Design, Implementation, and Experimentation on Mobile Agent Security for Electronic Commerce Applications[†]

Anthony H. W. Chan, Caris K. M. Wong, T. Y. Wong, and Michael R. Lyu
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N. T., Hong Kong

Abstract *In this paper, a Shopping Information Agent System (SIAS) is built based on mobile agent technology. It sends out agents to different hosts in an electronic marketplace. The agents collect and report information such as prices and availabilities about products specified by users. Snapshots of the system in use are shown. Security is a major problem of mobile agent systems, especially when money transactions are concerned. Possible security attacks by malicious hosts to agents in SIAS are discussed, and a solution to prevent these attacks is presented. Finally, security of the solution is analyzed, and the performance overhead introduced is evaluated.*

Keywords: mobile agents, security, information agents

1 Introduction

Mobile agents are becoming a major trend of distributed systems and applications in the coming years. It can bring benefits such as reduced network load and overcoming of network latency [1]. Nevertheless, security is one of the blocking factors of the development of these systems. The main unsolved security problem lies on the possible existence of malicious hosts that can manipulate the execution and data of agents [2].

In this paper, security issues of the mobile agent technology are discussed (Section 2). A Shopping Information Agent System (SIAS) is built using the Concordia [3] architecture (Section 3). The system can collect and compare the prices of a set of products specified by users from different seller hosts in an electronic market. Security issues of the system are addressed, possible attacks by malicious hosts to the system are described, and a solution to protect the system against these attacks is devised and implemented (Section 4). Finally, the solution devised is evaluated according to the security provided and the performance overhead introduced (Section 5).

2 Security Issues of the Mobile Agent Technology

Any distributed system is subject to security threats, so is a mobile agent system. Issues such as encryption, authorization, authentication, non-repudiation should be addressed in a mobile agent system. In addition, a secure mobile agent system must protect the hosts as well as the agents from being tampered by malicious parties.

First, hosts must be protected because they continuously receive agents and execute them. They may not be sure where an agent comes

[†] The work described in this paper was supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region (Project No. CUHK4432/99E).

from, and are at the risk of being damaged by malicious code or agents (Trojan horse attack). This problem can be effectively solved by strong authentication of the code sources, verification of code integrity, and limiting the access rights of incoming agents to local resources of hosts. This is mostly realized by the Java security model [4].

The main security challenge of mobile agent systems lies on the protection of agents. When an agent executes on a remote host, the host is likely to have access to all the data and code carried by the agent. If by chance a host is malicious and abuses the code or data of an agent, the privacy and secrecy of the agent and its owner would be at risk.

Seven types of attack by malicious hosts [2] can be identified:

- Spying out and manipulation of code;
- Spying out and manipulation of data;
- Spying out and manipulation of control flow;
- Incorrect execution of code;
- Masquerading of the host;
- Spying out and manipulation of interaction with other agents; and
- Returning wrong results of system calls to agents

There are a number of solutions proposed to protect agents against malicious hosts [5], which can be divided into three streams:

- Establishing a closed network: limiting the set of hosts among which agents travel, such that agents travel only to hosts that are trusted.
- Agent tampering detection: using specially designed state-appraisal functions to detect whether agent states have been changed maliciously during its travel.
- Agent tampering prevention: hiding from hosts the data possessed by agents and the functions to be computed by agents, by messing up code and data of agents, or using cryptographic techniques.

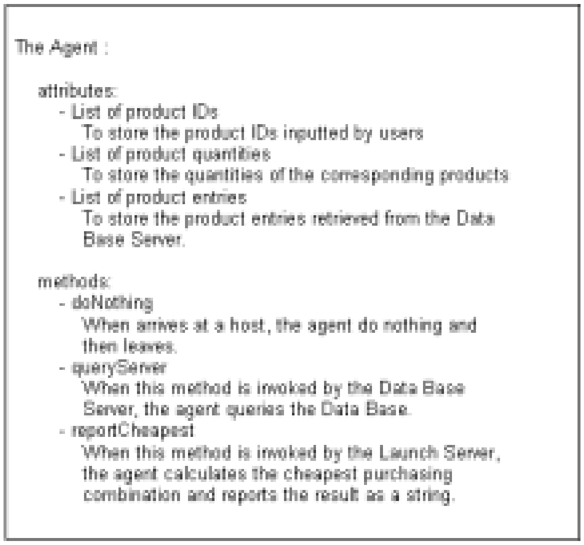
None of the proposed solutions solve the problem completely. They either limit the capabilities of mobile agents, or are not restrictive enough. A better solution is being sought, and there is no general methodology suggested to protect agents. In the mean time, developers of mobile agent systems have to develop their own methodologies according to their own needs.

