Comparative Study on Approaches to Mobile Agents Security from Malicious Hosts in Mobile Adhoc Networks

Chethan B K[a], M Siddappa[b], H S Jayanna[c]
[a]Assistant Professor Department of ISE SSIT, Tumkur
[b]Prof and Head Department of CSE SSIT, Tumkur
[c]Prof and Head Department of ISE SIT, Tumkur

Abstract

A mobile agent is a composition of computer software and data which is able to migrate from one computer to another autonomously and continue its execution on the destination computer. Several advantages have been identified in using mobile agents in Mobile Ad-hoc Networks. Some of them include reduction of network load, decrease in communication latency, dynamic adaptation and better support for mobile devices with intermittent connections. However, the benefits offered by mobile agents have not been sufficient to stimulate their widespread deployment. The main reason why mobile agents have not been widely adopted yet, despite their technological benefits, is their inherent security risks. So there is a critical need for protection of mobile agents against malicious hosts in Mobile Adhoc Networks.

Software security to protect mobile agent consists of lots of aspects like cryptography, access control and trust management, intrusion detection and tamper resistance, authentication and privacy, signature schemes, e-commerce, security analysis, mobile computing security etc. So, to design and develop security mechanisms for mobile agents against malicious hosts this paper identifies different kinds of attacks and relationships between them. Security issues, measures and requirements are analyzed to focus on protecting mobile agents from malicious hosts in Mobile Adhoc Networks.

KEYWORDS: Network load, Communication latency, Cryptography, Intrusion detection, Access control, Trust management, Authentication, Signature schemes, Mobile computing, Code obfuscation.

1. INTRODUCTION

A mobile agent can be thought of as a software program, which can travel from one place to another. The migration of the whole running process, along with its state, code and its resources is what makes the mobile agents different from other kinds of distributed applications. Because of mobility of mobile agent, the security problems becomes more complicated and have become a bottleneck for development and maintenance of mobile agent technology especially in security sensitive applications such as e commerce, military applications, scientific applications etc[1]. Security issues are becoming more significant in this age of pervasive mobile network computing where we have different types of information being used by mobile and fixed large scale distributed applications interacting over wireless and wired network to deliver useful services to enterprises and users.

2. SECURITY OBJECTIVES

Mobile agent systems provide a computing infrastructure upon which mobile agents belonging to different and potentially untrusted users can execute. And also trusted agents are supposed to get executed in various unknown platforms by sharing their access of whole running process, along with its state, code and its resources. The communication medium can also be inherently insecure and the different agents and agent systems may have conflicting objectives. So any mobile agent based system environment should be able to achieve the following security objectives for the efficient and trusted working of the applications:

2.1 AUTHENTICATION

Authentication requires verifying the identity of a user, process or device before allowing access to resources in a system. Authentication requires that the identity of an entity or the originator of the data can be verified and assured to prevent it from faking or masquerading [2].

2.2 AUTHORIZATION

Authorization is to grant or to deny access rights to a user, program or process. This objective requires that only legitimate users have rights to use certain services or to access certain resources keeping unauthorized users out [3].

2.3 ACCOUNTABILITY

Accountability requires that users and administrators will be held accountable for behavior that impacts the security of information. Accountability is often an organizational policy requirements that directly supports non-repudiation, deterrence, fault-isolation, intrusion detection and prevention and after- action recovery and legal action [4].

2.4 ASSURANCE

Assurance grounds for confidence that other security goals (including integrity,

availability, confidentiality, and accountability) are adequately met [5].

2.5 AVAILABAILITY

Availability requires that data and system can be accessed by legitimate users within an appropriate period of time [6]. Some attacks like Denial of Service (DoS) or instability of the system can cause loss of availability.

2.6 CONFIENDIALITY

Confidentiality requires that data should be protected from any unauthorized disclosure i.e. data can only be accessed by persons or machines for which it is intended [7] A loss of confidentiality will not prevent data privacy.

2.7 INTEGRITY

Integrity can be divided into two aspects: Data integrity and System integrity. Data integrity is the objective that data should not be altered or destroyed in an unauthorized manner to maintain consistency. System integrity is the objective that a system should be free from unauthorized manipulation.

2.8 NON-REPUDIATION Non-Repudiation requires that either side of a communication cannot deny the communication later. To achieve this, important communication exchanges should be logged so as to prevent denials by any party of a transaction. It relies on authentication to record the identities of entities [8].

3. TYPES OF ATTACKS AND SECURITY ISSUES IN MOBILE AGENTS

Since the beginning of mobile agent research, many security issues have been identified. These issues are classified according to the source of the attack and the entity being attacked [9]:

i) Agents against Agents.
ii) Agents against Platforms.
iii) Others against platforms.
iv) **Platforms against Agents**.

