Security of Database Systems: Authorization Features and Mechanisms

Csaba Egyhazy

TR 86-32

Security of Database Systems: Authorization Features and Mechanisms

1.0 Introduction

Database security has become an essential issue in assuring the integrity, protection, and reliability of the data stored in a database management system (DBMS). The authorization mechanism is the component of the database security system which has the primary responsibility of safeguarding the previously defined data and access rules needed for database access control. The data and rules for authorization control assist in the enforcement of access controls regarding the list of authorized users, the data objects which the authorized users are allowed to manipulate, and the operations that these users can perform on these objects. As part of its tasks the authorization mechanism can grant or deny access to any user or group of users as appropriate.

The next section introduces three known authorization mechanisms. It begins with an overview of each and concludes by comparing their drawbacks and merits.

Centralized and decentralized control to implement any of the three authorization mechanisms is discussed in Section 3.0 and 4.0. Comments and conclusions are given in Section 5.0.

To offer clarification regarding some of the terminology associated with the authorization process a distinction will be made with respect to the terms, authorizer and non-authorizer. An authorizer is defined as any individual who can designate access rules and delegate authorization privileges. Furthermore,

a distributed database is defined as a collection of data which is distributed over a computer network. Processing a query usually requires multiple accesses to geographically separated databases. The data in relational databases are stored in tables, called relations. The relational data model allows the use of powerful, set-oriented, associative expressions instead of the one record at a time primitives of the procedural models.

2.0 Overview and Comparison of Three Authorization Mechanisms

The types of controls that can be implemented and enforced through the authorization mechanisms can be classified as value independent controls, value dependent controls, context dependent controls, and in relation to statistical databases, statistical controls. Value independent controls allow for decisions on whether to grant or deny a user's access request based on the names of the data objects, and not their values. Value depend controls allow for a decision on whether to grant or deny a user's access request based on the values of the data objects themselves in conjunction with an optional predesignated access predicate. The access predicate allows an authorizer to limit the range of values that a given user may access for a given data object. Context dependent controls make use of references to system variables in the predesignated access predicate. Some of the system variables employed in the access predicate may refer to the time of day, a specific terminal number, or a terminal address. Statistical controls add another dimension by allowing the access matrix to contain references to typical statistical operators such as sum and average in addition to the other

privileges.

One implementation employed in the authorization process utilizes a mechanism known as the authorization matrix. The authorization matrix consists of a table in which the rows identify authorized DBMS users and the columns correspond to the data objects that are to be controlled. Access privileges and the specification of operations on given data objects are assigned by filling in the appropriate areas and fields corresponding to a particular user. A null entry signifies that a particular data object cannot be accessed by a particular user. Some of the operations that can be specified in the matrix involve the selection, modification, insertion and deletion of data objects. These privileges can be assigned either singularly or in any combination.

Since owners of files may designate access privileges to their files through the utilization of the concept of ownership, the authorization matrix must be updated constantly in order to reflect these changes.

Another mechanism employed in the authorization process is based on the concept of views. Through the utilization of views a database administrator can build views that consist of various combinations of data objects taken from underlying base tables [1]. Views allow for the hiding of sensitive information from unauthorized users by restricting the data that they can actually access. Through the view mechanism the range and scope of values that can be seen by a user can be specified through the use of an optional access predicate. A drawback in this mechanism is that it does not allow an authorizer to specify the operations that an

authorized user can perform on those objects. An additional mechanism would have to be set up to complement the view mechanism in order to handle the specification of privileges over the objects. Another drawback in the view mechanism itself becomes cumbersome in cases where users are to be given different levels of access to different part of a given relation or base table. Furthermore, in the view mechanism, implemented in System R [2], if a record is either inserted or modified through a view, the system does not check the record with regards to its consistency in terms of the original view definition. In other words, it is totally possible to corrupt the data stored in a view without being aware of it. If this situation occurs, upon retrieval of the data in the view, the corrupted and inconsistent data will never be displayed to the user since the data does not satisfy his query in terms of the original view definition. As a result, the data will always be invisible from the user's point of view. The problem then arises where the inconsistent data will go undetected, and will still remain in the underly view. In a distributed environment checks must be incorporated to test and validate any insertions or modifications against the predefined view definition in order to avoid this pitfall.