Apart from attacks by malicious hosts, it is also possible that an agent attacks another agent. However, this problem, when compared with the problem of malicious hosts, is less important, because the actions of a (malicious) agent to another agent can be effectively monitored and controlled by the host on which the agent runs, if the host is not malicious.

3 Design and Implementation of SIAS

SIAS is a web-based mobile agent system that provides users with information of products for sale in an electronic marketplace. Advantages of SIAS include such properties as reduction of communication costs and delegation of tasks, which are the intrinsic advantages of a mobile agent system. It is written in the Java programming language and on top of the Concordia [3] application-programming interface (API).

The Concordia architecture, among different mobile agent platforms developed worldwide, is chosen for implementation of SIAS mainly because of the API simplicity, and the ability it allows to manipulate agent code execution, which is good for simulating malicious attack behaviors. The Java programming language is a natural choice for the Java-based Concordia



The Agent :

attributes:

- List of product IDs
  To store the product IDs inputted by users
- List of product quantities
  To store the quantities of the corresponding products
- List of product entries
  To store the product entries retrieved from the Data Base Server.

methods:

- doNothing
  When arrives at a host, the agent do nothing and then leaves.
- queryServer
  When this method is invoked by the Data Base Server, the agent queries the Data Base.
- reportCheapest
  When this method is invoked by the Launch Server, the agent calculates the cheapest purchasing combination and reports the result as a string.

Figure 1: Object details of Agent

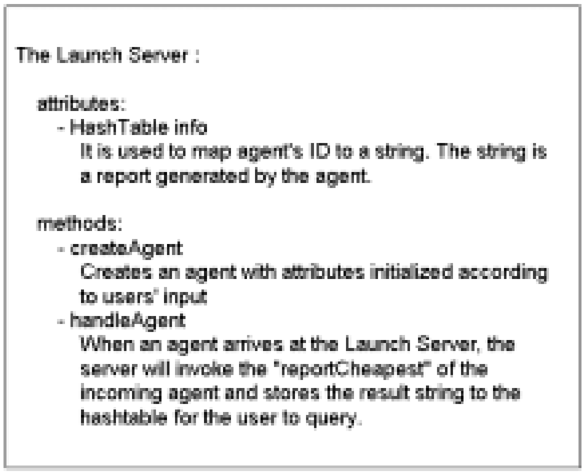
API.

SIAS implements mobile agents to retrieve product information in an electronic market for users. An electronic market consists of hosts that sell products on the network. Each seller maintains a database that stores the prices and quantities in stock of different products available at that host. Three objects, namely Agent, LaunchServer, and DatabaseServer, have been designed to implement the system. The objects details is shown in Figure 1, 2, and 3 respectively, and the control flow of the system is described by Figure 4.

A series of snapshots of SIAS in use is shown in Figure 5, 6 and 7. Figure 5 describes the user interface of SIAS. Figure 6 shows the moment when a user is choosing the products required, and Figure 7 shows the corresponding execution results.

4 Security Design of SIAS

Both host security and agent security [5] would be issues of SIAS. However, agent security is the primary interest of this paper. The Java sandbox model has largely simplified the host security part.



The Launch Server :

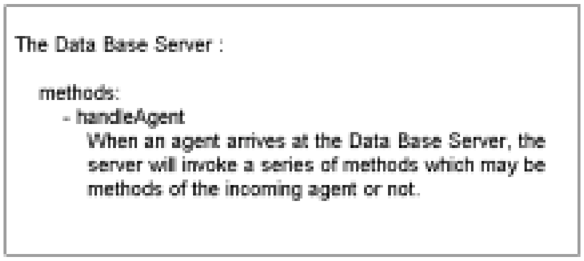
attributes:

- HashTable info
  It is used to map agent's ID to a string. The string is a report generated by the agent.

methods:

- createAgent
  Creates an agent with attributes initialized according to users' input
- handleAgent
  When an agent arrives at the Launch Server, the server will invoke the "reportCheapest" of the incoming agent and stores the result string to the hashtable for the user to query.