In the first category—Agents against Agents—we can find attacks in which agents modify or access another agent's data, disguise their identity in order to falsify a transaction, or repeatedly send messages to another agent in order to launch a denial of service attack among others.
The second category—Agents against Platforms—includes threats in which agents perform some malicious action on a resource they can access to (e.g.,

deleting a file), consume an excessive amount of system resources, gain access to a service to which they are not entitled, and so on.
With regard to these two first categories, in which the attacker is an agent, sound solutions have already been proposed. Among the solutions that provide an acceptable level of protection, the most efficient one is called **Software-based Fault Isolation**. This mechanism, also known as **sandboxing**, is based on limiting program accessibility to a closed domain, in such a way that the program address space and available resources are confined within this domain.

In the third category—Others against Platforms—the source of the attack can be any external entity that is not part of the agent platform. This external entity can perform attacks against the platform resources (files, communication ports, etc.) or against the platform's communications with the outside. In these cases, security greatly depends on the mechanisms

provided by the operating system. Additionally, a secure communication channel, established using mechanisms such as Transport Layer Security or IPSec can be used to secure the communication between the platform and other parties.

The last type of attack—**Platforms against Agents**—is the most difficult to prevent. It is obvious that if a platform is to execute an agent, it must have complete access to the agent code, state and data. There is nothing to prevent the platform from analyzing the agent code, from corrupting its state or data, from manipulating its execution environment, or from executing it multiple times in order to increase its execution time or to generate erroneous output, for example, generate multiple purchases in a shopping scenario. So there is an inevitable need for agent data to be kept secret from the malicious platform, it must be stored in a way that even the agent itself cannot directly access its code without proving its authorization.

These kinds of Problems are popularly known as **malicious host problem**. Several mechanisms have been proposed to address the malicious host problem. First of all, we present those that have a limited applicability because they can only be used if certain assumptions hold. Then, we will present those approaches that can be effectively implemented in real world applications.

4. APPROACHES TO SECURE MOBILE AGENTS FROM MALICIOUS PLATFORMS

Some of the better known solutions to the malicious host problem are impractical, for they have been designed for particular scenarios that are actually rarely found in real-life applications. The following is a discussion of the better known techniques.

4.1 EXECUTION TRACING

Execution tracing [10] is a technique that allows unauthorized modifications of an agent to be detected upon completion of the agent execution. The technique is based on recording the agent's behavior on each platform in order to build a trace of its execution. The trace is composed of a sequence of identifiers corresponding to the operations executed by the agent. Platforms must produce and maintain traces of all executed agents, so that agent owners can request these traces after the agent has terminated its execution, and verify that the agent code or state has not been maliciously modified.

This approach has several drawbacks, such as the size and the number of logs to be kept by platforms, or the possible lack of connection between the owner and the platforms once the agent has returned to the home platform. Besides, the verification mechanism is too expensive to be applied systematically, and can only be used when the owner has a suspicion that the agent execution has been corrupted.

4.2 CO-OPERATING AGENTS

In [11], Roth describes a protocol for detecting manipulations of the agent execution through co-operating agents. Roth considers applications to be designed using two or more mobile agents that cooperate in order to achieve their goals.

The itineraries of these agents must have no single platform in common, and platforms can collude with each other as long as the collusion does not involve platforms of different agent itineraries. The itinerary and the operations performed by every agent must be tracked by their cooperating agents, so that any tampering with the execution of an agent can be detected by its co-operators. In order for this protocol to be secure, the interaction between the agents must always take place over a secure authenticated channel.

Roth's protocol suffers from several limitations, the first one being the complexity of defining subgroups of platforms that will not collaborate with each other to attack the application. The second limitation is the need to establish a secure authenticated channel between the agent and its co-operators, which may not be possible to provide in all scenarios. Besides, this technique undermines the agent's autonomy, for it requires the agent to interact with other agents in order to carry out its tasks.

4.3 CODE OBFUSCATION

Code obfuscation [12] aims at generating executable agents that cannot be attacked by reading or manipulating their code. This technique is based on transforming the agent code in such a way that it is functionally identical to the original one, but it is impossible to understand it. The approach also establishes a time interval during which the agent and its sensitive data are valid. After this time elapses, any attempt to attack the agent becomes worthless. In [13], a modification of this approach is presented to prevent hosts from repeatedly executing an agent in order to obtain different outputs and draw conclusions about its behavior. This modification is based on recording every input event on a trusted third party.

The major drawback of these techniques is the difficulty in establishing the time required by an attacker to understand an obfuscated code. Similarly, no mechanism is currently known for quantifying the amount of time required by an agent to accomplish its task, especially in heterogeneous environments. As a result, restricting the lifetime of a mobile agent is not feasible in practice.

4.4 COMPUTING WITH ENCRYPTED FUNCTIONS

Computing with encrypted functions is a technique proposed by Sander and Tschudin [14] to achieve code

privacy and code integrity. Their technique is based on creating encrypted programs that can be executed without decrypting them. Supposing that a mobile agent has to execute a certain function f, then f is encrypted to obtain E(f), and a program is created that implements E(f). Platforms execute E(f) on a clear text input value x, without knowing what function they actually computed. The execution yields E(f(x)), and this value can only be decrypted by the agent owner to obtain the desired result f(x).The main problem of this technique is that the authors have only found encryption schemes for polynomials, using homomorphic encryption and function composition techniques. Thus, their proposal is not suitable for general programming.

