Chapter 7.12
Modeling Fault Tolerant and Secure Mobile Agent Execution in Distributed Systems

H. Hamidi
Iran University of Science & Technology, Iran-Tehran

K. Mohammadi
Iran University of Science & Technology, Iran-Tehran

INTRODUCTION

A mobile agent is a software program which migrates from a site to another site to perform tasks assigned by a user. For the mobile agent system to support agents in various application areas, the issues regarding reliable agent execution, as well as compatibility between two different agent systems or secure agent migration, have been considered. Some of the proposed schemes are either replicating the agents (Hamidi & Mohammadi, 2005) or check-pointing the agents (Park, Byun, Kim, & Yeom, 2002; Pleisch & Schiper, 2001;) For a single agent environment without considering inter-agent communication, the performance of the replication scheme and the check-pointing scheme is compared in Park et al. (2002) and Silva, Batista, and Silva (2000). In the area of mobile agents, only few works can be found relating to fault tolerance. Most of them refer to special agent systems or cover only some special aspects relating to mobile agents, such as the communication subsystem. Nevertheless, most people working with mobile agents consider fault tolerance to be an important issue (Izatt, Chan, & Brecht, 1999; Shiraishi, Enokido, & Takzawa, 2003). Mobile agents are becoming a major trend for designing distributed systems and applications in the last few years and foreseeable future. It can bring benefits such as reduced network load and overcoming of network latency (Chan, Won, & Lyu, 1993). Nevertheless, security is one of the limiting factors of the development of these systems. The main unsolved security problem lies in the possible existence of malicious hosts that can manipulate the execution and data of agents (Defago, Schiper, & Sergent, 1998). Most distributed applications we see today are deploying the *client/server paradigm.* There are certain problems with the client/server paradigm, such as the requirement of a high network bandwidth, and continuous user-computer interactivity.

In view of the deficiencies of the client/server paradigm, the *mobile code paradigm* has been developed as an alternative approach for distributed application design. In the client/server paradigm, programs cannot move across different machines and must run on the machines they reside on. The mobile code paradigm, on the other hand, allows programs to be transferred among and executed on different computers. By allowing code to move between hosts, programs can interact on the same computer instead of over the network. Therefore, communication cost can be reduced. Besides, *mobile agent* (Fischer, Lynch, & Paterson, 1983) programs can be designed to work on behalf of users autonomously. This autonomy allows users to delegate their tasks to the mobile agents, and not to stay continuously in front of the computer terminal. The promises of the mobile code paradigm bring about active research in its realization. Most researchers, however, agree that security concerns are a hurdle (Greenberg, Byington, & Harper, 1998).

In this article, we investigate these concerns. First, we review some of the foundation materials of the mobile code paradigm. We elaborate Ghezzi and Vigna's classification of mobile code paradigms (Ghezzi & Vigna, 1997), which is a collection of the *remote evaluation, code on demand,* and *mobile agent* approaches. In the next section, we address the current status of mobile code security. The following section presents the model for fault-tolerant mobile agent. In the next section, security issues of the mobile agent are discussed, and we discuss security modeling and evaluation for the mobile agent in the section after. In the following section, simulation results and influence of the size of agent are discussed. We then conclude the article.

THE MOBILE CODE PARADIGM

The mobile code paradigm is essentially a collective term, applicable wherever there is mobility of code. While different classes of code mobility have been identified, Ghezzi and Vigna proposed three of them, namely *remote evaluation, code on demand,* and *mobile agent* (1997). This classification, together with the client/server paradigm, is summarized in Table 1.

In particular, the "know-how" in Table 1 represents the code that is to be executed for the specific task. In the mobile code paradigms (*remote evaluation*, *code on demand*, and *mobile agent*), the *know-how* moves from one side to another side regarding where the computation takes place; while in the client/server paradigm, the *know-how* is stationary on the remote (server) side. *Resources* are the input and output for the code, whereas *processor* is the abstract machine that carries out and holds the state of the computation. The arrows represent the directions in which the specific item should move before the required task is carried out. Ghezzi and Vigna's classification is found to be comprehensive and representative of most existing mobile code paradigms (such as the rsh utility, Java applets and mobile agent systems), and we will base our discussion on this classification.

SECURITY CONCERNS OF MOBILE CODE PARADIGMS

In this section, we discuss some possible security attacks to different mobile code paradigms, and possible mechanisms against these attacks.

Security Attacks

A security attack is an action that compromises the security requirements of an application. Applications developed using different paradigms are