One advantage of the view mechanism over the authorization matrix is that the view is not as prone to the dynamic changes in authorization states under the circumstances outlined before. Since the creation of views is based on the definition of underlying base tables, and the views themselves are designed and authorization is determined by the database administrator, the possibility of having users creating new views, deleting views,

and modifying views is reduced. Another point is that the views themselves are static in nature once they are defined, which assists in eliminating constant and dynamic changes in updating the authorization rules and data. Although at times changes may occur that will call for changes in the authorization data and corresponding rules, these changes can be made by the DBA in an orderly and predetermined fashion. As a result the need for immediate changes and immediate updates can be reduced significantly.

System R has two commands that deal with the granting and revocation or recall of privileges which complements the view mechanism. These commands are GRANT and REVOKE, respectively. The granting of privileges to a user does not allow him/her to propagate these privileges to another user automatically. This allows for the control of the propagation of privileges, which may get out of hand. To propagate privileges to another user, an authorized user must have also been given the capability of propagating privileges. In this case, one can consider the user as an authorizer.

An important aspect of the recall or revocation of privileges is that it is a mechanism whose effect filters down to more than one user. For example if one user grants a privilege to another, and this user in turn propagates the privilege to a third party user, the original authorizer, in this case the first user, can revoke the privileges to both the second and third party users simply revoking the second party's privileges. The advantage of this approach is that it gives the original authorizer the responsibility of keeping an audit trail of the fan out

related to the proliferation of the propagation of privileges.

A third authorization mechanism utilizes the idea of attaching a classification level to the data objects and the establishment of clearance levels for DBMS users. Within the classification levels additional caveats can be employed to restrict authorization and access even further. This type of environment is utilized in military installations where huge amounts of information is handled and processed constantly. In this type of environment two rules are utilized in the authorization process. The rules are commonly known as the simple security property and the confinement property, also known as the star property or *-property. The simple security property stipulates that no subject has read access to a given object that has a classification level greater than the security clearance of the subject.

The star property stipulates that no subject has append access to an object whose security level is not at least the current security level of the subject; that no subject has read-write access to an object whose security level is not equal to the current security level of the subject; and no subject has read access to an object whose security level is not at most the current security level of the subject. The importance of the star property is that it not only assists in authorization control, but it also forces additional controls on authorized users. The added controls assist in the resolution of the problem concerning the flow of information. The star property comes into play in those instances where an authorized user with a high security clearance may be tempted to utilize his/her

clearance level to copy data objects with high classification levels into files with lower classification levels, thus making this highly sensitive data available to unauthorized users. Very sensitive data and communications messages must be encrypted, which adds costs in both the hardware necessary to encrypt and decrypt the information and in the time it takes to complete a transfer.

3.0 Centralized Authorization Control

In a distributed database system an authorization mechanism can be set up to handle the concept of local views versus global views. The local views can be utilized to designate the data fragments that can be seen by users at his/her respective node, whereas global views can be implemented to designate the data fragments that users can have access to and manipulate which do not reside at the local node. The concept of a local view allows for the hiding of sensitive data at the level of the local node. The global view concept can be implemented in a distributed environment to control the authorization of access privileges over data fragments dispersed over different nodes. Together with the Grant and Revoke mechanism, the concept of local and global views can be tailored to control the propagation of privileges that involve data manipulation operations.

Data is not the only entity for which access must be restricted. Programs and other applications must also be controlled. The Grant and Revoke scheme may be utilized to control access to programs and other database related applications.

The system dictionary contains all the information and data required for the authorization mechanism. The contents of the dictionary themselves must also be controlled to avoid unauthorized tampering with the authorization matrix and other security related information. Access to the system dictionary must be limited only to the DBA and other security officials. Any changes to the dictionary must be logged just like any other transaction processed by the DBMs in order to have an adequate and accurate audit trail for evaluation, recovery, and analysis purposes. In addition, the log must be secured in order to preserve its contents for those who are privileged to see it.