4.5 TAMPER PROOF DEVICES

The use of tamper-proof devices is based on performing part or the entire agent execution on a physically sealed environment, which can be trusted to execute the agent correctly. Tamper-proof devices can be provided by a trusted third party and, if necessary, they can be inspected periodically to verify that their security has not been compromised. Tamper proof devices can be used to carry out cryptographic operations with a private key that must be kept secret from the remote host. They can also have their own private key, for example, to sign partial results generated by the agent.

Approaches such as [15] propose performing the entire agent execution on tamper-proof devices. The cost of these solutions is high because each platform must be provided with an expensive

tamper-proof device. Besides, these techniques are only suitable for closed environments, such as corporate networks or computational grids, where a tamper-proof device has been installed in every platform. As a result, these techniques imply a loss of agent autonomy.

In order to reduce the cost of the solution, other approaches based on smart cards have been proposed. In these solutions, the tamper-proof device has limited computation capabilities, and is only used to execute security-sensitive operations. However, the security of these approaches is limited because the platform controls the communication between the agent and the tamper-proof device. Thus, the inputs or outputs that are provided to or produced by the device can be easily tampered.

5. CONCLUSIONS

Comparative Study on Approaches to Mobile Agents Security from Malicious Platforms in Mobile Adhoc Networks infers that there is a critical need of research to identify different realistic approaches to secure mobile agents. More focus on protecting the agent’s itinerary and agent’s computational results will help us to provide useful solutions to Mobile Agents Security.

6. FUTURE SCOPE

Wide range of research is going on towards providing security to mobile agents.

Some noticeable ways for further development of realistic solutions to mobile agent's system security are:

- Real Time detection of attacks on mobile agents.
- Mutual authentication between mobile agent and its host.
- Protecting the agent computational results.
- Protecting the agent's itinerary. Self-protected mobile agents.

7. REFERENCES

[1] Chess D.M: Security issues in mobile code systems. In: mobile agents and security, Editor Vigna, vol. LNCS1419. Springer-Verlag 1998.

[2] Mousa Alfalayleh and Lijiljana Brankovic, "An Overview of Security Issues and Techniques in Mobile Agents", The School of Electrical Engineering and Computer Science, The University of

Newcastle, Newcastle, NSW 2308, Australia.

[3] Borselius, N: Mobile agent security.

Electron. Communication Engineering. IEEE London 14(5),211-218(2002).

[4] J.Algesheimer et al.,"Cryptographic Security for Mobile Code,"Proc.2001 IEEE Symp. Security and Privacy, IEEE Press,2001.

[5] N.Karnik, Security in Mobile Agent Systems. Ph.D thesis, dept. of CS&E Univ. of Minnesota.

[6] Schelderup, K.olnes, J: Mobile agent

security-Issues and Directions. In: Proceedings of the 6th Int. Conf. on intelligence and services in networks Barcelona, Spam, Apr 1999.

[7] Butscgje, L.; Paprzycki, M.; and Ren, M.2004. Mobile agent security-an overview, In E. Niedzielska et al (eds), Modern information Technologies in Management Wroclaw Univ. of Economics Press,pp.600-608.

[8] William M.Farmer, Joshua D. Guttman, and Vipin Swarup, "Security for Mobile

Agents: Authentication and State Appraisal", pp. 5-11, pp. 118-130, 1996 European Symposium on Research in Computer Security (ESORICS).

[9] J. Mir and J. Borrell. Protecting Mobile Agent Itineraries. In Mobile Agents for Telecommunication Applications (MATA), volume 2881 of Lecture Notes in Computer Science, pages 275–285. Springer Verlag, 2003.

[10] G. Vigna. Cryptographic Traces for Mobile Agents. In Mobile Agents and Security, volume 1419 of Lecture Notes in Computer Science, pages 137–153. 1998.

[11] V. Roth. Mutual protection of co operating agents. In Secure internet programming: Security issues for mobile and distributed objects, volume 1603 of Lecture Notes in Computer Science, pages 275–285, 1999.

[12] F. Hohl. Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts. In Mobile Agents and Security, volume 1419 of Lecture Notes in Computer Science, pages 92–113. Springer Verlag, 1998.

[13] F. Hohl and K. Rothermel. A Protocol Preventing Blackbox Tests of Mobile Agents. In Proceedings of Kommunikation in Verteilten Systemen (KiVS '99), pages 170–181. Springer-Verlag, 1999.

[14] T. Sander and C. F. Tschudin. Protecting Mobile Agents Against Malicious Hosts. In Mobile Agents and Security, volume 1419 of Lecture Notes in Computer Science, page 44. Springer Verlag, 1998.