Fine-Grained Data Access for Networking Applications

Gábor Hosszú
Budapest University of Technology and Economics, Hungary

Harith Indraratne
Budapest University of Technology and Economics, Hungary

INTRODUCTION

Current-day network applications require much more secure data storages than anticipated before. With millions of anonymous users using same networking applications, security of data behind the applications have become a major concern of database developers and security experts. In most security incidents, the databases attached to the applications are targeted, and attacks have been made. Most of these applications require allowing data manipulation at several granular levels to the users accessing the applications—not just table and view level, but tuple level.

A database that supports *fine-grained access control* restricts the rows a user sees, based on his/her credentials. Generally, this restriction is enforced by a query modification mechanism automatically done at the database. This feature enables per-user data access within a single database, with the assurance of physical data separation. It is enabled by associating one or more security policies with tables, views, table columns, and table rows. Such a model is ideal for minimizing the complexity of the security enforcements in databases based on network applications. With *fine-grained access controls*, one can create fast, scalable, and secure network applications. Each application can be written to find the correct balance between performance and security, so that each data transaction is performed as quickly and safely as possible.

Today, the database vendors like Oracle 10g, and IBM DB2 provides commercial implementations of fine-grained access control methods, such as filtering rows, masking columns selectively based on the policy, and applying the policy only when certain columns are accessed. The behavior of the fine-grained access control model can also be increased through the use of multiple types of policies based on the nature of the application, making the feature applicable to multiple situations. Meanwhile, Microsoft SQL Server2005 has also come up with emerging features to control the access to databases using fine-grained access controls.

Fine-grained access control does not cover all the security issues related to Internet databases, but when implemented, it supports building secure databases rapidly and bringing down the complexity of security management issues.

BACKGROUND

Modern database applications with large numbers of users require *fine-grained access control* (FGAC) mechanisms at the level of individual tuples, not just entire relations/views, to control which parts of the data can be accessed by each user. Consider the following scenario:

In a commercial organization's human resources database, the human resources manager should have access to all the personal details of employees. At the same time, individual employees should only be able to see their particulars, not other employees' information.

In the above case, authorization is required at a very fine-grained level, such as at the level of individual tuples. Similar scenarios exist in many environments, including finance, law, government, and military applications. Consumer privacy requirements are yet another emerging driver for finer control of data.

Currently, general data authorization mechanisms in relational databases permit access control at the level of complete tables or columns, or on views. There is no direct way to specify fine-grained authorization to control, which tuples can be accessed by users. In theory, *FGAC,* at the level of individual tuples, can be achieved by creating an access control list for each tuple.

However, this approach is not scalable (Jain, 2004) and would be totally impractical in systems with millions of tuples and thousands or millions of users, since it would require millions of access control specifications to be provided (manually) by the administrator. It is also possible to create views for specific users, which allow those users access to only selected tuples of a table, but again, this approach is not scalable with large numbers of users.

In some occasions, *FGAC* is often enforced in the application code, which has numerous drawbacks; these can be avoided by specifying/enforcing access control at the database level. Current information systems typically bypass database access control facilities, and embed access control in the application program used to access the database. Although widely used, this approach has several disadvantages, such as access control has to be checked at each user-interface. This increases the overall code size. Any change in the access control policy requires changing a large amount of code. Further, all security policies have to be implemented into each of the applications built on top of this data. Also, given a large size of application code, it is possible to overlook loopholes that can be exploited to break through the security policies (e.g., improperly designed servlets). Also, it is easy for application programmers to create trap-doors with malicious intent, since it is impossible to check every line of code in a very large application (Rizvi, Mendelzon, Sudarshan, & Roy, 2004)

Fine-grained access control methods based on query modification approaches, such as *Oracle VPD,* have its own drawbacks. Specifically, implementing policies on improperly designed tables may result in inconsistent query results and unanticipated execution times. Proper database design and use of indexes for predicate values may overcome these drawbacks.

For the above reasons, fine-grained access control should ideally be specified and enforced at the database level. Today, both *Oracle 10g* and *SQL Server 2005* have captured the attention of the database community because of the new, exciting database features included in their latest releases (Gornshtein & Tamarkins, 2004). In this article, we present a *FGAC* security model for *SQL Server 2005* similar to the *FGAC* method implemented in *Oracle 10g* as *Oracle VPD*.

Fine-Grained Access Control in Oracle 10g

The model implemented by Oracle (Oracle, 2005) for fine-grained access controls is called *Virtual Private Database* (VPD) and restricts the rows a user sees based on his/her credentials (Loney, 2004). *Oracle Database 10g* VPD introduces column-relevant security policy enforcement and optional column masking. These features provide tremendous flexibility for meeting privacy requirements and other regulations (Needham & Iyer, 2003). As presented in *Figure 1, VPD* restriction is enforced by a WHERE clause automatically appended to the original query, based on the *application context* information gathered at the user log-on time. This clause, called a *predicate*, is generated by a user-defined function called *policy function*.

Figure 1. Oracle's virtual private database (VPD) feature