

INTELLIGENT AND MOBILE AGENT FOR INTRUSION DETECTION SYSTEM : IMA-IDS

Farah Barika farah.barika@ragingbull
Nabil El Kadhi el-kad_n@epitech.net
Khaled Ghedira khaled.ghedira@isg.rnu.tn
Laboratoire SOIIE, Tunisie & Laboratoire LERIA, France.

Abstract: Over the years computer systems have successfully evolved from centralized monolithic computing devices supporting static applications into distributed computing called Networks, therefore our systems are becoming more open and subject to set of security threats. Thus, a key challenge is to provide all computer systems with the appropriate mechanisms to offer security services such as authentication, secret preservation and automatic attacks detection commonly known as Intrusion Detection. Intrusion Detection Systems (IDS) are used to discover several kinds of attacks. Commercial solutions are generally centralized and suffer from significant limitations when used in high-speed networks. This is one of our major motivations to use the distributed model based on mobile agent platforms. We believe that agent characteristics will help collecting efficient and useful information for IDS. Thus, in this paper, we propose an Intelligent Mobile Agent model for distributed IDS called IMA-IDS. Before introducing our global Intelligent Mobile Agent IDS architecture, we will first argue for the use of mobile agents in IDS and then, we will choose an agent platform which can offer security mechanisms needed by IDS solutions. Last we will demonstrate the feasibility of our model with a working prototype. The result obtained from the implementation of our IMA-IDS is a proof of the suitability of using Agent concept for IDS architecture.

Keywords: Network, Security, Intrusion detection system, Distributed intrusions detection, Intelligent mobile agent.

1 INTRODUCTION

Security is crucial to the success of active networking especially when the current network is characterized by a dynamic nature and an increasing distribution. Traditional network relies on security mechanisms and policies deployed on the underlying operating system. Nevertheless, these measures are insufficient and they present, in general, a set of flaws that result in security vulnerabilities [CLM + 99]. The field of automated computer security intrusion detection gives result to Intrusion Detection System (IDS for short). The goal of IDS is to analyse events on the network and identify manifestations of attacks. Commercial solutions are generally centralized and suffer from significant limitations when used in high-speed networks. The identification of distributed intrusions requires cooperation of different sensors so it is advisable to consider mobile devices as a challenge to intrusion detection. Our motivation is to distribute intrusion detection using mobile and intelligent agents. This paper is organized as follows. This is the introduction, section 2 presents the most important features of existing IDS solutions and their limitations. In section 3 we argue for the use of mobile agent. Section 4 compares the most used agent platforms and concludes by choosing Aglets platform [DM98] that offer the most interesting security features for our solution as presented in [EKBBGe03]. Finally section 5 describes our IMA-IDS by presenting its architecture and the actual prototype implementation.

2 BACKGROUNDS IN IDS

Intrusion Detection Systems (IDS) plays an important role in achieving survivability of information system and preserving their safety from attacks [VEK]. Attacks against a system are informally defined as a deliberate attempt to violate the security policy. Intrusion detection has been achieved by following two different strategies of analysis [MHL94]:

- *Anomaly detection*: relies on models of "normal" behaviours of a computer system. Behaviour profiles may be focused on users, applications or networks. Anomaly detection compares the defined profiles against the actual usage patterns to detect "abnormal" activity patterns. These patterns will be considered as intrusions.
- *Policy detection*: relies on a set of attack descriptions called attack-signatures. Both anomaly and policy detection present advantages and disadvantages. In fact, policy detection is limited by attack definitions. Anomaly detection permit

detection of previously unknown attacks; this advantage causes a large number of false positive occurring when an IDS sends an alarm for an event that is not an intrusion [VCF02]. Commercial IDS products such as NetRanger [http], RealSecure [iteB7] and Omniguard Intruder Alert [httpa] are in general based on policy detection.

IDS are usually classified as network-based (NIDS) or host-based (HIDS) IDS [MHL94]. The most important difference between these two IDS categories is the fact that NIDS rely on information obtained by monitoring the network, while the HIDS perform their analysis on information collected at a single host. Since HIDS works above the network layer, it is unable to detect some kinds of attacks [Ran01]. Otherwise, NIDS infer their decision from low-level network packets travelling among hosts [HDL + 90] and are generally able to detect such attacks.

