Security Issues in Distributed Transaction Processing Systems

R. A. Haraty
Lebanese American University, Lebanon

INTRODUCTION

Transaction-processing systems (TPS) are becoming increasingly more available as commercial products. However, the approaches to the issues associated with using TPS in multilevel secure environments are still in the research stage. In this article, we address the issues of multilevel security in distributed transaction-processing systems. A distributed transaction-processing system (DTPS) is a collection of a finite number of centralized transaction-processing systems connected by a computer network. Each of these transaction-processing systems is controlled by a software layer and can be accessed both remotely and locally. Properties of a DTPS, such as data replication, may have a substantial effect on the security of the system. The security policies and integrity constraints adopted at each site may result in global security having inconsistent states. We address the issues of achieving a multilevel secure DTPS, and discuss the security constraints and data replication.

In this work, we address the issues of achieving a multilevel secure DTPSs system and discuss the security constraints and the replication of data items. The next section provides some background. Then, next, an overview of a distributed transaction-processing system is presented. In the fourth section, security-related issues are discussed. In the fifth section, a multilevel secure distributed transaction-processing system is presented. Then, in the next section, future trends are presented. The final section concludes the article.

BACKGROUND

Several commercial and military applications require a multilevel secure transaction-processing system (MLS/TPS). In an MLS/TPS, users are assigned classification levels that we denote by "clearances," and data items are assigned sensitivity levels. There are three interesting architectures that have been used to build MLS/TPSs from untrusted ones. These architectures are known as the integrity lock architecture, the kernelized architecture, and the data distribution architecture (Air Force Studies Board, 1983). While most of the techniques for TPS security are developed for traditional centralized TPSs, more TPS researchers are making substantial contributions to the development of a distributed TPS (Getta, 2003; Haraty, 1999; Haraty & Rahal, 2002; O'Connor & Gray, 1988).

A DTPS is a collection of a finite number of TPSs connected by a computer network (Ozsu & Valduriez, 1999). Each of these TPSs is controlled by a transaction management software layer and can be accessed both remotely and locally. A DTPS integrates information from the local TPS and presents remote users with transparent methods to use the total information in the system. An effective TPS system serves to maintain the ACIDity properties (i.e., atomicity, consistency, isolation, and durability) of transactions and must be superimposed on the preexisting local TPSs (Gray & Reuter, 1993).

One proposed architecture for MLS/TPS is the replicated architecture. This approach is being explored in several ongoing research efforts, including the Naval Research Laboratory Secure Information through replicated architecture (SINTRA) project (Thuraisingham, 1987). Data replication in DTPS has several implications for the security of the system. Replication allows data items in different local TPSs to be identified as logically belonging to the same entity. The security policies adopted by each site may result in global security having inconsistent states, because of the difference of local representation and management.

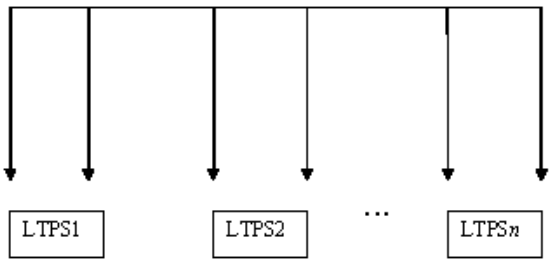
OVERVIEW OF DISTRIBUTED TRANSACTION-PROCESSING SYSTEMS

A DTPS consists of a set of preexisting local TPSs {LTPS*i* | 1 d•*i* d" *m*}, distributed among several interconnected sites. Each LTPS*i* is a software layer on a set of data items D*i*. Figure 1 depicts the architecture of a DTPS.

SECURITY ISSUES

Processes that execute on behalf of users are referred to as subjects. Objects, on the other hand, correspond to a data item. Objects can be files, records, or even fields. In this section, we present the notion of object classification with emphasis on the problem of conflicting security constraints due to replication.

Figure 1. Distributed transaction-processing system



A security classification is a function that associates each subject and each object with a given level of security. Many classifications, such as the security lattice, exist (Denning, 1976). However, a well-known classification is four-value function (DOD paradigm) that classifies objects into unclassified (U), confidential (C), secret (S), and adopt top secret (TS). A simple policy that can be established using a classification function SL is as follows:

Subject X can access (read) Object Y iff SL (Y) ≤ SL(X)

A security constraint consists of a data specification and a security value. The data specification defines any subset of the TPS. The security values can be given by a classification function. Specific values are unclassified, confidential, secret, and top-secret. Thuraisingham (1987) defined two types of security constraints—internal constraints and external constraints:

1. Internal constraints classify the entire TPS as well as relations, attributes, and tuples within a relation. These constraints can be applied to data, as they are actually stored in the TPS.
2. External constraints classify relationships between data and the results obtained by applying operations on the stored data, such as sum, average, and count. Among these constraints are the functional constraints and the dynamic constraints.

These security constraints are subject to inconsistency and conflicting local security constraints. A good global security approach should reject inconsistent security constraints and inconsistent clearance of users. Examples of the inconsistencies encountered include:

- **Conflicting security constraints:** Such constraints classify the same facts into different categories.
- **Overlapped security constraints:** These constraints cover overlapped data domains.
- **Inconsistent security level of replicated data:** Cases where different copies of replicated data may belong to different security cases.
- **Access privileges of users to replicated data:** Instances where a user may have different access rights on replicated data at different sites.

Several solutions have been proposed to solve these inconsistencies and define a global security policy that respects the local ones (Pfleeger, 1989; Thuraisingham, 1987).

There are several ways to combine local policies. The optimal combination should give a policy that defines all component policies and is still secure.

MULTILEVEL SECURE DISTRIBUTED TRANSACTION-PROCESSING SYSTEMS

There are two strategies for building MLS/DTPS from DTPS. These strategies include data replication and per-level-based distribution. The scope of this article does not include the issues associated with network security; but, it is particularly important to have the various local TPSs. Instead, we will assume that interconnection between the various local TPSs is secure and focus attention on security that has to be provided due to replication and other properties specific to the TPS.

The data distribution approach physically replicates low-level data at all higher-level TPSs. The advantage of the replicated architecture is that is fairly secure (McDermott & Sandhu, 1991). No performance overhead is associated with multilevel queries, because they are locally executed. On the other hand, because data is replicated, there is overhead associated with broadcasting updates of lower-level data to higher-level TPSs in a correct and secure manner. This broadcasting mechanism is known as "data synchronization" (Air Force Studies Board, 1983).

In the per-level-based approach, data are physically stored in separate local TPSs according to sensitivity level. Early examples of this approach were presented by Hinke and Schaefer (1975). The advantage of this approach is that updating transactions does not produce inconsistencies. Performance overhead associated with multilevel queries is a major disadvantage.

Global Commitment in Secure Environment

An important aspect of a correct TPS is atomic commitment (Bernstein et al., 1987). Unfortunately, the local TPS in a MLS/DTPS system cannot support atomic commitment, so the two-phase commit (2PC) protocol (Bernstein et al., 1987) cannot be implemented. 2PC is known to introduce covert channels. In order to establish a covert channel, there must be two cooperating agents/subjects in the system and an encod-