictures of the schema ("sensitive" segments in IMS...). This concept is of primary importance for security purposes (non-data-dependent security) since users cannot see and have access to the hidden data. Subsequently the advent of high level non-procedural languages made it possible to define control systems which were no longer "non-discretionary-oriented." This trend is in general noticeable in relational-system research (in SYSTEM-R, the "user is" the data base administrator (DBA)).

A DBMS will be said to be "general-purpose" when it offers a high-level non-procedural interface to the users; i.e. it appends a new degree of data independence -independence with regard to the access strategies-.

Presentation of Data Security

Definitions

Data security is a major concern in computer systems today and this is reflected in current literature. Data security is a crucial issue in the data management systems (DBMS), more important than in any other type of software. As a matter of fact, data are the most sensitive part of any system and its loss or compromise can have disastrous consequences.

CH1522-2C/80/0000-0046$00.75 © 1980 IEEE

Reprinted from PROCEEDINGS OF THE 1980 SYMPOSIUM ON SECURITY AND PRIVACY, April 1980

A data base, in contrast with a file, may be viewed as a dynamic model for some real-life system which can be shared by a large population of users having different needs, rather than a passive collection of data. Because of the possibility to retrieve large amounts of integrated and interrelated data via a powerful and convenient user interface, data security needs to be clearly defined and designed as a component module of a general-purpose DBMS.

Data security is intrinsically twofold; it includes the privacy protection and the soundness of the stored information. Maintaining the security of the data base can be viewed as protecting the data against illegal or invalid retrieval or modification. The first aspect of data security concerns access control in direct access and in "indirect" access to the data so as to restrict users to retrieve only the data structures or values for which they receive authorization. The second aspect is called integrity, which basically encompasses two major features: the update synchronization of shared data by concurrent users ("external integrity") and the correctness of data modification depending on predefined constraints ("internal integrity"). Both aspects require system routines for surveillance, threat monitoring,...(privacy protection), journaling, dumping, resiliency,..(integrity).

Moreover, the unique identification and authentication (based on properties of the requesting or requested entities are needed before performing security controls. We shall not consider here these aspects of data security and assume these informations are provided by the underlying operating system (given to the DBMS in the form of a user profile.)

To summarize, let us say that data security mechanisms must ensure that a user is authorized to have access to data for the performance of specified operations, that any changes resulting from a user's alteration of the data base are valid without affecting the consistency of the whole data base.

Discretionary versus non-discretionary security systems.

It is important to specify the kind of user interface we want for a DBMS. Most of the existing security systems are "non-discretionary", non-discretionary in the sense that the DBA(s) define the security system and its protection data which cannot be changed at the discretion of users. Examples of such systems are "transaction-oriented" systems, where application designers define application programs and decide what information should be available and accessible to the applications [75]. An extreme example is a military security system where a double hierarchy is defined, ,including a hierarchy of objects associated with "rights", and a hierarchy of subjects related to "clearances" [42,47, 66]. Such systems enable a security formalization and the confinement problem is argued to be decidable [47]. What is interesting about these systems which can model many real time environments is their simple implementation (with some restricting provisions limiting the communications between sensitive levels like the "property" defined in [42] and their strict security policies which could be required in some particular situations. However these systems, which embody policies, (instead of mechanisms) are not flexible, and do not meet the basic goals of general-purpose which must be reflected in the security system, such as ad-hoc processing, casual user orientation, and powerful user interface. Therefore, casual oriented systems ("discretionary" systems) have been considered in the literature [1,71] ...In the remainder of this paper we do not make distinctions between discretionary and non-discretionary systems (except in explicit cases). We want to pay particular attention to security mechanisms in which different policies (hierarchies...) could be embedded.

Granting/revocation of access rigts

Generally speaking, authorization is the responsibility of the DBA or someone delegated by the DBA. The ability to delegate security functions and access rights is of practical importance. Any user in a general DBMS should potentially be authorized to create new data structures, selectively grant and revoke authorizations for his objects to other users as well. The interested reader can refer to [9,26,38] for the detailed description of mechanisms which tackle this issue.

Consideration of the objectives of a general-purpose DBMS for data security

General-purpose DBMS represent the current step in the evolution of the information system. These are characterized by the following objectives which are presented with their consequences for the security system :

Data independence (logical and physical, [14, 76], regarding access-paths [69])

This basic objective implies a clear distinction between the functional levels of the DBMS (external, conceptual, access-path, internal levels). Since a user only manipulates logical entities contained in the schema or subschema, the representation of the security features must be done at the logical level (protection of logical

entities...). Logical independence implies the definition of views or subschemas, which are manipulated by the end-users. The intrinsic property of subschemas is that they offer a direct protection of the hidden data (value-independent security). However, a consequence of the definition of complex subschemas (as in the relational data model) is the existence of intricate mapping involved between the two logical levels (schema-subschema), and the logical level-physical level. This mapping is very important for security purposes since the modifications or security controls (locking...) which are performed at the logical level, must correctly propagate down to the physical level for protection to be maintained. This mapping should be reliable for the security mechanisms stated at the logical level. The separation of the DBMS in different functional levels renders possible a microscopic access to the data by a malevolent or masqueraded user; this would require some form of encryption of data stored in the physical device.*

Data relatability

This objective concerns the schema definition and specification which must faithfully model some real world environment. In order that this representation be reliable (basic need for security), we require a theoretical and rigorous foundation of the treatment of data. This can be done through an axiomatization of the schema to generate the logical structure in the most automatic fashion possible. Relatability could be enhanced by the use of abstract data types in the schema specification and in the application programs. Since the schema models the real world environment, it must embody the security features as part of the data definition. The real world environment is dynamic; therefore, the security system must have dynamic characteristics. Since an end-user need not know the ever-chan ging characteristics of the security system for convenience reasons, the system will have to manage them in a transparent fashion for users, by considering some forms of query modification at the query language level or at the subschema level. Another consequence of the ever-changing characteristics of the real-world environment is that the DDL used for the security definition must encompass some features of the DML (Data Manipulation Language).