Figure 2: Object details of LaunchServer



The Data Base Server :

methods:

- handleAgent
  When an agent arrives at the Data Base Server, the server will invoke a series of methods which may be methods of the incoming agent or not.

Figure 3: Object details of DatabaseServer

Three particular security problems for SIAS can be identified:

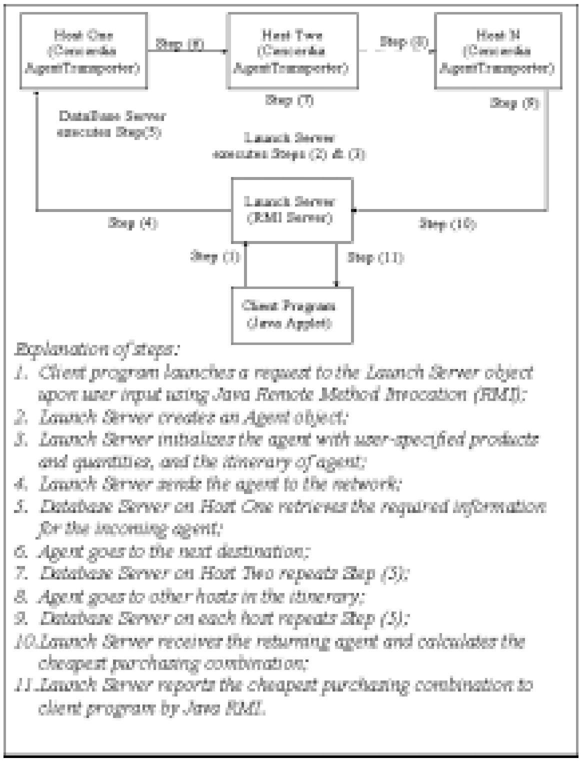
1. Modification of query products by a malicious host;

2. Modification of query quantities by a malicious host; and

3. Spying out and modification of query results.

This is only a subset of possible attacks. There are other attacks such as replaying of query results and masquerading of hosts. For simplicity, only the three attacks described above are considered.

Having figured out the above system vulnerabilities, a simple but original approach is



Explanation of steps:

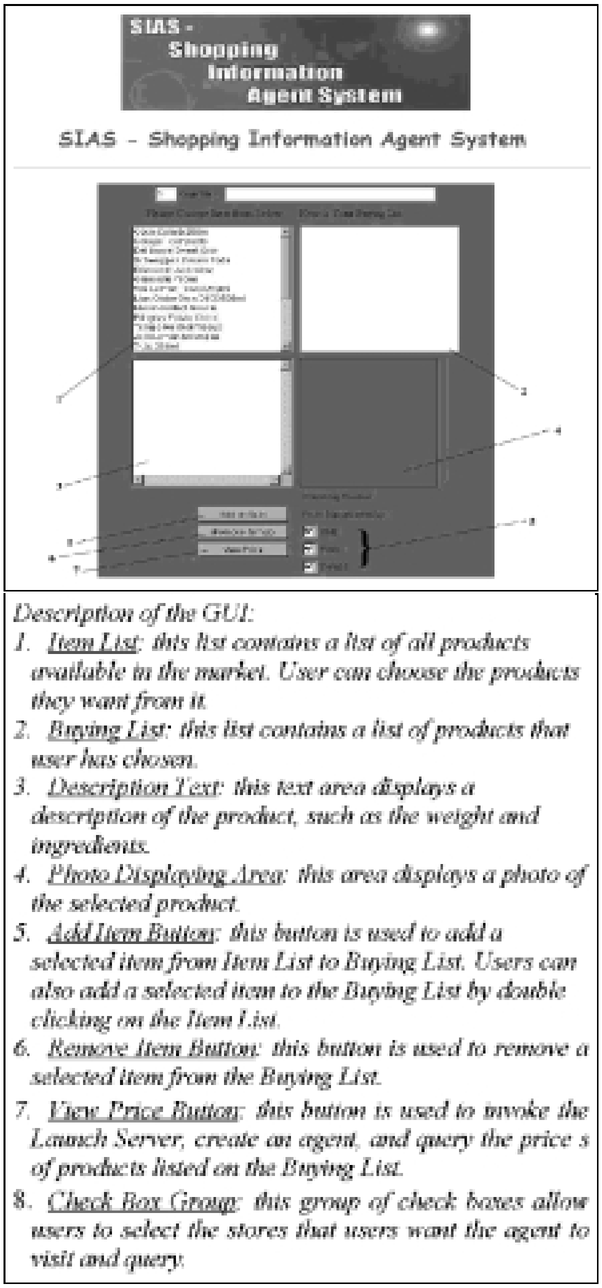
1. *Client program launches a request to the Launch Server object upon user input using Java Remote Method Invocation (RMI);*
2. *Launch Server creates an Agent object;*
3. *Launch Server initializes the agent with user-specified products and quantities, and the itinerary of agent;*
4. *Launch Server sends the agent to the network;*
5. *Database Server on Host One retrieves the required information for the incoming agent;*
6. *Agent goes to the next destination;*
7. *Database Server on Host Two repeats Step (5);*
8. *Agent goes to other hosts in the itinerary;*
9. *Database Server on each host repeats Step (5);*
10. *Launch Server receives the returning agent and calculates the cheapest purchasing combination;*
11. *Launch Server reports the cheapest purchasing combination to client program by Java RMI.*