2.1 IDS Requirements

In [JMKM99], authors have defined a set of desirable characteristics for an IDS by focusing two themes : functional and performance requirements. In the following section, we summarize some of these characteristics.

2.1.1 Functional Requirements

- IDS must continuously monitor and report intrusion.
- IDS should have a very low false alarm rate.
- IDS should provide enough information to repair the system in the case of detection of intrusion. Notice that this characteristic depend on IDS goals. In fact, many IDS solutions focus only on alerting administrators without suggesting any corrective actions.
- IDS must detect and react to distributed and coordinated attacks. This detection feature is one of the most difficult because it needs a huge distributed amount of information in addition to the hard task of synchronisation between different hosts.
- The IDS should be adaptive to network topology and configuration changes.

2.1.2 Performance Requirements

- Intrusion should be detected in real-time as it should be reported immediately in order to minimize network damage.
- The IDS must be scalable in order to handle additional computational and communication loads.

2.2 IDS Limitations

The most common IDS shortcomings include the following :

- High number of false positives.
- Lack of efficiency : usually when an IDS is faced with a very large number of events in the network, it slows down a system or drops network packets.
- Vulnerability to attacks : many IDS have hierarchical structures. This fact gives attackers the opportunity to harm the IDS by cutting off a control branch or even by tacking out the root command.

In addition to the forementioned shortcomings, our solution aims to overcome the following limitations :

- Many of the existing network and host based IDS perform data collection and data analysis essentially by using a monolithic architecture [BGFI + 98]. The centralized detection scheme suffer from a number of problems :
 - A central analyser presents a favourable target to attackers. If an intruder manages to decapitate it, the entire network loses protection.
 - A high network load leads to excessive data traffic, the system suffers from scalability problems. A single analyser unit limits the network size.
 - Since network data collection is performed in a host different than the one in which analysis is performed, intruders can perform insertion and evasion attacks [PN98].

Intrusions can be conducted through several steps that occur at different hosts, and consequently cannot be detected by a single sensor. The cooperation of different sensors becomes an imperative for the identification of distributed intrusions. Thus, mobile devices offer a new approach to IDS implementation. Typically, mobile agent technology can solve the set of the shortcomings mentioned above.

3 USEFUL CHARACTERISTICS OF MOBILE AGENTS

Agent Systems are used in various applications such as workflow, scheduling and optimisation [Ghe93]. It is advisable to define what is an agent. We refer to [Pal98] :

- An agent is a physical or a logical entity characterized by the following attributes :

- Autonomy : agents are independently running entities, they operate (in ideal cases) without human control.
- Mobility : agents are able to suspend processing on one platform and to move to another one where they resume execution.
- Rationality: agents embody the capacity to analyse and solve a problem in a rational manner.
- Reactivity: agents perceive their environment and adapt their behavior in a dynamic way to match, as soon as possible, new environment parameters.
- Inferential capability: agents are able to share a set of knowledge in order to achieve a specific goal.
- Pro-activeness: agents can decide to adapt their behaviour to their environment.
- Social ability: agents are able to meet and interact with other agents. The interaction and collaboration between agents is achieved by an agent communication language and it may depend on an ontology.

Accordingly to the previous attributes, we will argue for the use of Mobile Agent to improve the characteristics of the IDS and to overcome the limitations described in section 2.2.

- Reducing Network Load : Existing IDS are faced with the problem of performing a huge amount of data over transfer. Abstracted forms of this data are usually sent from all locations in the network to the central site in order to be processed. Sending a huge amount of data causes an increase of a network loads. Mobile agents offer the opportunity to overcome this problem by eliminating the need of so much data transfer. The processing program (agent) can be dispatched to the host containing crucial data. This will reduce network traffic since an agent is smaller than the processed data.
- Overcoming Network Latency: Mobile agents are able to dispatch from a host to carry out operations directly to the remote point of interest, thus agent scans provide an appropriate response faster than a hierarchical IDS that has to communicate with a central coordinator based elsewhere on the network.
- Asynchronous Execution and Autonomy: Agents can be stopped and started without disturbing the rest of the IDS. Notice that the mobile agents are able to continue to operate autonomously even if the host platform where it was created is not available or is disconnected from the network. Mobile agent frameworks provide IDS with the possibility of continuing to work even when a central controller is down.