Data sharability

The purpose of this concept is to avoid physical data redundancy present in early information systems, which was the source of consistency problems and high storage costs. Lack of physical redundancy (which in some cases can be maintained for performance reasons or for connection data like in the "physical pairing" of IMS) does not mean lack of logical redundancy. Generally the schema offers a user redundant logical structures which are desirable to present simple logical structures ("virtual source" concept in DBTG schema, "virtual pairing" in IMS...). The major consequences of this concept are the need for control of data sharing and modification by concurrent processes ("external integrity") and the inadequacy of user profile (identification/authentication) to achieve security.

Access flexibility

This feature is related to the powerful user interface provided in today's general-purpose DBMS's (a limited interface reduces the ability of a user to manipulate data base). This powerful user interface is characterized by potential ad-hoc processing. The first consequence of ad-hoc processing for security purposes is the possibility for malevolent users to indirectly retrieve the data (problems of inference and reference). Another characteristic is the casual-user orientation attained by providing for high-level non-procedural languages by which the users define what they want without any concern for the access-paths to the data. These access paths are "intelligently" managed by the system at the access-path level below the logical levels (external and conceptual levels). This characteristic requires powerful direct controls (context dependent security, content dependent security) which cannot be provided by the security system of the underlying operating system (insufficiency of the user profiles, passwords, need for privacy and integrity, complex constraints, fine grain of protection...). Therefore an independent security system must be designed for the general-purpose DBMS (we presently have the same trend to embed access control in programming-language design). Moreover, since user privileges are based on the characteristics of the stored data items, the solution of having a table of commands for each group of users is not satisfactory. A survey of the splitting between OS and DBMS security systems with its handling in existing DBMS's is presented in details in [31].

Data base performance/efficiency

The general trade-off in a DBMS (which is crucial in a distributed environment) is :
-what is desirable at the user interface?
-what could be efficiently implemented?
This trade-off is more significant in a DBMS embedding its own security system. The performances of the whole DBMS must

*Example: ADABAS security system

not be drastically cut down by the privacy and integrity controls. In [30], overhead attached to direct access control has been estimated; a retrieval with a security verification which does not depend on the content of the data is in the order of 22 to 32 per cent less efficient than a retrieval without any control while it reachs 187 per cent for a content-dependent security control. Consequently the bulk of security control must be performed at compile time.

Data base administrator (DBA)

The DBA is responsible for the schema definition and specification, for the definition of access paths to data, the definition of access methods in the DBMS based on the access methods of the operating system and the physical distribution of the data on the physical devices. He is also responsible for defining the security system, the protection data (semantic constraints...) and the enforcement rules. For this purpose, the DBA needs powerful tools to enter and manipulate protection data in the security system (high-level data security language for defining and modifying the security informations (both DDL and DML)....).

From the considerations noted above we can point out a dilemma for the enforcement of privacy protection in a general DBMS : the need for powerful controls and data independence require that access restrictions be stated at the logical level, at the highest level possible. This feature has the disadvantage of requiring that the software involved in the logical-physical mapping be secure. However, such a software which can be very large and complex, is not certifiable in the actual state of the art Therefore the dilemma is that we may have non-reliable powerful security system in DBMS !

Design principles for a data security system.

The design principles that apply particularly to protection mechanisms are as follows :

- Least privilege ; every process should operate with the least amount of access rights and power necessary to complete its tasks.
- Least common mechanism (isolation guide); in the structuralization of the system, the amount of common mechanism should be minimized.
- Complete mediation; every access to every object must be checked.
- Powerful and selective controls; direct access (macroscopic or not) and indirect access (masquerading, problems of inference/reference) must be controlled.

Powerful semantic constraints should be defined and enforced for privacy and integrity purposes.

The other characteristics which have been mentioned in the previous section are simplicity of implementation, small execution time overhead, flexibility, and fail-safe default (mechanism based on permission rather than exclusion, important for the inference problem).

Sketch of a data security system

In operating systems, associated with each type of object (files, devices...) there is a monitor which controls the access to these objects . The same concept can be extended to DBMS for the data security system.

In the following we discuss a data security system (Fig.1) for privacy protection. It is convenient to regard all the informations specifying the types of access subjects have to objects as constituting a protection state of the system. The data security system encompasses two major modules, the authorization process and the enforcement process :

- The authorization process which received a good deal of interest in relational data base systems (much more ignored in the other data models). [9,27,28,38,61]

The concept of data security languages [41] which embodies the DDL features with some power of a DML needs to be introduced to the authorization process for privacy protection (and integrity also). The purpose of this language is to enter and modify the protection data specified as part of the data definition (rather than as part of the user definition). A fundamental requirement of a data security system is to provide a convenient interface for the DBA(s) to permit an easy, consistent, and efficient way of defining and maintaining the protection data. This data security language is a powerful and flexible means of expressing complex access restrictions (privacy protection and security). This process also includes the following functions : authorization validation, translation of the data-security language expression into internal representation ; control of the display of the protection data. Some optimization can be performed at that level by storing only some "basic" access rules whose "derived" rules can be deduced by an appropriate algorithm [27] . Here we have not mentioned the kind of user interface we have. We could decentralize authority from the bottleneck of a single DBA at different levels and allow subjects to alter the protection state to a certain extent(case of discretionary systems).

-The enforcement process ("checker","controller").

This process is in charge of transacting with the requester,causing subjects to have access to objects only as permitted by the protection state,consulting the information provided by the authorization process and rendering a decision to the data requests.
The enforcement process determines whether the access request subspace is contained within the subspace characterized by the access restrictions. One of the major issues is to ensure the reliability of the enforcement process at every step (crucial in a distributed environment) Other issues concern the storage and manipulation of the protection data in a performant and reliable fashion.