In INGRES (Interactive Graphics and Retrieval System) the authorization mechanisms include access control restrictions that specify the types of queries which a given user is allowed to use [3]. These restrictions, as part of the query language QUEL, define integrity constraints on a relation or attribute and level protection onto a relation or attribute.

To define integrity constraints for a set of data, or a relation, the DBA qualifies this in a query to the database. For example, in QUEL, 5

```
RANGE OF E IS EMPLOYEE
INTEGRITY CONSTRAINT IS E.SALARY $8000
```

states that all employees in the EMPLOYEE relation must earn more than $8,000.

Protection of a relation may also be defined by the DBA through a query. this is examplified in the query.

```
RANGE OF E IS EMPLOYEE
```

```
PROTECT EMPLOYEE FOR ALL (E.SALARY; E.NAME)
WHERE E.MANAGER = *
```

which allows only a person's manager to alter his salary.

Views are also available for use as protection mechanisms. These too must be defined by the DBA. Above the aforementioned mechanisms, the DBA has the ability to create sharable relations and allocate access privileges to other users. He also has the ability to destroy any relations in his database, except for certain system generated ones or private relations created by other users.

Implementation of the INGRES authorization mechanism is accomplished by using system catalogs which have predefined names and are created for each database. The two catalogs of interest are the PROTECTION and INTEGRITY catalogs.

The PROTECTION and INTEGRITY catalogs contain protections and integrity predicates for each user as defined by the DBA. These predicates are appended to queries made by the restricted user. The predicates are simply ANDed to the original query when the query is parsed and a modified query is then allowed to act on the database. For example, suppose a manager wishes to decrease Brown's salary by 10%. This would consist of the manager entering the following query:

```
RANGE OF E IS EMPLOYEE
REPLACE E (SALARY = .9 X E.SALARY0
WHERE E.NAME = "BROWN"
```

Now the integrity constraint mentioned above disallowing salaries under $8,000 is appended to the query. This modified query is:

```
RANGE OF E IS EMPLOYEE
REPLACE E (SALARY = .9 X E.SALARY)
WHERE E.NAME = "BROWN"
AND .9 X E.SALARY  $8,000
```

This ensures that Brown's updated salary will be more than $8,000.

The INGRES database also contains an administration file. This file contains the user-id of the DBA and initialization information.

Due to the centralized DBA concept of INGRES, the database does not lend itself to internal decentralization of authorization. There is no legal way to propagate authority to other users. THe DBA is the sole authorizer of the database.

In a distributed database system the concept of centralized authorization control uncovers a number of disadvantages. If the system dictionary is not stored redundantly at several of the other nodes comprising the system, there is the danger that this data could either be lost or would have to be reconstructed from an archive version of the dictionary if the files comprising the dictionary were corrupted, lost, or otherwise rendered unreliable. Another disadvantage is that if each node did not have a copy or a part of the system dictionary, all access requests would have to go through the central site which would cause unnecessary message traffic to flow through the network. With numerous access requests flowing through the network from all directions to the central facility, there would be a considerable time lag in the confirmation or rejection of the requests due to the backlog of authorization requests awaiting processing. The

time wasted in processing the authorization requests could have been employed in processing transactions. The centralized authorization approach to security through the utilization of the concept of authorization through sole central site takes away the effectiveness and speed of processing offered by a DDS. Another problem is that if there is a failure at the central site, the other nodes cannot have their authorization requests fulfilled, which signifies that their operations are hampered since they cannot work on any of the transaction processing awaiting authorization confirmation. This results in a degradation of system performance due to the low throughput and high idle time.