- **Dynamic Adaptation:** Mobile agents can be retracted, cloned, dispatched, killed or put to sleep as network's configuration; topology and traffic characteristics change over time. As the number of nodes in the network increases, agents can be cloned and dispatched to these new computing elements.
- **Robust Behaviour:** Mobile agents have the ability to react dynamically to security conditions making it easier to build robust distributed systems.
- **Scalability:** Distributed mobile agents IDS are one of several options that allow computational load and diagnostic responsibilities to be distributed throughout the network [JMKM99]. This improves scalability and maintains fault-resistance behaviour.

4 RELATED WORK

The idea of distributing the intrusion detection system using agent software is not entirely new. However, most of the related work emphasize static agents instead of mobile ones. Applying mobile agent technology to IDS gives results to only a few research projects. In 1999, a project at The Information-Technology Promotion Agency (IPA) in Japan involved an Intrusion Detection Agent (IDA) System [AOTG99]. IDA is a classic host-based system that relies on mobile agents mainly to trace intruders among the various hosts involved in an intrusion. In the same year the project Micael [DQDCP99] pursued a more ambitious aim where the entire system is based on mobile agents. Nevertheless, only the architecture description has been presented and no details have followed so far. In 2000, an IDS framework based on mobile agents has been described in [BDSM00]. Unfortunately, detection is dealt with superficially. In 2002, [TAP + 02] describes an IDS designed as mobile application that roams the network to detect attacks and track intruders. IMA-IDS is a distributed intrusion detection system using mobile and intelligent agents. It is too tedious to detect a malicious action through the network, especially when considering multiple distributed events and even simultaneous one. Most of the current IDS assume that the environment is static, whereas in reality the environment is dynamic and unpredictable. So to detect, without mistakes, an intrusion and to make the appropriate decision at an optimal time a cooperation of different sensors and analysers is required. Thus, to apply agent mobility to avoid shortcomings of current IDS.

5 OUR SYSTEM : IMA-IDS

In this paper, we advocate the idea that in the security domain, especially when we are faced to the contemporary computer distributed environment, a mobile agents framework enhances the performance of IDS and even offers it new capabilities. We argue this point of view by explaining our system called IMA-IDS (Intelligent Mobile Agents for Intrusion Detection System). IMA-IDS main purpose is to achieve in automatically and real time the intrusion detection by mobile, intelligent and cooperative entities which are the agents.

5.1 Architecture Overview

In this section, we will introduce IMA-IDS architecture by recalling the main important properties to be verified by almost all IMA-IDS components:

- Information collection and filtering: Agents have to guarantee that collected information is of a good quality. The most important question here is how to evaluate information quality. We believe that information can be classified by:
- Information relevance: does the agent collect the important and useful information for the requested work? Is there any guarantee of information correctness? formal agent behaviour proof can be used here to ensure these properties.
- Information / event indication: remember that IDS are of two kinds: On line and post analysis IDS. In each case, agents must be able to report, as soon as necessary, any significant event for intrusion detection. We also believe that we need to define a kind of database rules that will allow agents to decide whether or not to report an event and to which agent (typically analyser agent) should this information be pointed up.
- *Information trust level*: one of the major drawbacks of actual IDS (signature or behavioural IDS) is positive and negative errors or missed attack signalisation. Ensuring that analysis agents have the appropriate information with all correlated events for analysis will help to reduce such errors. By assigning a trust level to each event depending on agent source, any Analyser agent can, in addition, decide whether or not to take in account such information.
- *Agent communication protocol*: in addition to the classic agent communication models (authentication, private or public channel), a specific protocol schema for crucial information communication will be introduced. In fact, suppose that an agent A asks for crucial information (M) possessed by the agent B. Suppose also that agent B

classifies this information as Top Secret so he asks for a hard cipher channel communication for M. If agent A considers M as just secret or public, agent B can even refuse communicating M to A, oblige A to upgrade to top secret class or communicate M to A and forbid any further communication of M by A. As presented by figure 1, IMA-IDS includes, in addition to the manager agent, three agents' categories :