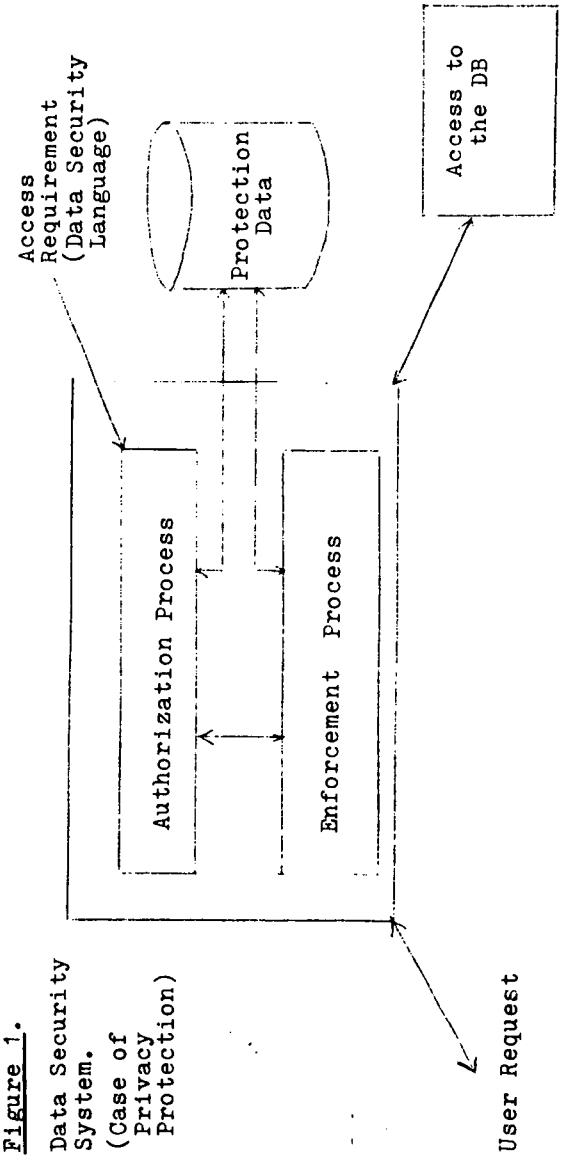

Figure 1. Data Security System. (Case of Privacy Protection)

Privacy Protection and Integrity Features

Privacy protection features

Privacy protection encompasses two general aspects,access control and user interface control. We are going to present them is turn by pointing out the major characteristics of these features.

Access control

The problem of access control has been studied in detail in the case of operating systems where important concepts have been defined : capability [25,51],confinement [4],encapsulation[4],and overall kernel concept [63]. Some of these concepts have been extended to apply to data base management systems. For this reason,we rapidly examine the access control methods defined in operating systems.

Access control in operating systems (OS)

The principles of a protection system were initially formalized as a model defined by [50]. The three basic elements,subjects (Sj),access types (tk), and objects (Oi) define a model for protection data which can be described by means of an access matrix,with subjects corresponding to the rows and objects to the columns. Access types are written in the intersection of rows and columns.Graham extended[34] this model by introducing the concept of domain of protection (for instance,Ring in Multics). The subject can now be regarded as a pair(Uj,dR), The most important aspect of the model is the notion that each process Uj has a unique identification number. These models assume that an access type applies uniformly to an object and all its components. Two basic implementations of this access matrix have been generally used for access control, one by row,the other by column. The two corresponding categories are respectively called "ticket" or "capability"-oriented, and "access-list"-oriented. In both cases protection data must be maintained but the crucial distinction is whether the protection data are maintained by users (capability systems) or in relation with the objects (access-list system). In capability systems,each subject must keep a collection of keys for the objects of interest,keys issued by the owners of these objects with possible granting/revocation privileges(discretionary control orientation). These systems require each subject to save and protect the set of keys which he requires and present the correct key whenever a privilege is requested.

They allow a very rapid implementation of access control at execution time. In access-list systems,the "checker" for each protected object includes or has access to a table of authorized subjects with an indication of the type of access

authorized (non-discretionary control orientation). These systems require that the checker search his table whenever access is requested and oversee that all access conditions are met. The subject only needs to cater to identification/authentication information which will lead to the initialization of his profile. For this user-convenience reason, such systems could be appealing for data base management systems (as they are for file systems).

These access-list systems permit the revocation of privileges without consultation with the subject who is losing the privilege. They permit an auditor to determine the range of access without having the access himself. Generally, they provide adjustability and make it easier to understand the implication of making an authorization. However, because of response-time requirements, the overhead of access-list resolution for each transaction may be prohibitive. This aspect is very important in generalized DBMS where efficiency is one of the basic requirements. This is the reason IMS has a "ticket" orientation visible to the end-user. This corresponds to the protection of the data objects by making use of passwords.

The use of capabilities to check object at execution time could be combined with the use of data types to check object access at compile time.

Access control in data base management systems versus operating systems. The models defined in operating systems are hardly adequate to protect data in a general DBMS. In both access-list and capability systems, what is controlled is access to the container of the data rather than access to the data itself. Neither system limits what a program can do with informations it has derived from data it was authorized to manipulate* (problem of user-interface control crucial in DBMS). In operating systems, control mechanisms permit different users different types of access to the protected objects (Read, Write, Execute, Merge, Sort, Append, Delete..). In data bases, all users are essentially performing read access. Therefore, the access control mechanisms could not be based on the type of access.

More complex controls are required in a general-purpose DBMS to ensure :

-(i) container-dependent security (i.e. "user Uj cannot access the domain SALARY")

-(ii) value-dependent security (i.e. "user Uj can retrieve the domain SALARY if, and only if, salary value is less than $2000")

-(iii) context-dependent security (i.e. "user Uj cannot retrieve the domain SALARY together with the domain NAME "or the DBA doesn't want any user to obtain the following information, "for the same job, and the same qualifications, a male Employee makes more than a female Employee"!!!). These controls expressed by privacy semantic constraints have to be stated in a logical form and cannot be provided by a straightforward capability or access-control mechanism. Except for very rarely altered domains, content- dependent security must be enforced at execution time (the same kind of complexity and checking is required for integrity purposes). Compile-time-protection enforcement appears to be of much higher utility in DBMS than in OS (where a rechecking overhead is generated at each recompilation). Consequently, highly constrained user interfaces are possible in DBMS, but not in OS.