4.0 Decentralized Authorization Central

In decentralized authorization control an individual or group of individuals could be designated as the authorizers for one particular site in the network. Local database administrators may be appointed to oversee the operations at each respective local site. Each local database administrator will then be responsible directly to the chief DBA at the central organization. This allows for the implementation of authorization mechanisms that serve the goals of the local organization while meeting the overall objectives of the parent organization. In other words each individual site can implement authorization mechanisms that can be specifically tailored to its needs as long as they provide the same level and consistency of security as predefined in the organizational security policy.

The basis of the System R authorization mechanism is the concept of ownership. Each table in the database is controlled

by the creator of that table, where a table is either a physically stored relation or a virtual table providing a dynamic window into the database (view). If an owner wishes to share the table with other users, he may use the GRANT command of the SEQUEL language to give various privileges to other users. This command is in the form of:

ALL RIGHTS

GRANT <privileges ON <table TO PUBLIC [WITH GRANT OPTION]

ALL BUT <privileges <user

The privileges include:

- READ - the ability to retrieve data from the table
- INSERT - the ability to insert new tuples into the table
- DELETE - the ability to delete tuples from the table
- UPDATE - the ability to update tuples in the table
- DROP - the ability to delete an entire table from the database

The owner may grant all privileges or a subset of them to other users. He may also propagate his ownership of the table, or a subset of it, to other users. The propagation of ownership to another user permits the user to further grant access and control privileges to other users, thereby decentralizing authorization through propagation. This concept is best visualized as a directed graph, where each node is a user and directed edges are privileges which are granted from user to user. For example, the owner X of table R wishes to grant READ and UPDATE privileges to user Y along with the right to propagate his

allocated sphere of influence (READ and UPDATE of R). The SEQUEL command would be:

X: GRANT READ, UPDATE ON R TO Y WITH GRANT OPTION

The directed graph for this command is shown in Figure 1.

User Y may further grant access privileges or propagate ownership to user Z:

Y: GRANT READ ON R TO Z.

User Z may now read tuples from table R but cannot update nor does he possess the power to propagate his right acquired from Y (Figure 2).

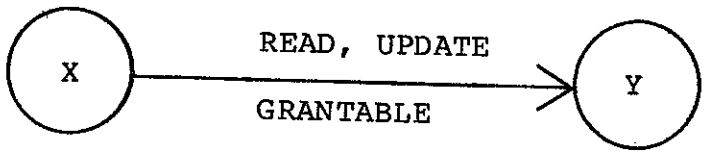

Figure 1. Directed Graph for the Grant Command by X

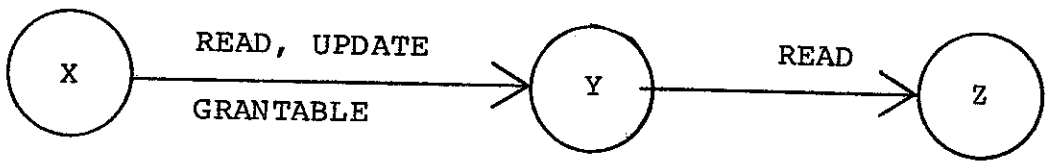

Figure 2. Directed Graph of Figure 1 with the Grant of Y

Any privileges granted through this process may also be revoked by the grantor. The SEQUEL command for this is:

REVOKE { ALL RIGHTS | <privileges> ON <table> FROM <user> }

Privileges on the names table are defined to the revokee, unless the revokee has another (independent) source of the same privilege.

The need to revoke a previously granted privilege significantly complicates the system R authorization mechanism. The act of revoking a previously granted access privilege revokes that privilege from the revokee. But when revoking a propagation right from a user, not only are the revokee's access and propagation revoked, but all rights propagated by the revokee are in turn revoked.

In summary, the system R authorization mechanism consists of:

1. a user creating a relation or view
2. the availability to grant access rights on his relation or view, or subset of either, to other users (e.g., READ, INSERT, DELETE, UPDATE, DROP)
3. the ability to grant the right of control of his relation, view, or subset of either to other users (propagation of ownership)
4. the ability to revoke items 2) and 3)
5. the ability to delete, or drop, a relation or view.

At this point, the algorithm would discover the circular propagation and revoke both X's and Y's rights as qualified by A's revocation.