- *Collector agent*: This kind of agent will be cloned and distributed throughout the network. This agent patrols the network and collects all the events occurring in the host to which it is related. Notice that the goal is to have specialized collector agent. The idea is that the collector agent will be interested in a set of event categories. So many collector agents can run on the same host, depending on applied analysis. It is also possible to merge collector agent abilities in to a single agent.
- *Correlator agent*: This is a particular agent that will hurry the specific information, called critical, and send it to the appropriate analyser agent without passing through the manager agent. The default communication protocol (presented later) is centralized. It supposes that a collector sends a kind of report to the manager agent. The manager will decide whether to dispatch data to the analyser or not. This communication model is inefficient for online detection since some crucial events must be taken in to account by the analyser as soon as they occur. That is why each correlator agent will use a set of rules that clearly specifies the crucial events, contexts and analyser agents concerned by an urgent reporting event mechanism.
- *Analyser agent*: Analyser agents are the engines of our solution. Several kinds of analysis such as classical signature detection, anomaly detection and a new security protocol analysis based on abstract interpretation as introduced in [EKOBY03] will be integrated. We are also working a be-haviour analyser that will use a kind of statistical model to define what can be concerned as "normal" system behaviour is also being developed.
- *Manager agent*: This agent gathers collected information and distributes it to analyser agents. This communication process does not allow online analysis. For this reason correlator agents can decide to communicate directly with analyser agents.

The administrator can distribute our IMA-IDS over any number of hosts in the network. Each host can receive any number of collector agents that monitor all events occurring in the host. All the collector agents report their results to the manager agent which transmits them to the analyser agents. The intelligence in our approach is attempted by communicating the critical events detected by the collector agents to the correlator agents. Notice that a critical event is any event liable to be part of a scenario of attack. The

correlator agents take charge of hurrying the critical events received from the collector agents and transmitting them to the concerned analyser agents. The analyser agents receive the events.

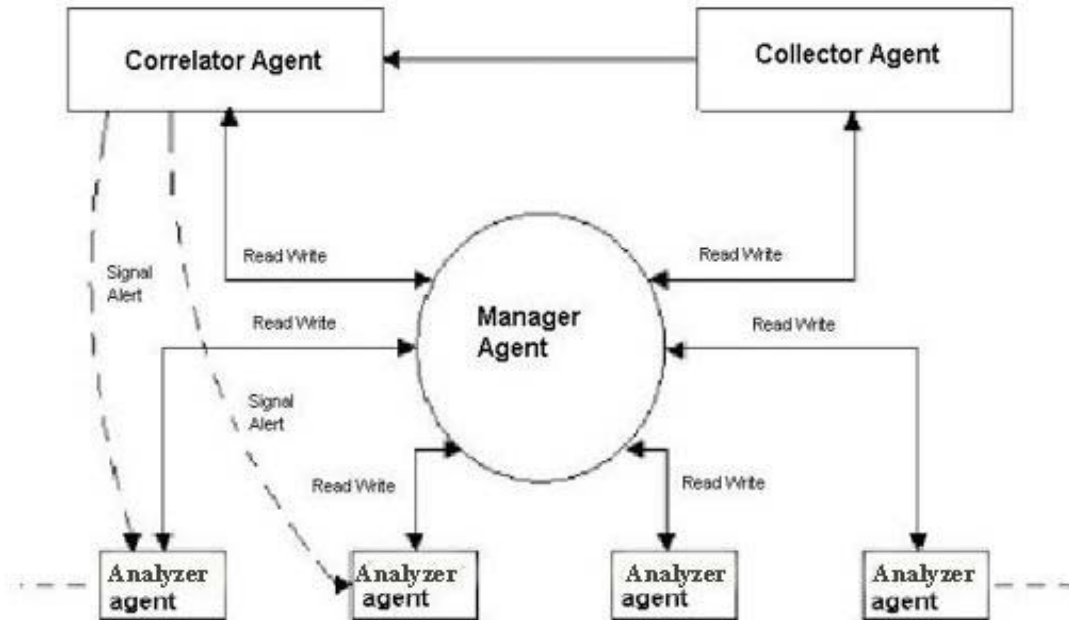


Figure IMA-IDS Architecture

From the manager agent and those from the correlator agent, perform a higher-level analysis and correlation [EKOBY03] (Anomaly and Policy detection). The analyzer agents report their results to the manager, and they generate alarms if they detect any anomaly. Agents in our system are cooperative because they respond to requests for event or critical events from other agents.

The administrator, according to its needs and via the external interface of the manager agent can stop the execution of these agents, send them to other locations and reactivate them. An agent in IMA-IDS can make a decision to dispatch itself. In others words, it can stop its execution, move to another location and restart its execution. Also, it can clone itself especially in the case of increased network loads.