Two basic mechanisms have been defined to implement access control at the external level of a general-purpose DBMS (control of EXTERNAL ACCESS) :

- (i) views (subschema)[8]
- (ii) query modification[70]

The former mechanism offers the users the sensitive data he has the right to manipulate; other data are hidden. Such a view is built (dynamically or staticly) from the global schema and the security constraints.

In the latter mechanism, the user request is considered as a VIRTUAL one. The actual interaction with the data base takes place after appending (logical AND) the concerned security constraints to the user request; consequently such an interaction will not violate security. These security schemes are compared in[55].

Another contrast between the OS and DBMS access control mechanism concerns the grain of protection. If the grain is similar, the data security mechanism provided by the underlying operating system can be used for the DBMS protection (as noted in[42] with Multics). Generally the grain of protection is finer in DBMS than in OS; therefore, we cannot use a capability mechanism to protect the access to a large number of objects, since it would carry a very high cost for storage and manipulation of capabilities. Moreover, in DBMS, different subsets of data may require different protection status. In a DBMS, the definition of the data structure and the different access paths to the data (indices) have to be protected. Encryption could be a basis for the general access-control architecture (only) when the grain of storage encrypted by an individual key matches the protection grain (however, this is inadequate for complex controls such as value dependent security). Encryption is appealing in systems where MICROSCOPIC ACCESS (access to the internal level of a general-purpose DBMS) is possible.

*due to functional access to the data...

Another difference is connected to the handling of multi-object requests. In OS a complex operation is broken down into a set of accesses to individual objects and each access permission is determined independently of the other. In a DBMS, a decision must be made whether the global request should be permitted in the first place.

However, there are some similarities between DBMS and OS for data security. One of the most important similarities concerns the kernel architecture which corresponds to the desirability to partition the software into security-relevant and security-irrelevant portions; it is possible, through the simplification of protection features to reduce the size of the security-relevant portion to a limited "kernel" which could be certified. Such an approach enhances the overall reliability [19,20,21,46,48,63,64,81]

Some proposals to extend the access matrix concept to DBMS have been made [12, 76] They examine the possibility of extending the use of capabilities by associating procedures (which could implement value-dependent security) to them. The size of the matrix can be reduced by introducing user classes and data aggregates. In [12], an enforcement scheme is presented for data-dependent security, based on a security matrix and four security functions which can be invoked at translation or execution time. Value-independent privacy decisions can be enforced at translation time (needs to be exercised only once per request in contrast to value dependent security which requires evaluation for every data value). This distinction is appealing in a batch environment, not in an on-line application where one request at a time is processed.

User interface control

This aspect of privacy protection is concerned with INDIRECT access to the data and includes two basic problems :
- the problem of reference
- the problem of inference.

The problem of reference concerns the disclosure of confidential information that the user has the right to manipulate but not the right to retrieve, whereas the problem of inference concerns the confidential information a user can deduce from a prior (authorized or not) knowledge of the data base. While the problem of reference was given some satisfactory solutions [57,59], that is not yet the case with the problem of inference where currently promising formal research is conducted [2,17,18,45,65,82].
However, these problems are still open fields for investigation. They are problems about the control over what a user is able to do with the information retrieved from the data base [57,61,67].

Problem of reference. Sometimes in order for some types of functional access to be accomplished (i.e., compute the average salary, give a raise of 10% to all salaries) it is necessary for the corresponding application programs to give more rights to the objects that the caller had. This is called "amplification" in [44] . In addition, problems of reference can include the hiding of intermediate results of computation. This problem of reference can be satisfactorily solved by assuring a control over the internal behavior of user's program which interacts with the data base. Examples of shortcomings of direct access control mechanisms are given in [57]. A satisfactory and more efficient solution based on the concept of abstract data type and indirect access to the objects via a "confidence module" [57] whose output channels are strictly controlled, can be envisionned.

Problem of inference . This problem is connected to the powerful user interface available in today's general-purpose DBMS's and to statistical DBMS's [17,45,65,82]. A malevolent user could "infer" confidential data from the outputs of a series of adequate requests (functional access, direct access) for which he is fully authorized (i.e., combine the external knowledge he has of his salary or some other friends and the output of an "appropriate" functional access to deduce the salaries of employees he is not entitled to know). The direct access control solutions like query modification and tagging are shown to be inadequate in [2] .
Proposed solutions are based on the analysis and control of the potential scope of users instead of the actual scope corresponding to direct access control (analysis of data independence) and on query history for a given user. In order to do so, the concept of "confidential propositions" is introduced in [2] and an exhaustive estimate, checking these propositions with the allowable requests, is calculated. If it fails, the request is discarded. The inadequacies of this solution are : first, that it could be too restrictive since most users are not malevolent a priori and it does not meet one of the basic objectives of general-purpose DBMS, availability; second, that such an exhaustive checking is so costly that the introduced overhead will reduce the performance of the system. (The impracticability of this scheme in a distributed environment is obvious). Consequently, this solution although it is formally satisfying is not practically appropriate in its raw form. It could be extended by restricting the checking to a limited set of "confidential propositions" and allowable queries to decrease the control overhead. However, the danger, as pointed out in [18] in this case, is that too much, instead of too little, may be

released. Another promising approach has been proposed. It is based on an information flow graph [59] constituting a minimum security schema which encompasses domains and manipulation procedures as nodes and and access paths as vertices. Such a solution is to be considered with the other axiomatization studies conducted in general-purpose DBMS's(mainly relational ones).

