Chapter 11

SECURE REAL-TIME TRANSACTION PROCESSING

Jayant R. Haritsa  
Database Systems Lab, Supercomputer Education and Research Centre  
Indian Institute of Science, Bangalore 560012, INDIA.  
haritsa@dsl.serc.iisc.ernet.in

Binto George  
Dept. of Computer Science  
Western Illinois University, Macomb, IL 61455, USA.  
b-george@wiu.edu

1. INTRODUCTION

Many RTDBS applications arise in electronic financial services, safety-critical installations and military systems where enforcing security is crucial to the success of the enterprise. For example, consider the environment of an electronic (open bid) auction on the World-Wide-Web with online auctioneers and bidders. Typically, the auction database contains “secret” information such as bidders personal details including private keys, credit-worthiness and past bidding patterns; the purchase price and ownership history of the items that are being auctioned; the list of “in-house bidders” – these are bidders planted by the auction house to provoke other bidders by artificially hiking the maximum bid; etc. The database also contains “public” information such as bidder public keys and authentication certificates; the starting bid price, the minimum bid increment and the time for delivery for each item; the sequence and state of bids for items currently under auction; etc. It is expected that the secret information is known only to the auctioneers whereas the public information is available to both bidders and auctioneers.

In the above environment, the auction service provider faces a problem of three dimensional complexity: (1) There is a considerable amount of data to be consulted, processed and updated, and while doing so the database
consistency should be maintained; (2) There are time constraints associated with various operations – for example, a bid is valid only if registered in the database within a pre-specified time period after submission (in the Flash Auction at http://www.firstauction.com, bids that arrive more than five minutes after the previous bid is registered are invalidated); (3) During every stage of the bidding process, data security must be ensured — unauthorized access to the secret information by bidders may help them gain unfair and financially lucrative advantages over other competitors.

We investigate here the design of information systems that can simultaneously and effectively meet the above three challenges, that is, on the design of Secure Real-Time Database Systems (SRTDBS). In particular, we focus on the development and evaluation of high-performance secure real-time concurrency control (CC) protocols.

A number of requirements have been identified in the literature for a DBMS to be considered secure [6]. We restrict our attention here to secrecy, that is, the prevention of unauthorized knowledge of secret data, which is an especially important requirement for RTDBS due to the sensitive nature of their application domains. More precisely, we wish to enforce multilevel secrecy wherein, given a set of transaction security classes, the data associated with a particular class should not be revealed to transactions of lower security classes. In the sequel, we use the term security synonymously with multilevel secrecy.

Our study is carried out in the context of real-time applications with “firm-deadlines” [18] – for such applications, completing a transaction after its has expired is of no utility and may even be harmful. Therefore, transactions that miss their deadlines are considered to be worthless and are immediately “killed” – that is, aborted and permanently discarded from the RTDBS without being executed to completion. Accordingly, the performance metric is KillPercent, the steady-state percentage of killed transactions.¹

2. SECURITY MECHANISMS

Most secure DBMS attempt to achieve the secrecy objective by incorporating access control mechanisms based on the well-known Bell–LaPadula model [22]. This model is specified in terms of subjects and objects. An object is a data item, whereas a subject is a process that requests access to an object. Each object in the system has a classification level (e.g., Secret, Classified, Public, etc.) based on the security requirement. Similarly, each subject has a corresponding clearance level based on the degree to which it is trusted by the system.

The Bell–LaPadula model imposes two restrictions on all data accesses:

¹Or, equivalently, the percentage of missed deadlines.