MAIS: MOBILE AGENT INTEGRITY SYSTEM
A Security System to IDS based on Autonomous Agents

Rafael Páez, Joan Tomàs, Jordi Forné and Miquel Soriano
Telematics Engineering, Technical University of Catalonia, Jordi Girona 1-3, Barcelona, Spain

Keywords: Multiagent systems, mobile agents, Software Watermarking, Intrusion Detection Systems.

Abstract: Intrusion Detection Systems based on autonomous agents are a promising technology due to their scalability, resilience to failures, independence and reduction of network traffic. However, when used to protect critical systems, the IDS by itself can be the target of malicious attacks. In this paper we propose a security system to verify the integrity of the IDS agents during their execution time, by using software watermarking techniques.

1 INTRODUCTION

The security of software systems has become an important topic because they provide the functionality of critical systems controlling important infrastructures like centres for disasters prevention, intelligent buildings, planes' functions automation, etc. So, many human lives and important amounts of money strongly depend on the confidentiality, integrity and availability of software systems which must be protected to warranty the required level of security. There are several tools that are used to provide this security, such as firewalls, honeynets, honeypots, and Intrusion Detection Systems (IDS). However, since the reliability of the whole system relies on the proper function of them, the tools their selves become objectives susceptible to be attacked and therefore they also need to be protected.

Intrusion Detection Systems detect suspicious activities and possible intrusions in a system or private network at the moment at which these happen. The different entities that compose the IDS need to communicate among them, therefore is important to keep in mind security communication services such as integrity of the information, authentication and access control.

One of the important characteristics of security systems and particularly of IDS, is the cooperation among its components in order to achieve their global objective and to reduce central processing. By this reason, an agent-based technology has been proposed to be integrated with IDSs, since they carry out the processing *in-situ* and they can autonomously communicate to each other.

The main security limitations that affect the deployment of mobile agents are multiplied in IDS based on autonomous agents, since IDS by itself are one of the main objectives to be attacked by malicious users. In this article we focus our attention in Autonomous Agents for Intrusion Detection, identifying a particular threat for these systems and then proposing a solution to increase the security against this potential attack. Our proposal is based on an IDS system architecture based on autonomous agents named Autonomous Agents For Intrusion Detection (AAFID). In the AAFID system there are three types of entities: monitors, transceivers and agents, hierarchically organised in a tree infrastructure.

Our objective is to analyze a risk scene and to propose a possible solution. In section 2, we introduce the related background, including software agents, watermarking techniques and IDS based on agents and its security. In the section 3, we present a risk scene. In section 4 we present a system named MAIS, its architecture and the operation protocol. Finally section 5 concludes.

2 STATE OF THE ART

In this section we overview the related background necessary to understand the solution that we present here. Likewise, we analyze the security problems

Páez R., Tomàs J., Forné J. and Soriano M. (2007).
MAIS: MOBILE AGENT INTEGRITY SYSTEM - A Security System to IDS based on Autonomous Agents.
In *Proceedings of the Second International Conference on Security and Cryptography*, pages 41-47
DOI: 10.5220/0002120500410047

and some solutions that have been presented in the literature. More specifically, we introduce Intrusion Detections Systems, agents, software watermarking techniques and the main existing proposals about IDS based on autonomous agents, including a security analysis.

2.1 Intrusion Detection Systems

An Intrusion Detection System tries to detect and to alert about suspicious activities and possible intrusions in a system or particular network. An intrusion is an unauthorized or non wished activity that attacks confidentiality, integrity and/or availability of the information or computer resources. To reach its goal an IDS monitors the traffic in the network or gets information from another source such as log files. The IDS analyzes this information and sends an alarm to the system administrator. The system administrator decides to avoid, to correct or to prevent the intrusion.

Basically an IDS has an events generator, an analyzer or sensor and a response module. The event generator sends the packets to the events collection module that communicates with the sensor. The sensor filters the information and discards irrelevant data. The response module decides whether to send or not an alarm according to the policy held in its database (Goyal, Sitaraman, and Krishnamurthy 2003). An IDS can be classified according to its *location*, it can be Network based IDS (NIDS) or Host based IDS (HIDS); according to the *detection mechanisms*, it can be misuse detection or anomaly detection; and according to its *nature* it can be passive or reactive.

