Countering Security Vulnerabilities in Agent Execution using a Self Executing Security Examination

John Page, Arkady Zaslavsky, Maria Indrawan
School of Computer Science & Software Engineering, Monash University, Melbourne
{john.page,arkady.zaslavsky,maria.indrawan}@infotech.monash.edu.au

Abstract

The paper describes the evolution and implementation of a self-executing security examination (SENSE) for agents executing in non-trusted domains. It outlines the shortcomings of some of the existing agent security schemas. To meet these shortcomings, the paper proposes the implementation of SENSE as a reliable method for detecting a malicious action attempted on an executing agent at run-time. The incorporation of SENSE into its normal operating process will make the agent self-reliant with respect to its security function and allow it to confidently execute in an alien environment, without having to depend on an external support for security. To build a strong case for its adoption, the paper elaborates the ease of operation, simplicity and efficiency of the schema.

1. Introduction

Several real world applications in different business domains extend the belief that agent technology will be the solution for a resource-starved network centric distributed computing world. While several definitions have been put forward, Franklin and Graesser’s definition [1] captures best the flavour of an agent. It perceives an autonomous agent as a system that exists within an environment, which it senses and reacts to over time while pursuing its own goals so as to, affect its future reactivity to the system.

A security schema built for agent security should allow the agent to manage its own security related functions effectively without having to depend on the agent server for resources. The self-executing security examination (SENSE) is a security schema under which the agent is programmed to carry its own security implementation, as a scanning algorithm. This algorithm scans the agent code and verifies the integrity of the agent at random intervals. There are two aspects to the implementation of this schema. The first is the scan algorithm and the second is hiding the algorithm within the business function of the agent. To implement the scheme, the agent’s code is divided into different virtual zones. On arriving at a particular remote host, the agent reads its own binary images of its own code from these different virtual zones. The creation of zones is handled internally by the agent and is transparent to the rest of the schema. The values returned by this scan are compared against a code map carried by the agent. This code map is provided to the agent by its creator and is protected using a digital signature. Every scan of the agent code is compared against the code map. If the server at which the agent is docked is malicious and attempts to modify the agent code, in order to subvert or to manipulate the agent’s behaviour, the agent is able to detect these attempts. By re-executing the scan algorithm at various random intervals, the agent is able to verify its integrity at run time, without having to rely on the host server for any additional resources such as the execution of cryptographic routines to perform security checks.

Another advantage of this schema is that it does not require any external condition for execution. This is a definite advantage over the environment key generation [2] technique, which relies upon a particular environmental condition becoming true in order to generate a key. This key, cryptographically unlocks a statement, which triggers off the executable code. A significant drawback of this method was the reliance of the agent, on the platform to execute it. The SENSE schema is able to counter this disadvantage by enabling an agent, the possibility of executing the algorithm at will. The next section describes the implementation of the security schema.

2. Implementation of SENSE

The SENSE schema has been implemented successfully with Grasshopper [3] mobile agents. The agent platform used is Grasshopper version 2.2.4b, working on WINDOWS 2000 machines. SUN’s Java version 14.2_02 has been used. The prototype implementation of SENSE currently gives the agent an option of either initiating the security check or altogether skipping it. In trusted domains, the security schema can be suspended while the agent continues with its normal processing. While travelling through a non-trusted domain; the security function can be activated. The algorithm described in figure 1, illustrates the SENSE schema in operation from the point the agent arrives at the agent server. A detailed overview of the SENSE schema can be found at [5].
The trigger for activating the security could be inbuilt into the agent, or could be remotely controlled. On initiating the security option the agent will perform a scan of its code. After each scan, a report tabulating the scan results is prepared as shown in figure 2.

![Figure 1. SENSE Algorithm Psuedocode](image)

![Figure 2. SENSE Schema Scan Report](image)

Depending on their business functionality and resource requirements agents are moved between various agencies and places. While on one hand, this flexible mobile architecture allows the agent to react in various ways while attempting to meet its business requirements. Unfortunately, this feature also makes the agent a target for malicious attacks [4]. The SENSE schema allows an agent to scan its code at any particular instant of execution. The scan returns a list of agent code element values. These results are compared against the code map assigned to the agent at its home base. The MAS creating the agent digitally signs the code map. This code map refers to static chunks of agent code that are not intended to change during the course of its execution. Further, these code maps also refer to critical pieces of agent functionality, which if tampered with can lead to unpredictable and possibly undesirable consequences. Thus, using the SENSE schema the agent is able to detect a malicious action carried out against it, by a particular agent server. The decision to execute the schema rests with the agent and does not require any server intervention. This feature makes the SENSE schema self-reliant in nature as opposed to other proposed schemas. The agent, at run-time can decide the periodicity of the schema execution. Further, the schema execution time can also be pre-programmed into the agent’s functionality before it leaves its home base. The advantage of this flexibility is that the agent is allowed to manage its security depending upon the threat it perceives from a particular agent server.

3. Conclusion and Future Work

The paper examined proposed security implementations for agent security and highlighted their limitations. It also recognized the need for a self-reliant mode of security, which protects the interests of MAs by detecting malicious attacks against it by hostile servers. To overcome the existing limitations of security implementations, the SENSE schema is proposed. This schema succeeds in providing the agent with a flexible mechanism using which the agent can make reliable integrity checks at run time. Any malicious attempts to manipulate the agent code at run-time can be detected and evasive action can be taken.

The advantage of the SENSE schema over other proposed schemas is the self-reliance and independence it gives to the agent with respect to its security function. Further, the simplicity and openness of the SENSE schema gives to the agent with respect to its security function. The advantage of the SENSE schema over other proposed schemas is the self-reliance and independence it gives to the agent with respect to its security function. Further, the simplicity and openness of the SENSE schema encourages agent servers to accept agents rather than view it suspiciously and block its action. Future work will attempt to extend the SENSE schema to agent communities, giving agents the option of security at individual as well as at community level.

4. References