Figure 4: Control flow of SIAS

developed to protect agents in SIAS against attacks from malicious hosts, based on cryptographic techniques. A public-key infrastructure [6] is introduced into the system. Each host and agent in the system is required to possess a pair of keys for encryption and decryption. Therefore, each agent or host can encrypt or digitally sign the data items carried by an agent, and thus all the a) query products, b) query quantities, c) query results can be protected. Figure 8 illustrates changes to the system for security enhancement.

5 Evaluation and Experimentation on the Secure SIAS

The security of the additional measures lies mainly on the introduction of a key server that facilitates the use of public key cryptography. Assuming the key server, the communication channel with the key server are secure enough, and the keys are managed properly, the pre-



Description of the GUI:

1. *Item List: this list contains a list of all products available in the market. User can choose the products they want from it.*
2. *Buying List: this list contains a list of products that user has chosen.*
3. *Description Text: this text area displays a description of the product, such as the weight and ingredients.*
4. *Photo Displaying Area: this area displays a photo of the selected product.*
5. *Add Item Button: this button is used to add a selected item from Item List to Buying List. Users can also add a selected item to the Buying List by double clicking on the Item List.*
6. *Remove Item Button: this button is used to remove a selected item from the Buying List.*
7. *View Price Button: this button is used to invoke the Launch Server, create an agent, and query the price s of products listed on the Buying List.*
8. *Check Box Group: this group of check boxes allow users to select the stores that users want the agent to visit and query.*

Figure 5: The User Interface of SIAS

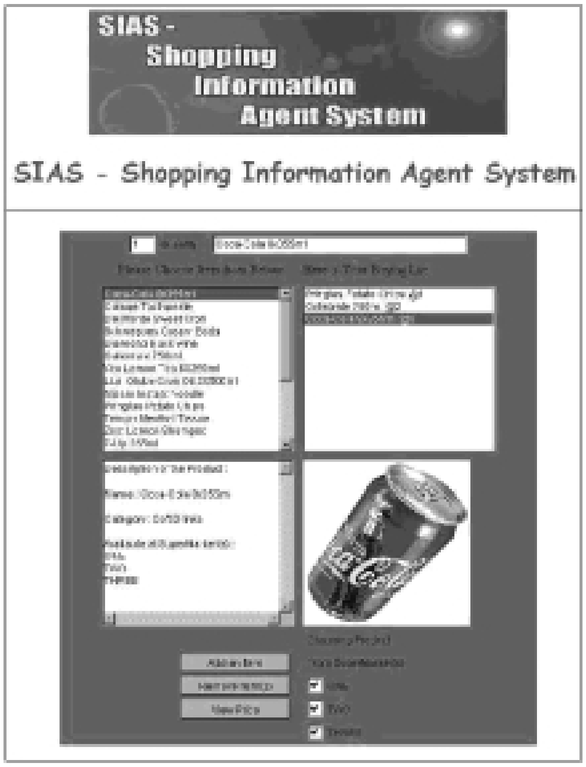
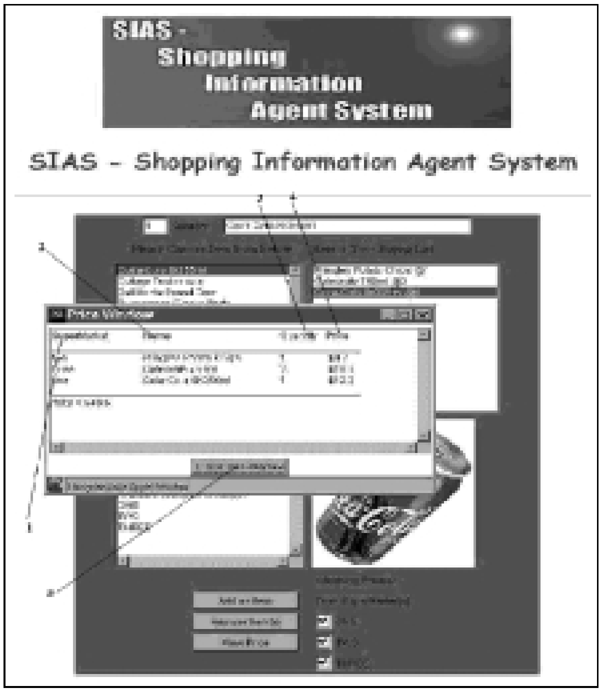