2.2 Agents

There are different definitions of agents (Balasubramaniyan et al, 1998), (Nwana, 1996), (Jansen et al, 2000). In general, an agent is a software entity that works autonomous and continuously gathering data to accomplish an action on behalf of a person or another agent. Autonomously means that it can work without direct intervention of a human or other system and has the control of its internal state and its actions.

2.3 Software Watermarking

Watermarking techniques have been basically used to ensure the protection of digital contents. With these techniques, some information (usually called mark), is embedded into a digital content like video, audio, software, (Figure 1). The main objective is to keep this information imperceptible in all copies of the content that we protect in such a way that we can later demand the authorship rights over these copies. In software watermarking, the mark must not interfere with the software functionalities. The mark can be: static, when it is introduced in the source code, or dynamic, when it is stored in the program execution states.

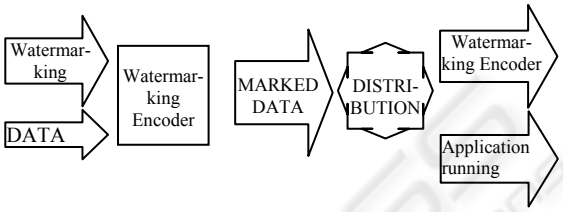

Figure 1: Software Watermarking.

There are three basic aspects to consider when a watermarking technique is designed: the required data rate, the type of source to mark (native binary code, bytecode, etc.) and the expected threat model (translation, optimization, obfuscation of code, etc.).

To retrieve the watermark we need a *recognizer*. Recognizers are designed to extract the watermark from the program execution with a specific input. Recognizers can be defined from **trivial** (does not assure that the watermark can be retrieved) to **strong or ideal** (resistant against all kind of transformations). And according their operation, recognizers can classified from **static**, when only the source code is analyzed, to **pure dynamic**, when only program execution state is examined.

2.4 IDS based on Autonomous Agents

According to (Jansen et al, 2000), (Lange et al, 1998) and (Dorothy et al, 1987), there are several advantages of mobile agents that make them appropriate to IDS: scalability, resilience to failures, independence, reduction of network traffic, when another agent is generated it is not necessary to restart the system, solution to complex tasks, etc.

The architecture for IDS based on autonomous agents has the following components: monitors, transceivers, agents and filters. Definition of each component and further information can be found in (Balasubramaniyan, 2003). The AAFID system (Balasubramaniyan et al, 1998) includes a user interface and several components of its architecture. User interfaces use APIs that the monitor exports, to ask for information and to provide instructions. In the AAFID system there are three types of static entities: monitors, transceivers and agents, hierarchically organised with a tree infrastructure.

2.5 IDS based on Autonomous Agents Security

To protect the entities of the IDS, it is necessary to protect both the platform and the agents. Mobile agents offer many functional advantages, but there are new threats due to their mobile nature. The more common threats are: agent against platform, platform against agent, agent against other agents and other entities against the agent system. Several solutions have been proposed to reduce these risks (Table 1) but particularly the threats of platform against agents are the most difficult to avoid.

Table 1: Countermeasures for attacks of platforms against agents.

CONTERMEASURES
Partial results encapsulation (Yee, 1997)
Mutual itinerary recording (Roth, 1998)
Itinerary recording with replication and voting (Schneider, 1997)
Execution tracing (Vigna, 1997)
Environmental key generation (Riordan, 1998)
Computing with encrypted functions (Sander et al, 1998)
Obfuscated code (Hohl, 1998)
Cooperative agents (Roth, 1998)
Limiting the execution time (Esparza et al, 2003)

2.6 Protecting Agents Against Malicious Hosts

Particularly, the attack carried out by a platform against an agent is very difficult to avoid, because the platform has total access to data, code and results of the agent. So, if a host is malicious, it can easily isolate the agent and extract information to corrupt it or modify its code or its state. Other extreme measures that a malicious host could perform are to analyze the operation of the agent or to apply inverse engineering to introduce subtle changes and to force the agent to be malicious, reporting false results.

