Mobile Agents and Security

Fei Xue
Monash University, Australia

INTRODUCTION

As an emerging technology, mobile agents can facilitate distributed computing applications over computer networks. During the past decade, the advance of computer software and hardware has led the structure and logic of mobile agents to become increasingly sophisticated. As a consequence, some security threats have started to appear in mobile agent systems (MASs).

Mobile agent technology derives from the concept of mobile code that Jeff Rulifson (n.d.) proposed in order to submit a set of program codes to a central machine in 1969. This concept was not transformed to mobile agents until 1994 when James E. White introduced Telescript, which is a runtime environment and programming language for mobile agents. One of Telescript's limitations is that it uses a proprietary approach lacking enough specifications, which thus has blocked its popularity (Minar, 1997). In 1995, a new mobile agent system named as MOLE was developed using Java, which is a computer language with the power of facilitating network environments (Baumann, Hohl, Rothermel, & Straßer, 1998). Since then, Java has become a common programming language for MAS, and this makes mobile agent technology more recognizable and acceptable to researchers and developers.

In 1997, the Object Management Group (OMG, n.d.) released a mobile agent standard entitled "Mobile Agent System Interoperability Facilities" (MASIF). This standard is known as one foundation of today's agent technologies. Since then, many mobile agent toolkits have been developed for the construct of MAS, such as Aglets, Tracy, JACK, Voyager, and so on (Braun & Rossak, 2005; Agentbuilder, n.d.). Today, mobile agents are suitable for many applications such as information searching and disseminating, distribution of the client-side software, semantic information retrieval, online service brokering, network management, and mobile computing.

BACKGROUND

A software agent is referred to as a software program that is authorized by its owner to work autonomously for the achievement of its objectives on behalf of its owner (Toivonen, 2000). When a software agent can migrate from one host to another over computer networks, it will be referred to as *mobile agent.* There exists a controversy about what properties can precisely characterize a mobile agent. The consensus is that a typical mobile agent must at least have the following key properties (Recursion Software, n.d.; Toivonen, 2000; Gupta et al., 2001, Chess et al., 1997):

- **Functionality:** A mobile agent must have the ability of functioning in a network environment to accomplish the tasks assigned by its user and to achieve its preset goals. During this functional process, the agent must be capable of reacting to any possible changes in its working environment. Preferably, a mobile agent should be able to communicate and collaborate with other agents and hosts.
- **Autonomy:** A mobile agent should act on its initiatives and perform its functions independently on behalf of its owner. For example, it can be unattended for a time period until a predefined event triggers its activation.
- **Mobility:** Mobile agents' mobility distinguishes them from any other software agents. This means that both code and data of a mobile agent can be moved within a network environment. The run-time execution of the agent can be suspended prior to its movement and can be resumed afterwards. A mobile agent can accomplish such movement by means of duplicating its run-time states. When it wants to move, it saves all states and transfers them to its destination host where its execution will be resumed according to those saved states.

- **Itinerary:** An itinerary-based MAS enables a mobile agent (referred to as *itinerant agent* in this case) to carry information about where it travels.

The power of a mobile agent can be further enhanced by some optional properties. A mobile agent may utilize artificial intelligence to get more smart features, such as learning new information, adjusting its behaviors, and deciding where to go and what to do. Security is another important property which requires that a mobile agent be designed to be unmalicious and immune to other agents and hosts. Although security is not a native property to mobile agents, it has become more and more important because of the increasing security concerns about mobile agents.

Mobile agents have some advantages and disadvantages (Harrison, Chess, & Kershenbaum, n.d.; Vigna, 1998; Lange & Oshoma, 1998; Recursion Software, 2001). On the one hand, a mobile agent can make computing systems work better in the following ways:

- Reduces network traffic by using fewer communication processes than traditional technologies like Remote Procedure Call (RPC) use.
- Its messaging processes are faster than those in traditional technologies.
- Supports autonomous query and interaction between hosts or between agents.
- Supports those who use mobile devices to perform functions over a network, even when their devices are off-line.
- Facilitates semantic information retrieval from different hosts, provided it is equipped with relevant intelligent algorithms.
- Provides lower overhead for secure transactions that apply encryptions than RPC-based systems.
- Its power can be enhanced by aggregation. The advantage of aggregate mobile agents is overwhelmingly stronger than what individual mobile agents can offer.

One the other hand, mobile agents have to overcome some problems. For example, transmission performance is an issue when a mobile agent carries a large volume of data, which results in a significant enlargement of its size. Compared with a small message transmitted over networks, its traveling time can be longer, and even may be intolerable in the circumstance that the agent is too large to be decomposed for online delivery. More seriously, security may be compromised in mobile agent-based systems. It is recognized that security attacks in MAS can take place in the following circumstances (Jansen, n.d.; Braun & Rossak, 2005; Jansen & Karygiannis, n.d.):

- **Agent-to-Agent:** Agent A can pretend to be an agent friendly to Agent B (known as masquerading), access the contents of Agent B without authorization, and even deny some services that Agent B provides.
- **Agent-to-Host:** An agent can pretend to be agent friendly to a host, access the resources on the host without authorization, and even deny the host's services.
- **Host-to-Agent:** A host can also pretend to be a host friendly to an agent, access its contents without an authorization, and even modify its code and logical structure.
- **Host-to-Host:** Host A can send a number of agents to collect and analyze the information about Host B. Host A can even change the code of the agents belonging to Host B when they travel to Host A.

Issues

One of the issues relating to mobile agents is that they may be abused. This means developers may sometimes use mobile agents to replace other traditional technologies in their computing systems, regardless of whether the agents can better suit their systems than those traditional technologies in terms of performance, ease of use, and security. They ignore the fact that mobile agents can be used only when their specific features are critically essential to the development or refinement of a computing system.

The increasing sophistication of MAS has motivated another issue—security concerns. Mobile agents can have a potential motivation to severely degrade and even destroy normal computing operations, when they are constructed with some malicious code or their logics include some bugs. Compared with the problems that ordinary communication technologies like RPC may cause, the destruction caused by that mobile agent may be worse, because its mobility capability can make this destruction more pervasive and its autonomy