Masquerading("Trojan horse") problem

This is a basic problem in data base security where a malevolent user could either have microscopic access to the raw data (countermeasure using cryptography),or downgrades confidential data from a high sensitive level to a lower level where unauthorized users can retrieve them (provisions similar to"*property" defined in [42] must be taken in non-discretionary security systems like those used in a military environment where hierarchic classes of users and objects are predefined). This problem is enhanced in a distributed environment since "masqueraded" users in a switching node can divert the flow of information received on that node to a malevolent end-user by manipulating the adressing informations (contained in the packet headers when packet switching techniques are used) and issuing proper acknowledgements. The masquerading problem in its general form is connected with the confinement problem. The confinement problem presented in [51] asks whether there exists a mechanism by which a subject who is authorized access to an object can leak the information contained in that object to another subject not having authorized access. If it can be shown that no such mechanism exists in a security system,then that security system is vulnerable to masquerading. It is important to point out that abstract data types are inadequate for preventing such a leakage of information,since the authorized masqueraded user can have access to the encapsulated confidential data and manipulate them with the meaningful operations of the type. However it has been shown in [40] that the confinement problem is undecidable for discretionary-security systems for which the access matrix is "fully general" (no strict hierarchies are defined on the objects (classes) and the subjects (clearances)). These systems are then vulnerable to masquerading attacks.

This is not the case in non-discretionary-security systems [42, 47, 66] ,where the confinement problem is claimed to be decidable and masquerading could therefore be avoived.

Integrity features

The notion of integrity is closely connected with the notion of concurrency control (external integrity) and the schema (semantic integrity). A schema can be viewed as a set of assertions about the DB contents which remain invariant during processing. These assertions are called semantic rules.

Semantic integrity

Semantic integrity encompasses two aspects: the data structure definition and the internal integrity. In [32] the value of predicate calculus is advocated for defining and evaluating integrity constraints which can be as complex as the privacy constraints. These constraints limit the states of a data base to those which conform with some expressed limitations. We can consider a value-dependent security constraint as an integrity constraint in the sense that they both describe properties of and the relationships between data structures or data values,respectively,before granting access to a dedicated user or after some data alteration to check its soundness. The same mechanism can be used for both purposes as in INGRES [70,72] or SYSTEM-R [1,7]. Semantic constraints are more logically associated with the data than with any given user and these will have to be specified as part of the data definition rather than as part of the user definition (therefore the abstract data type concept seems promising for the schema specifications).

Different levels of constraints can be defined in a data model. If we consider the relational model,two types of constraints have been proposed [39] for internal integrity (or privacy protection) : domain constraints which state properties that date items must satisfy (value-dependent orientation) and relation constraints which state properties of relationships that must hold between different relations or subparts of relations (non-value dependent,context-dependent orientation).

Different kinds of constraints can be considered [14,23]. We can differentiate them by whether or not they belong to the schema,by their behavioral properties (static versus dynamic) or by the enforcement time (immediate-after every modification,or delayed at the end of a "transaction"). Static constraints hold for every state of the data base (uniqueness,functionality,range of values for a given field,statical constraint). Dynamic constraints refer either to transitions from one state to another (corresponding to the NEW/OLD clause in System-R) or from non-existence to existence (requirement for a WHEN clause introduced in [75] which indicates at what time the constraint is to be applied). The WHEN clause may also be desirable to limit the number of integrity checking for performance reasons. "Deferred constraints" are

associated with the concept of "transaction" which represent and indivisible unit of work [7,36]. A data base is in a state of integrity before and after a transaction, but not in a intermediate step. Delayed constraints are only checked at the end of the transaction.

External integrity

The problem of internal integrity is increased if we consider concurrent access by more than one user. In addition, the system has to ensure that the users do not interfere with each other during update operations, (problems of "lost update", "dirty read" ...). To this end the system must provide external integrity mechanisms based on a serialization mechanism (locking, serializibility graph, sequencing algorithm) which ensures the correct execution (with regard to integrity) of concurrent transactions.

The concept of TRANSACTION, detailed in [36] and extended to distributed data bases in [37], represents an atomic interaction with the data bases which preserves internal integrity; a transaction encompassed atomic actions with regard to the operating systems; locking ensures that concurrent transactions are serialized (their execution is equivalent to a serial one).

A locking mechanism has been almost exclusively selected in commercial and prototype DBMS's (even in DBTG where the initial proposal based on a "monitoring" feature has been given up in the DBTG-type DBMS from UNIVAC, DMS 1100 [54])

In a general-purpose DBMS, locking can be
- "high-level" (external level) [22]
- "med-level" (access-path level) [58]
- "low-level" (internal level) [5,24]

The major advantages of high-level locking are the natural avoidance of the phantom-entity problem (locking objects which may not yet exist from being created), the reduction of the locking table (storage overhead...), the use of predicates....
- Med-level locking are the ability to lock every access-path to the data in a uniform way and the combination of the advantages of high and low-level locking.
- Low-level locking are the efficiency in terms of availability (only the required entities are locked) and the possibility to use the OS locking features.

Locking has as consequence the danger of deadlock.
Five necessary conditions for deadlock have been pointed out in the literature [24] concurrency (CN1), exclusive locking (CN2) incremential locking (CN3), circular waiting (CN4), lack of preemption (CN5).
Solutions to the deadlock problem are :
- ignorance (!)
- detection/roll-back
- avoidance
- prevention (constraints are introduced in the allocation process to prevent deadlock to occur by avoiding the feasibility of one necessary condition at least; the corresponding solutions are pre-ordering (CN1), preclaiming (CN3), presequencing (CN4), preemption (CN5)

Everest claims that prevention solutions are the only solutions feasible for a general-purpose DBMS [24].
The two solutions which have been largely reported in commercial and prototype DBMS's are :
- detection/roll-back [54], (SYSTEM-R [1,5])
- preclaiming (INGRES...)

Many tradeoffs which are not detailed here are involved in the design of an external-integrity mechanism (granularity of locks, degree of concurrency, consistency, frequency of detection calls....)

An integrity system must also embody system routines for journaling, dumping, recovery, back-out, check-point-restart and detection.