3 RISK SCENE

In an IDS based on autonomous agents, a monitor controls a network segment and it sends a transceiver to each host. Likewise, various agents are generated by a transceiver in order to monitor a determined type of traffic and they send alerts of suspicious activities to the transceiver on which they depend within the tree structure. One of the existing threats in these systems is when an intruder attempts to replace any IDS entity by another with similar characteristics but subtly modified in order to avoid a particular suspicious activity. So, if an agent or transceiver is modified or replaced, they will not report their correct results to their correspondent monitor and likewise, if a monitor is replaced it will not avoid or prevent the forthcoming attack.

Security solutions in IDS based on agents are the same that are offered for any environment that use agents. However, all the requirements are not covered; in particular, the threats against the IDS, its components and communications are not faced. So, in this paper we propose to detect attacks against any IDS entity with a new security scheme named MAIS.

4 MAIS (MOBILE AGENT INTEGRITY SYSTEM)

We propose a new system to verify not only the integrity of transceivers located in different hosts of the IDS architecture, but the correct execution of the transceivers during its operation. The MAIS system architecture is similar to AAFID system, but the transceivers and monitors behave like mobile agents and their mobility is limited, they only can displace to their corresponding trusted entity, that is to say, the upper level entity from which they depend. The data collection agents are static and they conserve the same characteristics of the AAFID system agents.

4.1 MAIS Architecture

The MAIS architecture has three essential components: monitors, transceivers and data collection agents. The monitors are agents that are located in the high levels of the infrastructure, they carry out correlation of information of high level and they control a network segment. There is a root monitor located in the higher level. It has the ability to communicate with an administrator interface and it also can provide the access point for the whole MAIS system. The administrator interface is independent of the IDS entities, in order to permit different implementations. The monitors can also control other monitors and besides they are in charge of emitting and to control another type of agents called transceivers.

In MAIS, the monitors are also Trusted Parties, which are in charge of identifying the entities that

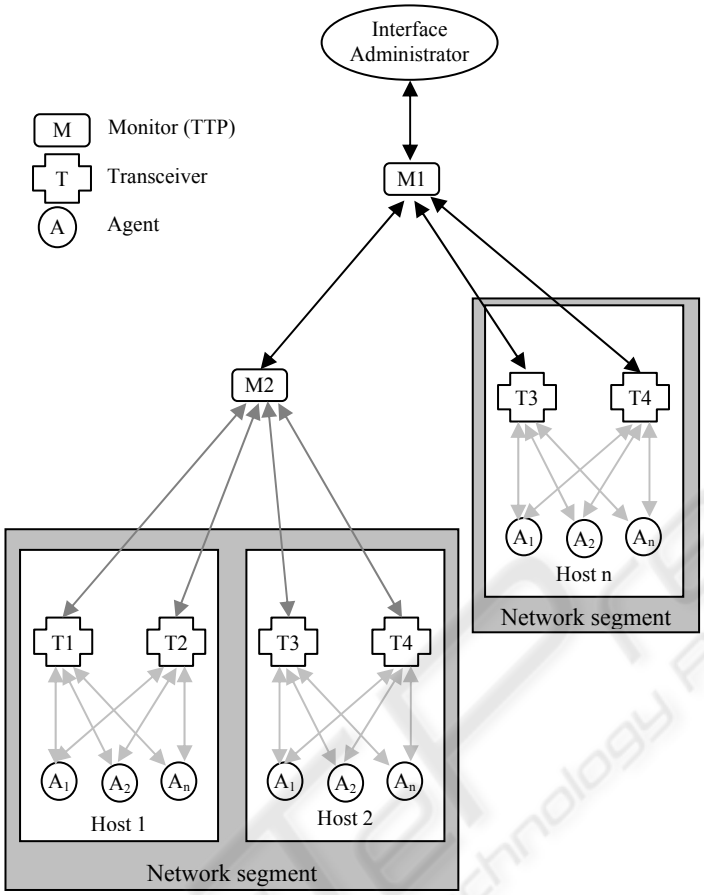

Figure 2: MAIS system architecture.

they control and to carry out the process of watermarking recognition. The watermark allows us to verify not only the transceiver's or monitor's integrity but also their correct execution (a wrong execution generates a wrong watermark).