Figure 6: User choosing products from SIAS

vention of modification of the signed product and quantity lists of an agent by a malicious host is supported by the security of the RSA encryption algorithm. The time complexity for breaking the RSA cryptosystem depends on the length of the key in number of bits. The longer the key is, the more secure the system would be. In our implementation, we have chosen a key length of 128 bits. This would be sufficiently secure for our security purpose.



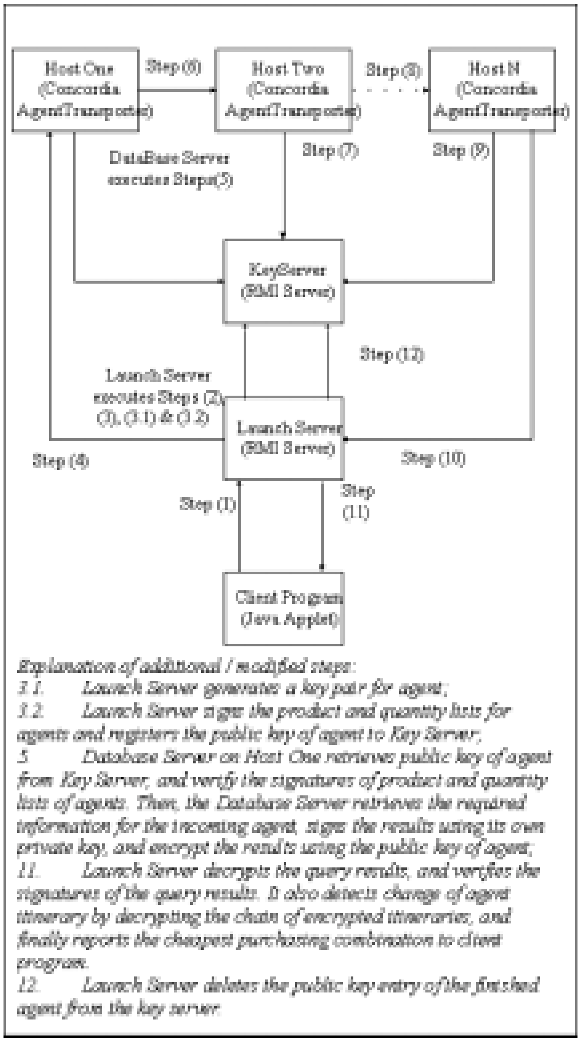
Description of report window:

1. *Supermarket column: this column displays, for each product, the store that is selling at the lowest price.*
2. *Name column: this column displays the name of each product.*
3. *Quantity column: this column displays the quantity of each product that users have specified.*
4. *Price column: this column displays the price of each product at the quantity specified by user.*
5. *Close Window Button: this button is used to close the report window.*

Figure 7: SIAS reporting query result to user

To evaluate the performance overhead introduced, the times for SIAS to launch a single agent have been tested with and without security measures. Round trip times (RTTs) required for an agent to travel around an electronic market of different number of hosts, with and without security enforcement, are measured respectively. Queries of different sizes (number of product items) have been tested. The results are plotted in Figures 9(a) (without security) and 9(b) (with security) below.