Conclusion

In figure 2 we represent data-security features with their potential or promising solutions.
We do not have place here to detail and discuss these solutions.
Generally speaking, it should be pointed out that the more remote we are from the physical data structures, the more we can define powerful security mechanisms but the more complex is the implementation and the more unreliable are the solutions.

Acknowledgments

Special thanks are due to Gerry POPEK (UCLA) for his invaluable discussions on the topic of data security and to IRIA which granted me a post-graduate fellowship at UCLA in 1977-1978.

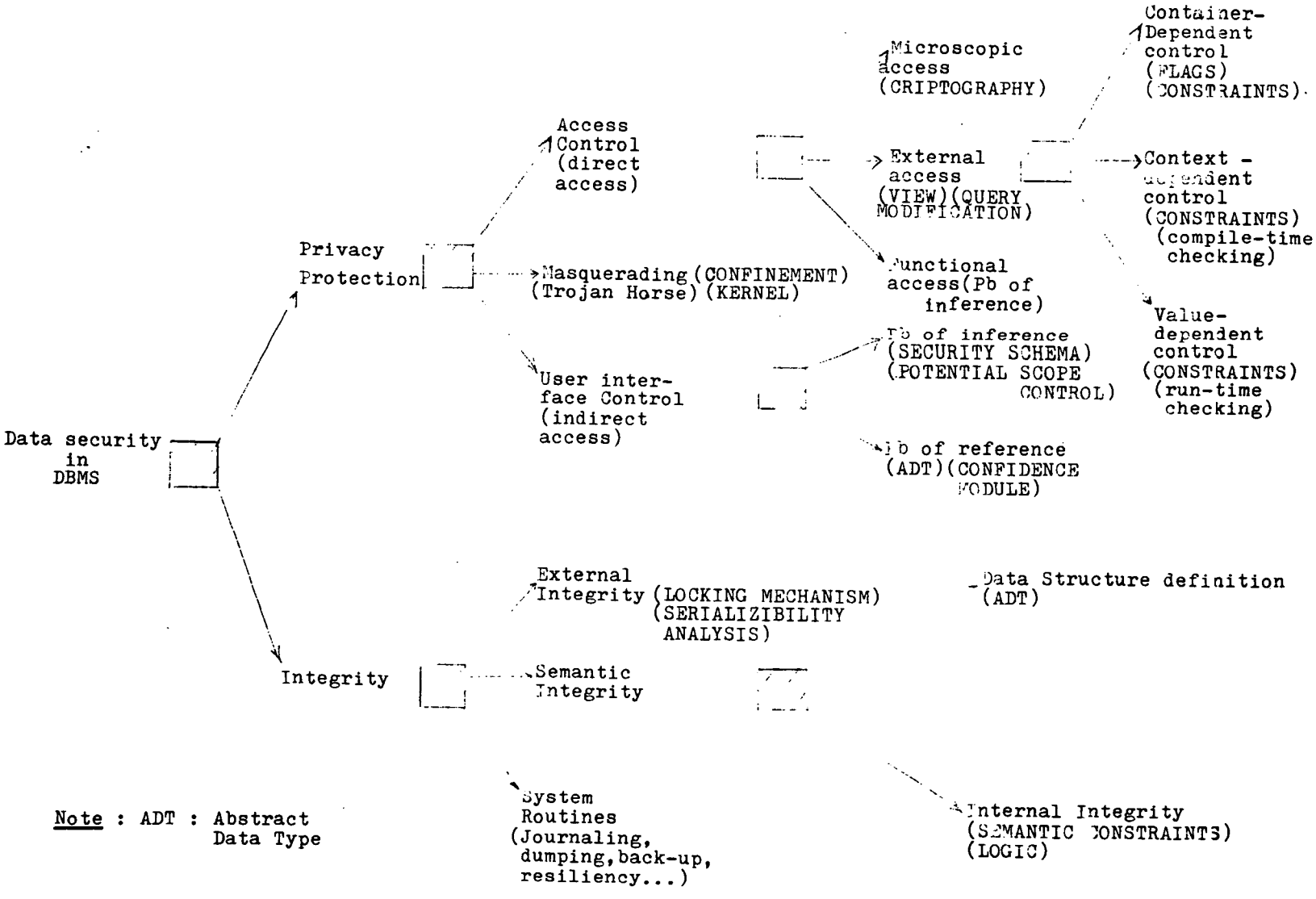

Figure 2. Data Security in general-purpose DBMS (SOLUTIONS)

References

(1) Astrahan et Al "SYSTEM-R : a relational approach to data base management" ACM-TODS, Vol 1, n°2, June 1976, pp97-137

(2) Bancilhon, F.M., Spyratos, N. "Protection of information in relational data base management systems" Proc. of Conf. on VLDB. Tokyo, Japan, October 1977

(3) Bayer, R. "Integrity, consistency, recovery in data bases". Proc. ECI Conf. 1976 Lect. Notes in Comp. Science, Vol 44, Springer-Verlag (1976)

(4) Bisbey, I.I., Richard, L, Popek, G.J. "Encapsulation: an approach to OS security". USC/ISI report, July 1975 (17 p)

(5) Blasgen, N.W. et Al "SYSTEM-R: an architectural update" IBM research report RJ 2581 (33481) 7-17-1979.

(6) Cash, J., Whinston, A.B., Haseman, W.D. "Security for the G Plan system". Information Systems, Vol 2, pp 41-48, Pergamon Press, 1976.

(7) Chamberlin, D.D., Boyce., Traiger, I.L. "A deadlock free scheme for resource locking in a DB environment" IFIP, August 1974, pp 340-343.

(8) Chamberlin, D.D., Gray, J.N., Traiger, I.L "Views, authorization and locking in a relational data management system" IBM research report, San Jose, CA, RJ 1486, Dec 1974, also Proc. of Nat. Computer Conference, 1976

(9) Chamberlin, D.D. et Al "Data base system authorization". IBM research report RJ 2041, July 1977, San Jose (18 p)

(10) Chin, F.Y. "Security in statistical data bases for queries with small counts" ACM transaction on data base systems, pp 92-104, March 1978.