Transceivers carry out correlation functions and they send the information to the monitor which they depend from. Transceivers have information about the host where they reside and also control the underlying agents. The main differences between an AAFID transceiver and a MAIS transceiver are the mobility and the mark.

The data collection agents inside the MAIS infrastructure are in charge of monitoring a host and its behaviour. The agents and their transceivers are located in the same host.

In the MAIS system, the transceivers and monitors must be mobile because they have to displace from its host to their TTP. This TTP is the immediately superior entity in the infrastructure, which will be able to do the mark verification; therefore, it is necessary to establish new characteristics for the system. The first one is that all the monitors and transceivers of the IDS must be mobile. The second one is that an entity which controls to another entities must behave as a trusted party when thus be required and to perform the mark verification. The third one is that each host must have at least two transceivers being able to carry out the same function, so when an agent is sent to the TTP to verify the integrity of its code and of its execution, another agent replaces its functionality.

The transceivers depend on monitors and monitors likewise can depend on other monitors (Figure 2), but the transceivers can only control their underlying static agents (data collection agents). So, the monitors are required to be trusted parties and they control the marking and verification processes to its underlying entities. The monitors have an overview of a network segment and the transceivers have an overview of a host

4.2 MAIS System Operation

The system operation protocol is as follows:

1. A monitor generates an entity and its corresponding support entity.
2. Subsequently it performs the watermarking process on each entity and sends them to the destination host conserving its timestamp.
3. The entity moves to its destination host to carry out its function.
4. The agents periodically move to their generating entity according the established time (timestamp). While an entity goes to verify the integrity of its code and of its execution, the support entity continues carrying out their work.
5. When an agent arrives to verify its integrity, the issuing entity performs functions of third trust party verifying the mark of the agent. If the agent has been compromised, the TTP eliminates it and isolate the host in which was residing, considering it malicious.
6. In case that an agent do not arrive on the established time to perform the verification process, the host is isolated and the agent eliminated.

The issuing entity conserves the timestamp to verify that each agent arrives on time to control its integrity in determined periods of time. In each host there will be at least two entities executing the same function to provide service continuity while an entity displaces to carry out the verification of its integrity When the agent arrives to the verifier it verifies the mark. Is important to note that an incorrect transceiver's execution generates a wrong mark; so, the system administrator can detect the anomalous behaviour and perform the corresponding security measures.

On the other hand, watermarking techniques are used instead of digest techniques because the transceivers are constantly being self modified to incorporate the new collected information.

4.3 Watermarking Layer

The algorithm that we use to embed the mark is the *Dynamic Graph Watermarking* (Collberg and Thomborson, 1998) and (Collberg and Thomborson, 1999), but others watermarking algorithms may be considered.

4.3.1 Watermarking Algorithm Overview

The main characteristic of the *Dynamic Graph Watermarking* algorithm is that it offers protection against distortive de-watermarking attacks as obfuscation or optimization. The basic idea is to embed the mark into a graph topology. This graph is dynamically built during run time. As it is well known, dynamic graph structures are hard to analyze. On the other hand, semantic source code modification does not affect these algorithm performances because execution results must be the same, in other words, the agent has to generate the same graph structure of its watermark.

The mathematical hard problem used by this algorithm is the same that is used by public key cryptography: the prime number factorization. In other words, if n is the product of two bigger prime numbers p and q, calculate p and q from n is a hard computational problem. Applied in this environment, a system that is able to embed this number n into a graph structure in an agent can be a good option to prove the legal origin of one code. That is to say, as the legal owner has the values that factorize n and the method to retrieve the value of n, he can prove his ownership. From this point of view, the efforts to solve the watermarking problem will be concentrated in mark embedding and extracting methods.

4.3.2 Mark Embedding

Basically, there are two encoding techniques to embed the mark into a graph topology: Radix-K and Enumeration.

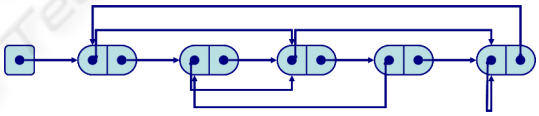

Figure 3: Radix-K encoding example.

