Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
ELSEVIER Electronic Notes in Theoretical Computer Science 128 (2005) 3–16
www.elsevier.com/locate/entcs

Data Privacy in Tuple Space Based Mobile Agent Systems[1]

Lorenzo Bettini

Dipartimento di Sistemi e Informatica, Università di Firenze
Via Lombroso 6/17, 50134 Firenze, Italy
bettini@dsi.unifi.it

Abstract

More recently, distributed variants of tuple spaces have been proposed to exploit the Linda model for programming distributed applications over wide area networks, possibly exploiting code mobility. However, the flexibility of the shared tuple space model opens possible security holes; it basically provides no access protection to the shared data. In this paper we investigate some possible scenarios where mobile agents can benefit from our cryptographic tuple space based framework, CryptoKlava, and sketch how to possibly implement such agents in order to keep the privacy of items collected by the mobile agent during its itinerary. The functionalities of the framework are general enough to be applied to other Java frameworks using multiple distributed tuples spaces possibly dealing with code mobility.

Keywords: Mobile sgents, wide area networks, Linda, Java, mode mobilty

1 Introduction

The Internet has been cursed by intrusions and attacks since the early days [23,30,22]. Such intrusions used to exploit bugs in existing software in order to gain access to the computer, replicate themselves, and spread to other computers. Thus worms and viruses can rely on the concept of mobile code. Indeed, the high flexibility of mobile agents, and mobile code in general, do not come at no cost. Downloading code from the network for local execution

[1] This work has been partially supported by EU within the FET – Global Computing initiative project *MIKADO* IST-2001-32222. The funding bodies are not responsible for any use that might be made of the results presented here.

doi:10.1016/j.entcs.2004.11.038

exposes to possible threats at a higher level with respect to an isolated context [26,8].

Distributed and mobile code systems raise new security issues mainly because they "violate a number of assumptions that underlie most existing computer security measures" [11]. Assumptions that can be safely accepted for isolated computers are destined to fall when the executing scenario scales to an open network. Interesting features of mobile agents, such as, e.g., autonomy, also have drawbacks, since the owner of a system may ignore that remote code is executing on his machine.

A successful approach to concurrent programming is the one relying on the Linda coordination model [17]. Processes communicate by reading and writing *tuples* in a shared memory called *tuple space*. Control of accesses is guaranteed by requiring that tuples selection be *associative*, by means of pattern matching. The communication model is *asynchronous*, *anonymous*, and *generative*, i.e., tuple's life-time is independent of producer's life time.

The Linda model has been adopted in many communication frameworks such as, e.g., *JavaSpaces* [2] and *T Spaces* [16], and for adding the tuple space communication model to existing programming languages. More recently, distributed variants of tuple spaces have been proposed to exploit the Linda model for programming distributed applications over wide area networks [12,4], possibly exploiting code mobility [13,27]. As shown in [15], where several messaging models for mobile agents are examined, the *blackboard* approach, of which the tuple space model is a variant, is one of the most favorable and flexible.

Sharing data over a wide area network such as the Internet, calls for very strong security mechanisms. Computers and data are exposed to eavesdropping and manipulations. Dealing with these issues is even more important in the context of code mobility, where code or agents can be moved over the different sites of a net. Malicious agents could seriously damage hosts and compromise their integrity, and may tamper and brainwash other agents. On the other hand, malicious hosts may extract sensible data from agents, change their execution or modify their text [36,28].

The flexibility of the shared tuple space model opens possible security holes; it basically provides no access protection to the shared data. Indeed there is no way to determine the issuer of an operation to the tuple space and there is no way to protect data: a process may (even not intentionally) retrieve/erase data that do not belong to it and shared data can be easily modified and corrupted. In spite of this, within the Linda based approaches, very little attention has been devoted to protection and access control.

In [6] we presented CryptoKlava, a Java middleware for building dis-