(11) Codd, E.F. "Access control principles for security and privacy in integrated data banks". IBM Internal Memo (1970), also revised form in BCS Symp. on Relational Data Base Concepts, London, April 1972

(12) Conway, R.W., Maxwell, W.L., Morgan, H.L. "On the implementation of security mesures in information systems". Comm. of the ACM, Vol 15, n°4, April 1972, pp. 211-220.

(13) Conway, R.W., Maxwell, W.L., Morgan, H.L. "Selective security capabilities in ASAP, a file management system". SJCC 72, pp 1181-1185, 1972.

(14) Date, C.J. "An introduction to database systems" Addison Wesley, 14456, 1977 (2nd edition).

(15) Davida, G.I., Kam, J.B. "Data security: theory and pratice" Tech. report, 76-2, E.E. Department, Univ. of Wisconsin, Milwaukee, 1976

(16) Dean, A.L. "Data privacy and integrity requirements for on-line data management systems". Proc. of the ACM Sigfidet Works. on Data Description, San Diego, CA, pp 263-278, 1971

(17) Denning, D.E. et Al "The tracker : a threat to statistical data base security". ACM-TODS, pp 76-96, March 1979.

(18) Dobkin, D., Jones, A.K., Lipton, R. "Secure data bases: protection against user inference". Yale Univ. Research Report #65, April 1976.

(19) Down, D., Popek, G.J. "Design of a secure data base management system". UCLA Research Report, 1976.

(20) Down, D., Popek, G.J., "Approaches to data management security". UCLA research Report, 1977

(21) Down, D., Popek, G.J. "A kernel design of a secure data base management system" Proc. of the Conf. of Very Large Data Bases, Tokyo, Japan, October 1977.

(22) Eswaran, K.P., Gray, J.N., Lorie, R.A., Traiger, I.L. "The notions of consistency and predicate locks in a data base system". IBM research report, San Jose, CA, RJ 1487, Nov. 1974, also CACM, Vol 19, n° 11, pp 624-633, Nov. 1976

(23) Eswaran, K.P., Chamberlin, D.D. "Functional specifications of a subsystem for data base integrity" Proc. of the Conf. on VLDB, Boston, Mass. Sept 1975

(24) Everest, G.L. "Concurrent update control and data base integrity". On data base management. J.W. Klimbee and K.L. Koffman Rds, 1974, pp 241-268.

(25) Fabry, R.S. "Capability based adressing" CACM, 17, 7, pp 403-412, July 1974

(26) Fagin, R. "On an authorization mechanism". IBM Research Report, RJ 2001 (28091), 5-6-77

(27) Fernandez, E.B., Summers, R.C., Lang, T. "Definition and evaluation of access rules in data management systems". Proc. of the Conf. on Very Large Data Bases, Boston, MA, September 1975.

(28) Fernandez, E.B., Summers, R.C., Coleman C.D. "An authorization model for a shared data base". Proc. of the 1975 ACM SIGMOND International Conf. on Management of Data, San Jose, CA, May 1975, pp 23-31.

(29) Fernandez, E.B., Summers, R.C., Lang, T., Coleman, C.D. "Architectural support for system protection in data base security". IBM Research Report. LA, CA, G 320-2683, Dec. 1976, also Proc. of the Third Annual Symp. on Comp. Architecture, Clearwater, Florida, Jan. 1976.

(30) Fernandez,E.B.,Summers,R.C."Integrity aspects of a shared data base".Proc. of the National Computer Conference, 1976,pp 819-827

(31) Fernandez,E.B.,Wood,C."The relations between operating system and data base system security:a survey".Proc. COMPSAC'77,November 1977

(32) Florentin,J.J."Consistency auditing of data bases".The Computer Journal Vol 17,n°2,pp52-58,Feb.1974

(33) Gardarin,G.,Labeux,P."Scheduling algorithms for avoiding inconsistency in large data base".Proc.of the Conf. on VLDB,Tokyo,Japan,October 1977

(34) Graham,G.S.,Denning,P.J."Protection: principles and practice".Proc.of the SJCC,1972,pp 411-429

(35) Gray,J.N.,Lorie,R.A.,Putzolu,G.R. "Granularity of locks in a shared data base".IBM Research Report,San Jose CA,1975,(also Proc.of the Conf.on VLDB,September 1975).

(36) Gray,J."Notes on data base operating systems".Lecture notes in computer science,edited by J.Hartmanis,Springer Verlag,Berlin,N.York,pp.394-481, 1978.

(37) Gray,J.N."A discussion of distributed systems".Proc.AICA,Bari,10-13 Oct.1979,pp 204-211.

(38) Griffiths,P.P.,Wade,B.W."An authorization mechanism for a relational data base management system".Comm.of the ACM transaction on Data Base Systems,Vol 1,n°3,Sept 1976,pp 242-255

(39) Hammer,M.M.,McLeod,D.J."Semantic integrity in a relational data management system".Proc.of the Conf.on VLDB,Boston,Ma,Sept.1975

(40) Harrison,M.A. et Al."Protection in Operating systems"CACM,19,8,pp 461-471,August 1976.

(41) Hartson. Hsiao,D.K."A semantic model for data base protection languages". Proc.of the Conf.on VLDB,Brussels, Sept.1976.

(42) Hinke,T.H.,Schaeffer,M. "Secure data management systems".System Development Corporation Report,Santa Monica CA,also Rome AIR Development Center Report,Griffiss-AFB,New York,TM-5407 007 00,June 1975,193 p.

(43) Hsiao,D.K."Access control in an online file system".In File Organization.IAG occasional publication, n° 3,1969.

(44) Jones.A.K."Protection in programmed systems". PH-D Thesis, CMU, 1973

(45) Kam,J.B.,Ullman,J.D."A model of statistical data base and their security".ACM Transactions on Data Base Systems,Vol 2,n° 1,March 1977

(46) Kampe,M.,Kline,C.,Popek,G.,Walton,E. "The UCLA data secure UNIK operating System prototype".UCLA report,July 1977,(126 p.).