Radix-K encoding consists in a graph with circular linked list structure. This list has an extra pointer field which encodes a base-K digit. A null pointer encodes a 0, a self-pointer encodes a 1, a pointer to the next node encodes a 2, etc. Figure 3 shows, as an example, the codification for $61 * 73 = 4453 = 3*6^4 +2*6^3 +0*6^2 + 4*6^1 + 1*6^0$. On the other hand, Enumeration Encoding is based in the work of F. Harary and E. Palmer in (Harary and Palmer, 1973).

4.3.3 Embedding Process

As shown in figure 4, the owner selects n as product of two big prime numbers p and q. n is embedded in the topology of a graph G. After that, a source code W which builds G is constructed. W is embedded into the original agent O and the watermarked agent O_0 is obtained. When O_0 is run with I as input, the

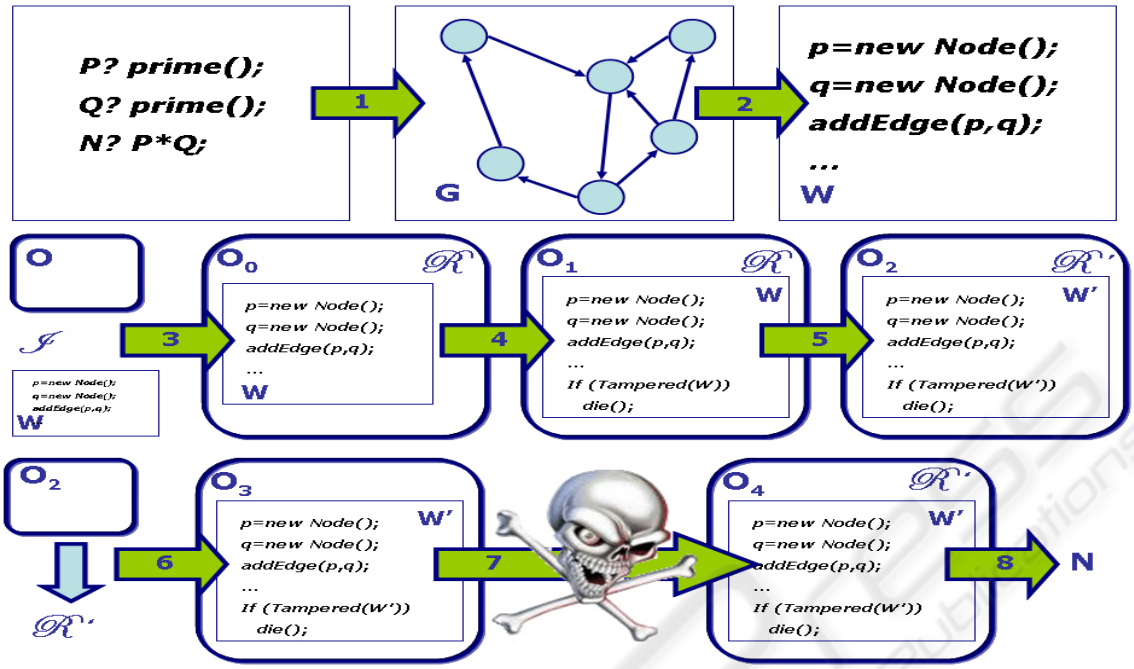

Figure 4: Embedding and Extraction process.

graph *G* will be built and the recognizer *R* is constructed. The objective of *R* is identified *G* on the heap of the agent execution. After that, tamperproofing and obfuscation techniques can be applied (see O_1 and O_2). Finally, the recognizer is extracted from the application and O_3 is sent to its destination host. When a malicious agent O_4 is moved to their generating entity, this entity can identify if the execution of this agent has been modified linking O_4 with *R* and executing them with *I* as input. As result, the modified watermark *n'* is obtained and this entity can verify that the original factors *p* and *q* can not factorize *n'* and it allows to detect the malicious agent. In other words, if n and *n'* does not match; the agent execution has been modified.

4.3.4 Mark Extraction

As was commented before, the idea is to construct a graph in memory which topology embeds the mark. To recover this mark, an extraction process is needed. One method can be to examine all reachable heap objects but this can be a hard computational problem. Instead of this, the input *I* is divided in parts an every part builds a portion of the watermark. As a result of the last part, the recognizer returns the *root* node of the watermark.