A tool used in this algorithm is timestamping. Timestamping is effective in finding the difference between a user circulating propagation or obtaining it from an independent source. A timestamp would indicate the relative time of a grant. This

timestamp may represent real time or it may be a system maintained counter. The timestamp is monotonically increasing and ensures that no two GRANT commands are tagged with the same timestamp. Privileges granted in the same command are tagged with the same timestamp.

Another method of revocation is that of labeled grants [5] where a grant of a privilege is implicitly a grant of exactly one of the grantor's privileges; a grant is labeled by its source. For example, when a grantor (A) revokes privileges from a user (B) who has propagated those privileges to C, the label assigned by A to B and C is used as the revoking mechanism. This would be implemented as an extra column in the SYSAUTH table and associated with each grant.

When a user propagates some or all of his rights of control to other users, he is decentralizing the authorization of the database. Decentralization in System R is natural in that the DBMS fully accepts this condition. Similarly, System R promotes the distribution of authority by allowing users to grant access to information they "own." This schema provides dynamic control of all access and control privileges. This would allow users to create tables, share all or part of the table with other users, allow others to manage his table, and still retain as much security on the table as warranted.

For example, suppose a company, Database Designers Corporation, was awarded a contract to set up a database for the U. S. Post Office. This database would have to be accessible from 2 post offices in each state (100 offices overall). Seventy percent (70%) of the information stored would be local in nature

and would usually not have to be queried by any other post office, 15% of the information stored at each office is considered public knowledge and totally accessible and another 15% is postal employee salary information and is to be totally secure data, accessible to the local managers and nationwide auditors.

The physical distribution of this database could be any one of several:

1. Regional host processor and DBMS, say one for every 10 post offices with terminal hook-ups from one post office to the other 9, and communication links between each of the 10 regional centers, or
2. A host processor and DBMS at each of the 100 locations with communication links available to any of the other locations, or
3. A central location containing all processing services linked to the 100 locations via some communication link.

Whichever is implemented, authorization via System R can be decentralized to; a data processing manager at each office. This manager can then further propagate access and control to other users in the same office (to administer the 70% of local information), to other users in different office (to allow access to the 15% of sharable data), and can restrict access over payroll to a select few (local management and nationwide auditors). The advantages of this are that it:

Provides a flexible authorization method.

Decreases the scope of bottlenecks (going through a central DBA).

Disadvantages include:

- Greater understanding of the DBMS must be afforded to the many authorizers.
- To obtain privileges, a user must know from whom to receive them.
- Decentralization of the database is constrained by the use of relations and views, which might be an undesirable quality.
- Decentralization is inherent in the Wood-Fernandez [7] proposal. When the database has been physically distributed, it may be desirable to distribute the descriptive data (including authorization-related information) on a near nonredundant distribution to support the decentralized administration function. The descriptive data includes:

1. Object definition;
2. Class definitions;
3. Class structure graph;
4. Authorization rules.

Assume that each class is associated with one node in the network and that all descriptive information related to a class is stored at the associated node. Thus if class D is associated with node N the following information will be stored at N:

1. The definition of D;
2. The description of objection class D (if D is a basic class);
3. The children of D in the class structure graph;
4. The authorization rules associated with class D and objects and views defined in terms of D:
5. A directory that indicates the association between

classes and nodes.

Before issuing an access request to the database the users must specify in which authorization context they are operating. Although a requested data object may be a member of more than one class, it is uniquely associated with one class for the purposes of the current interaction.

At each node there is a replicated class-node directory (which typically will be small compared with the number of authorization rules). Validation of an administrative or access request requires the reading of the directory to locate the node where the relevant authorization rules are stored and passing the request to that node. If the rule corresponding to the request does not exist at that node the request is denied. Rules at other nodes need not be searched. The delegation of a class requires access to authorization rules at possibly two nodes. Recall of a delegated class requires access to the rules stored at the nodes associated with the classes in the class structure subgraph. However, recall is not likely to be a frequent occurrence. Authorization-related functions can therefore be performed with the minimum of internode messages.