(47) Karger .P.A. "Non-discretionary access control for decentralized computing systems". Master Thesis,MIT, ILCS-ITR-179,May 1977

(48) Kemmerer,R.A. "Formal verification of the UCLA security kernel...".PH-D Thesis,UCLA-ENG 7956,SDPS 79001, June 1979.

(49) Lang,T.,Fernandez,E.B.,Summers,R.C. "A system architecture for compile time actions in data bases".IBM Research Report,Los Angeles,CA,G320-2682,December 1976.

(50) Lampson,B.W."Protection".Operating system review,Vol 8,1,Janvier 1974.

(51) Lampson,B.W."A note on the confinement problem".CACM,Vol 16,n° 10, pp 613-615,Oct.1973.

(52) Lomet,D.B."A practical deadlock avoidance algorithm for data base systems".IBM Research Report,RC6288, November 1976.

(53) Lorie,R.A."Physical integrity in a large segmented data base".IBM Research Report,San Jose,CA,1974.

(54) Macri.P.P. "Deadlock detection and resolution in a CODASYL-based data management system"

(55) McLeod,D.J."A framework for data base protection and its application to the INGRES and SYSTEM-R DBMS".Proc. COMPSAC'77,November 1977

(56) Minker,J."Performing inferences over relational data bases".Proc.of the ACM-SIGMOD Workshop on Data Description,Access and Control,San Jose,CA, May 1975.

(57) Minsky,N."Intentional resolution of privacy protection in data base systems".Comm.of the ACM,March 1976, Vol 19,n° 3,pp 146-159

(58) Miranda,S."Formal integrity system for distributed data bases".PH-D Thesis (in French)(forthcoming)1980

(59) Moore.C.G.,Conway,R. "Program predictability and data security".Cornell University Report,TR-74-212 (211), September 1974

(60) Nuez,F."An architecture for a distributed data base with very confidential utilizations".Symp.on data sharing.Aix.France,March 79,pp 629-652

(61) Owens,R.C."Evaluation of access authorization characteristics of derived data sets".Proc.of the ACM Sigfidet Workshop on Data Description,Access and Control.San Diego,CA,pp 263-278, 1971.

(62) Popek,G.J."A note on secure data base design".UCLA/ARPA Report,1974, 10 p.

(63) Popek,G.J."Principles of kernel design".UCLA quarterly,Oct.1976,Vol 4, n° 4,pp 59-73.

(64) Popek,G.J.,Kampe,M.,Kline,C.S.,Stoughton,A.,Urban,M. and Walton,E.J."UCLA secure unix".In Proc.1979,AFIPS NCC, Vol 48,AFIPS Press,Arlington,Va,pp 355-364.

(65) Reiss,S.P."Security in data bases: a combinatorial study".IACM,Vol 26, n° 1,pp 45-58,January 1979.

(66) Rzepka,W.E."Consideration in the design of a secure DBMS".Rome AIR Development Center Report,AD/A-039 169 Griffiss AFB,New York,March 1977.

(67) Saltzer,J.H."Technical possibilities and problems in protecting data in computer systems". In R.Dierstein,H. Fiedler and A.Shulz,Datenschutz und Datensicherung,J.P.Bachem Verlag,Cologne,Germany,Sept 1976,pp 27-36.

(68) Scott,G.M."Auditing large scale data bases".Proc.of the Conference on VL DB,Tokyo,Japan,Oct.1977,pp 515-521.

(69) Senko,M.E."DIAM as a detailed example of the ANSI-SPARC architecture" Proc.IFIP TC-2 working conference Jan.1976.

(70) Stonebraker,M,Wong,E. "Access control in a relational data base system by query modification".UC.Berkeley Memorandum,n°ERL M438,5-4-74.

(71) Stonebraker,M. "High level integrity assurance in relational data management systems". UC Berkeley Memorandum,n° EPL M473,August 1974.

(72) Stonebraker,M."Implementation of integrity constraints and views by query modification".Proc.of the ACM-SIGMOD Conference on Management of Data,San Jose,CA,pp 65-77,May 1975

(73) Summers,R.C.,Fernandez,E.B. "Access control in a shared data base" IBM Research Report,Los Angeles,CA,G-320 2666,December 1974,24 p.

(74) Summers,R.C.,Coleman,C.D.,Fernandez E.B."A programing language approach to secure data base access".IBM Research Report,Los Angeles,G 320 2662,May 1974,18 p.

(75) Summers,R.C.,Fernandez,E.B."Integrity aspects of a shared data base".Proc. of the National Computer Conference, 1976,pp 819-827

(76) Tsichritziz,D."A note on protection on data base systems".Proc.of the Workshop IRIA on Computer Security, August 1974

(77) Turn,R."Privacy protection in data banks : principles and costs".Rand Corporation Report,Santa Monica,CA, P5296,September 1974.

(78) Turn,R.,Shapiro,N.Z.,Juncosa,M.L. "Privacy and security in centralized vs decentralized data bank systems". Rand Corporation Report,Santa Monica CA,P5346,February 1975.

(79) Wee,R.S.S."Problems of the dynamic sharing of data in a relational data base environment.".IBM Scientific Report,United Kingdom,UKSC 0067,June 1975.

(80) Woodward,F.G.,Hoffmann,L."Worst case cost for dynamic data element security decisions". Proc.of the ACM Annual Conference,New York,1974.

(81) Wulf,W.A. et Al."Abstraction and verification in ALPHARD : introduction to language and methodology".Tech-Report,CMU,Pittsburgh,1976. CACM,Vol 20,n° 8,pp 553-563,August 1977.

(82) Yu,C.T.,Chin,F.Y."A study on the protection of statistical data bases". Proc.SIGMOD Int.Conference on Management of data,August 1977

(83) Zanon,G.,Minker,J."Data base integrity". Proc.Symposium formal tools in data bases,CERT-DERI,Toulouse,France, December 1979.