Results show that, the RTT for an agent to travel in SIAS changes more or less linearly over the number of hosts in the system. This is due to the additional time to travel an additional host, and the overhead for each additional host is more or less the same. Moreover, the RTT is also linearly increasing as the number of products of the query increases. This can be explained by the increases in number of database transactions and time to transport an agent. When security is enforced, the RTT



Explanation of additional / modified steps:

3.1. Launch Server generates a key pair for agent;

3.2. Launch Server signs the product and quantity lists for agents and registers the public key of agent to Key Server;

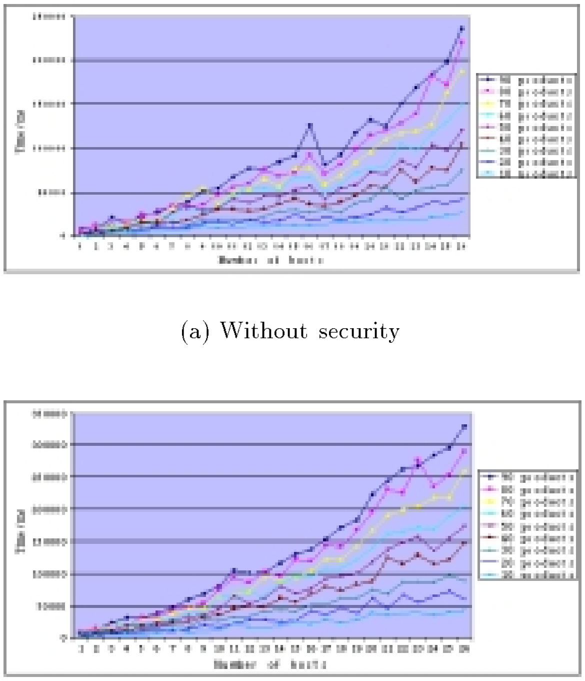
5. Database Server on Host One retrieves public key of agent from Key Server, and verify the signatures of product and quantity lists of agents. Then, the Database Server retrieves the required information for the incoming agent, signs the results using its own private key, and encrypt the results using the public key of agent;

11. Launch Server decrypts the query results, and verifies the signatures of the query results. It also detects change of agent itinerary by decrypting the chain of encrypted itineraries, and finally reports the cheapest purchasing combination to client program.

12. Launch Server deletes the public key entry of the finished agent from the key server.

Figure 8: Control flow of security-enhanced SIAS

increases in general. For the maximum number of hosts of 26, and maximum size of query of 90 products, the RTT increases by 100 seconds, from 230 seconds to 350 seconds. This can be explained by the extensive use of the RSA algorithm to encrypt and decrypt each item, which is time consuming, especially when the key is long. Therefore, trade-off between security and performance in SIAS is shown.



(a) Without security

(b) With security

Figure 9: Rount Trip Times of an agent, with different query sizes, against different numbers of hosts in SIAS

6 Conclusions

In this paper, the technology of autonomous mobile agents and the problem of malicious hosts in a mobile agent system are studied. SIAS is implemented as a sample application of mobile agents. Some of the security problems of malicious hosts in SIAS have been addressed, and a practical approach to protect the agents has been developed. The security of the approach developed has been analyzed, and is believed to be strong enough for the application. The performance overhead of the security measures are measured, and in conclusion, a trade-off between performance and security for SIAS is observed.

References

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley Publishing Company, 1986.

[2] Ree Source Person. A really great journal paper. *A really great journal*, 23(2):728–736, Oct 1976.

[3] Danny B. Lange and Mitsuru Oshima. "Seven Good Reasons for Mobile Agents". *Communications of the ACM*, p.88 - 89, 1999 Mar.

[4] F. Hohl. "A Model of Attacks of Malicious Hosts Against Mobile Agents". *Proceedings of the ECOOP Workshop on Distributed Object Security and 4th Workshop on Mobile Object Systems: Secure Internet Mobile Computations*, p. 105 - 120, INRIA, France, 1998.

[5] "Concordia - Java Mobile Agent Technology". http://www.meitca.com/HSL/Projects/ Concordia/

[6] "Java Security Architecture". http://java.sun.com/products//jdk/1.2/ docs/guide/security/security/spec/security-spec.doc1.html

[7] C. Tschudin. "Mobile Agent Security". *Intelligent Information Agents: Agent Based Information Discovery and Management in the Internet*, p. 431 - 446, Springer, 1999.

[8] R. Rivest, A. Shamir, and L. Adleman. "A Method for Obtaining Digital Signatures and Public Key Cryptosystems". *Communications of the ACM*, 1978 Feb.