4.3.5 Watermarking Justification

Digital signatures are widely used to guarantee the code integrity and authenticity. The digital signature can be used to verify, at a given moment, that a software code is exactly as created. However, it cannot assure that the code was properly executed over a period of time.

In the IDS, given that the transceivers are changing continuously because they are collecting information, digital signatures techniques are inappropriate. Moreover we want to provide not only transceivers integrity but the correct execution of the transceiver. Therefore, we propose to use a watermarking technique which is suitable because the mark is dynamically built during run time and if the semantic source code is modified the agent has to generate the same graph structure of its watermark, otherwise it indicates that the agent execution has been modified.

5 CONCLUSIONS

The attacks of malicious hosts against the agents are considered one of the problems most difficult to solve and there is not a form of protection that eliminates them completely. To offer a determined security level in an IDS based on agents is necessary

to combine different techniques that permit to detect an attack although it cannot be avoided. The drawback to send an agent to a malicious host is that this can be attacked, because of the host has total access to the code and data, therefore, to carry out a verification of its integrity, we propose the use of trusted monitors using watermarking techniques to verify the proper working of the IDS software components..

REFERENCES

B. Goyal, S. Sitaraman, S. Krishnamurthy, 2003. Intrusion Detection Systems: An overview. SANS Institute 2001, as part of the Information Security Reading Room.

J.S Balasubramaniyan, J.O. Garcia-Fernandez, D. Isacoff, E. Spafford, D. Zamboni, 1998. An Architecture for Intrusion Detection using Autonomous Agents, *Proceedings., 14th Annual Computer Security Applications Conference,* pages 13 – 24

H.S. Nwana, 1996. Software Agents: An Overview, *Knowledge Engineering Review,* 11(3), pages 1-40

W. Jansen, P. Mell, T. Karygiannis, D. Marks, 2000. Mobile Agents in Intrusion Detection and Response, *Proc. 12th Annual Canadian Information Technology Security Symposium*, Ottawa.

D. Lange and M. Oshima, 1998. *Programming and deploying java mobile agents with agle*, (Addison-Wesley)

Dorothy E. Denning, 1987. An intrusion detection model, *IEEE Transactions on Software Engineering*, 13(2), pages 222-232.

W. A. Jansen. Countermeasures for mobile agent security, 2002. *Computer communications, Special Issue on Advanced Security Techniques for Network Protection,* 25(15), pages 1392-1401

B.S. Yee, 1997. A Sanctuary for Mobile Agents. Technical Report CS97-537, University of California in San Diego.

V. Roth, 1998. Secure Recording of Itineraries Through Cooperating Agents, *Proc. of the ECOOP Workshop on Distributed Object Security and 4th Workshop on Mobile Object Systems: Secure Internet Mobile Computations*, France, pages 147-154.

F.B. Schneider, 1997. Towards Fault-Tolerant and Secure Agentry, *Proc. 11th International Workshop on Distributed Algorithms, Saarbucken*, Germany, pages 1-14.

G. Vigna, 1997. Protecting Mobile Agents Through Tracing, *Proceedings of the 3rd ECOOP Workshop on Mobile Object Systems*, Jyvälskylä, Finland.

J. Riordan, B. Schneier, 1998. Environmental Key Generation Towards Clueless Agents, *Lecture Notes in Computer Science*, 1419, pages 14-24.

T. Sander, C. Tschudin, 1998. Protecting Mobile Agents Against Malicious Hosts, *Lecture Notes in Computer Science*, 1419, pages 44-60.

F. Hohl, 1998. Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts, *Lecture Notes in Computer Science*, 1419, pages 92-113.

O. Esparza, M. Soriano, J. L. Muñoz, J. Forné, 2003. A protocol for detecting malicious hosts based on limiting the execution time of mobile agents, *8th IEEE Symposium on Computers and Communications*. 1, pages 251-256.

Christian Collberg and Clark Thomborson. On the limits of software watermarking.Technical Report 164, August 1998.

Christian Collberg and Clark Thomborson. Software watermarking: Models and dynamic embeddings. In Principles of Programming Languages 1999, POPL'99, San Antonio, TX, January 1999.

Frank Harary and E. Palmer. Graphical enumeration, 1973. Academic Press, New York