5.0 Comments and Conclusions

The subject of database security presents a multitude of problems for those responsible for safeguarding the actual physical contents of the database, the software needed to run the database management system, applications programs that interact with the database contents, and the actual hardware on which the DBMS and related software operate. Questions involving the

physical security of the DBMS hardware, general computer system facility operational problems, operating system security, and the issuance of a general policy statement for defining the rules and regulations for the organization's entire computing facilities represent just a few of the security issues that must be addressed before, during, and after the implementation of a database management system. An organization must define what the term security signifies and implies to its particular operation. Security may only imply the enforcement of privacy regulations, or the non-disclosure of sensitive information, or perhaps the maintenance and assurance of both the integrity and reliability of the data stored within the confines of the physical database. The range and scope of the security mechanisms implemented in any database management system must therefore be expressed in terms of the criticality of the data with respect to the organization's overall operation.

This paper introduced three known authorization mechanisms and discussed their features, merits and drawbacks. The first mechanism, known as the authorization matrix, utilizes a table in which the rows identify authorized users and the columns correspond to the data objects that are to be controlled. THe second mechanism is based on the concept of views, restricting the scope and values that can be seen by a user through the use of an access predicate which implements the specification of privileges over objects. The third authorization mechanism consists of classification levels for data objects an clearance levels for users. It's application is most prevalent among military installations where huge amounts of information is handled and

processed constantly.

It was concluded that the authorization matrix above should not form the basis for an authorization system. The difficulties encountered in handling dynamic states of authorization in the system will outweigh the efficiency of this mechanism. Even if an authorization matrix contained global information over data at dispersed nodes that a user may have access to, the posting of updates would be complicated by the fact that changes over the entire distributed system may have to be taken into account.
It was also concluded that the view mechanism together with the Grant and Revoke scheme offered a great deal of flexibility in the general application of authorization control. Control of the implementation of a given authorization mechanism, in a distributed database system, can be centralized or decentralized. In centralized control the system dictionary contains all the information and data required for the authorization mechanism. Access to the dictionary and authorization matrix (if one exists) is restricted to the DBA and other security officials. In decentralized authorization control an individual or group of individuals are designated as the authorizers for one particular site in the network. System R and the Wood-Fernandez proposal support the decentralized administration function. When the database has been physically distributed it may be desirable to distribute the authorization related information. Furthermore, the implementation of an authorization mechanism should be accomplished by using system catalogs which have predefined names and are created for each database.

The issues and problems related to the security of database

systems are certainly not new. Considerable progress has been achieved in the area of authorization mechanism and control. However, with the advent of distributed database systems renewed interest exists in both evaluating the applicability of the existing authorization mechanisms as well as proposing better ones.

BIBLIOGRAPHY

[1] J. B. Rothnie, Jr., et al., "Introduction to a System for Distributed Databases (SDD-1)" ACM Transactions on Database Systems, vol. 5, NO. 1 (March 1980), p.1.

[2] M. M. Astranhan, et al. "System R: Relational Approach to Database Management." ACM Transactions on Database Systems Vol. 1, No. 2, (June, 1976), pp. 87-137.

[3] M. Stonebraker, E. Wong, and R. Kress "The design and implementation of INGRES." ACM Transactions on Database Systems, vol 1, No. 3, (September, 1976) pp. 189-222.

[4] H. Boral, et al "Implementation of the database machine DIRECT." IEEE TOSE, Vol. SE-8, No. 6 (November 1982).

[5] P. Griffiths an B. Wade "An authorization mechanism for a relational database system. ACM TODS, Vol. 1, No. 3 (September 1976) pp. 242-255.

[6] D. McLeod "A Framework for database protection and its application to the INGRES and System R DBMS." Proc. of First Int. Computer Software and Applications Conf. (COMPSAC 77), 342-348.

[7] C. Wood and E. B. Fernandex "Decentralized authorization in a database system" IBM Los Angeles Scientific Center Tech. Rep. G320-2698, (March 1979).