

Chapter 1

Introduction

1.1 Background

Computer systems have evolved from early centralized computing systems into distributed computing systems. Distributed computing paradigms have also experienced a development from client-server paradigm to current mobile agent paradigm. Traditionally, *Remote Procedure Call (RPC)* is used in client-server paradigm. A client sends data to its server to invoke the execution of a remote static procedure on the server side and receives the results from the server after the computation is finished. RPC normally requires great amounts of data to be transferred across the network. Although the rapid growth of the Internet, especially the World Wide Web, provides an astounding amount of interconnected computing resources, most users' access to Internet resources are primarily restricted by the available network bandwidth. Compared to the speed of CPUs, the speed of network is still far behind. This fact stimulates seeking alternatives for moving data across a network to improve the computing efficiency.

The *remote evaluation (REV)* paradigm extended the client-server concept of distributed applications by transporting program code from the local client computer to the remote server computer for execution there. REV paradigm provided a new degree of flexibility in the design of distributed systems. In distributed systems using RPC, server computers are designed to offer a fixed set of services. On the contrary, in distributed systems using REV, server computers are more properly viewed as programmable processors. Because applications are usually more compact than the data they operate upon or produce [103], REV paradigm can reduce a great amount of communication that is required to accomplish a given task.

The *code on demand (CoD)* paradigm further extended the aforementioned concept, by transferring the application logic in the reverse direction (i.e. from the remote to the local computer) and providing the computational platform and resources locally to ascertain some result. The main form of application logic in CoD paradigm is “applet”. An applet is usually downloaded from a host computer as a server to a local computer as a client by using a web browser and gets executed on the local computer. The mobility of code for an applet is unidirectional and often occurs only once for one execution, although the same applet can be downloaded many times by the same computer or different computers at different times.

The *mobile agent (MA)* paradigm is a further extension to distributed computing paradigms. A mobile agent is a software program with mobility which can be sent out from a computer into a network and roam among the computer nodes in the network. It can be executed on those computers to finish its task on behalf of its owner. The direction of a mobile agent can be multi-way, from any computer to another within that network. The migration of a mobile agent can occur multiple times before it comes back to its home computer with the computation results. In addition, not only the application logic of a mobile agent is transferred between computers, but the application state can be transferred from one computer to another. The transferring of a mobile agent state facilitates it in working autonomously to travel between one or more remote computers. Figure 1 illustrates the distributed computing paradigms introduced above.

The use of mobile code has a long history dating back to the use of remote job entry systems in the 1960's. Today's mobile agent incarnations can be characterized in a number of ways ranging from simple distributed objects to highly organized software with embedded intelligence [64]. Mobile agent paradigm helps form a large-scale, loosely-coupled distributed system. Because of its many salient merits, it has attracted tremendous attention in the last few years and become a promising direction in distributed computing and processing, as well as high performance network area.

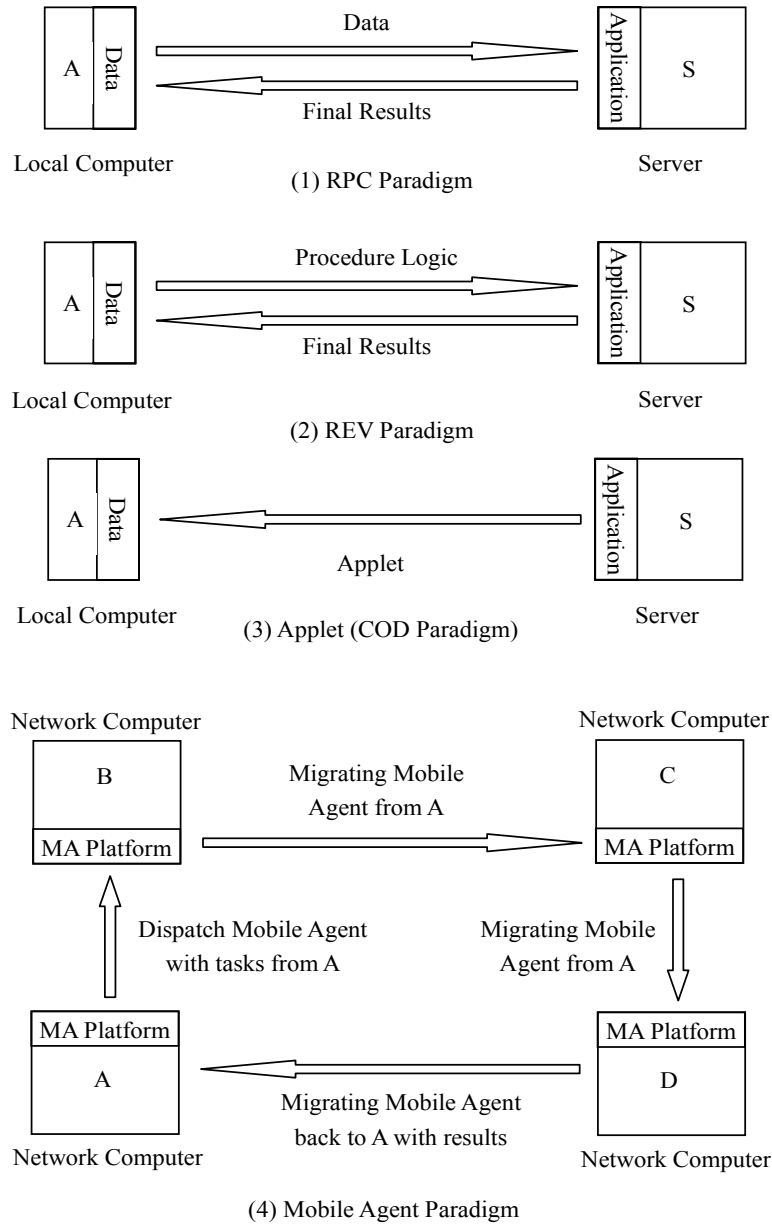


Figure 1 Evolution of Distributed Computing Paradigms

Although mobile agent technology has many notable advantages to be applied to a wide range, it also brings significant new security threats because the mobile code generated by one party will transfer to and execute in an environment controlled by another party. Several security issues arise in various areas for mobile agent computing, including authentication, authorization (or access control), intrusion detection, etc. A mobile agent system could be attacked by malicious agents, platforms and third parties. Many of these threats have counterparts in conventional client-server systems and have always existed in some form in the past. Mobile agents simply offer a greater opportunity for abuse and misuse, which broadens the scale of threats significantly. In addition, since mobile agents have some unique characteristics such as the mobility of a mobile agent, security problems become more complicated in mobile agent systems. Those security problems have become the bottleneck of the development and maintenance of mobile agent technology, especially in security sensitive applications such as electronic commerce.

A lot of research has been dedicated to address the security problems in a mobile agent system. This research differs in its aim, emphasis, base, and technique. Some work are towards building the foundations for the security of a mobile agent system; some propose security mechanisms following different approaches; some focus on introducing security mechanisms into the architectures of mobile code systems; and others implement real applications with security features. However, there has been limited research dedicated to provide an intuitive formal framework for a secure mobile agent system, including formal modeling of mobility, communication, and execution. This book introduces the concept and structure of mobile agent systems, discusses various attacks and countermeasures, and presents the security modeling and analysis of mobile agent systems. Our emphasis is on the formal modeling and analysis of a secure mobile agent system and its applications.