Kernelized Database Systems Security

Ramzi A. Haraty
Lebanese American University, Lebanon

INTRODUCTION

There are two main types of security in database systems: discretionary security and mandatory security. Discretionary security restricts access to data items at the discretion of the owner. Most commercial database management systems (DBMS) employ some form of discretionary security by controlling access privileges and modes of data users (Griffiths & Wade, 1976). Discretionary security is not adequate in a multilevel secure environment, however, because it does not prevent Trojan horse attacks and provides a low level of assurance. Mandatory security, on the other hand, restricts access of data items to cleared database users. It is widely exploited in military applications and provides a high assurance.

Numerous commercial and military applications require a multilevel secure database management system (MLS/DBMS). In a MLS/DBMS, database users are assigned classification levels and data items are assigned sensitivity levels. Usually, three architectures are used to provide security in MLS/DBMS. These architectures are the integrity lock or spray paint architecture, the data distribution architecture, and the kernelized architecture. The integrity lock and the data distribution architectures are outside the scope of this work. We focus only on the kernelized architecture.

In the kernelized architecture, data are physically stored in separate databases or containers according to sensitivity level. A multilevel relation is thus divided into single-level relations, each stored in a separate container. All containers are under the control of the common DBMS. The security of the system is largely dependent on the security of the operating system. Hence, the DBMS security is only as good as the underlying operating system. However, it is the responsibility of the MLS/DBMS to ensure that users can access only those data items for which they have been granted clearance.

The advantages of this architecture are that it is fairly secure and the kernel needed to implement it is relatively small. However, there are also several disadvantages. One major disadvantage is performance overhead associated with managing and executing multilevel transactions. In this article, we present an efficient model for concurrency control in kernelized databases. We show that the model is correct, secure, and provides a solution to the concurrency control problem.

BACKGROUND

The standard military security approach is followed, which consists of two components: a set of security classes and a set of nonhierarchical compartments. The security classes are totally ordered from the lowest to the highest as follows: unclassified << confidential << secret << top secret. Within each security class there can be zero or more compartments (i.e., conventional, chemical, and nuclear).

A security class, S1, is dominated by another class, S2 if S2 is hierarchically higher than S1 and contains all of its components.

In security terms, users or the processes that execute on behalf of them are referred to as subjects. Users are trusted, but processes are not. Objects, on the other hand, correspond to data items. Objects can be files, records, or even fields.

The Bell-LaPadula model introduced two security policies commonly accepted in a system that enforces multilevel security (Bell & LaPadula, 1976):

1. **Simple security policy**: A subject is allowed read access to an object if the subject’s classification level is identical or higher than that of the object’s sensitivity level.
2. **-policy**: A subject is allowed write access to an object if the subject’s classification level is identical or lower than that of the object’s sensitivity level.

These restrictions guarantee that proper access to objects will not be violated directly. However, they are insufficient to guarantee that proper access to objects is not violated indirectly through covert channels. Covert channels are channels, which are not intended to route information, but nevertheless do (Department of Defense, 1985). There are two main types of covert channels: covert storage channels and covert timing channels. Covert storage channels disclose information through the manipulation of a physical object, which either can or cannot be seen by low subjects. Covert timing channels can covertly send information from high to low subjects by modulating observable delay in the accessing of a common resource. A system is free from covert channels is called covert channel secure (CCS).