5.2 Communication Mechanism

The transmission of messages between agents is a central part of the functionality of our IMA-IDS. In order to communicate, agents in our system are able to know all information about the other agents created and running in the net-work (their locations, their number, and their identifier) by sending a request to the manager agent. To offer these capacities, and to be able to update the agent list, the manager uses the following two agents:

- Registry Agent: being present on all hosts running agents, it maintains information about the agents running in the host.
 - IdsHost Agent: it keeps track of all created and running agents.
- Agent communication can be divided into two categories :
- Peer-to-peer communication (Monocast): The message sender must know the identifier of the receiver to be able to send a message.
 - Indirect communication (Multicast): a kind of a meeting between agents coming from different hosts. The basic idea is that agents subscribe to one or more multicast message list and implement handlers for these messages. Multicasting message provides a powerful way for agent interaction and collaboration [DM98].

5.3 Prototypical Implementation

To demonstrate the feasibility of our architecture a working prototype has been implemented. Our implementation is written using the IBM Aglet [DM98] platform and Sun's Java Development Kit. This choice is not arbitrary, it is made after a comparison study of nine agent platforms [EKBBGe03] during which the security mechanisms deployed by each platform to protect agents has been scrutinized. This comparison is summarized figure 2.

- Integrity: corruption of information.
- Denial of service: effecting availability of the host or the agent process.
- Secrecy: disclosure of information.

Table 2: Security features agent platform comparison

	Authentication	Access Control	Encryption	Code Verif.
Concordia	Simple Auth.	J.S.M	SSL	ByteCode
JADE	Simple Auth.	J.S.M	SSL	ByteCode
Aglet	MAC	J.S.M	Java	ByteCode
Voyager	Auth. Server	J.S.M	No	ByteCode
AgentTCL	PGP	RSA System	RSA PGP	SafeTCL
MAP	PGP	RSA	Weak + MD5	Secured Scheme
JATLite	Simple Auth.	J.S.M	SSL	ByteCode
TACOMA	PGP	Firewall	PGP	Firewall
Grasshopper	X509	J.S.M	SSL	ByteCode

Let us consider the most important security feature for our IMA-IDS. In fact, it is important to build our solution with an open platform that allows agent migrations, cloning agents and that support different communication protocols. Those features are common to almost all agent platforms. What seemed to be most crucial are security features. The following criteria are the main focus:

- Agent authentication: Trusted and untrusted Agent authentication systems.
- Resource Access Control: Implementing or not a discretionary access control.
- Supporting encryption/decryption facilities: Using standard protocol such as SSL and supporting commonly used algorithm (DES, AES, IDEA, RSA and so on).
- Code verifier (Agent action Verifier): Including a particular byte code verifier or allowing this kind of add-on easily. We mainly plan to develop a specific Java byte code analysis as in [EKB01] for verifying applet security properties. Another alternative is to use such verifier as introduced by Michiaki Tsubori [Tat99].

According to figure 2, there is no single best agent platform. Each platform offers some interesting services and features and suffers from some other weakness and limitations. Concordia, for example, seems to have the upper hand in the area of security but unfortunately, its excellent security provisions are not available with an evaluation licence. Thus, the best choice is Aglets.

5.3.1 Security Issues

The open-source nature of the Aglets system, which has been made available through the SourceForge opensource initiative [Kit02], allows us to intercept the adequate security actions performed by Aglets and to log all the

useful information such as the requested operation, its parameters, its outcomes and the aglet identity. Mainly, Aglets support the specifically required security mechanisms. The security model provided by Aglets supports the definition of security policies and describes how and where a secure Aglet system enforces these policies. Aglets have authenticated identities that are used to enforce the policies defined by authorities and to identify the program host or developer. Aglets framework uses an asymmetric (public_private key pair) cryptography system to exchange private keys between hosts. These keys are useful to ensure agent identities when they are transferred over the network. Thus, the agent code is signed and can be authenticated before its execution, keeping the host platform protected. In Aglets, permission is defined as the capabilities of executing aglets by setting access restrictions and limits on resource consumption. An abstract syntax for permissions in Aglets is based on JDK policy [Gon97] file definition.

5.3.2 Our Prototype

The prototype that we are testing is intended as a proof of the concept for the architecture, the security capabilities of the Aglets workbench, the communication model established in our IMA-IDS and the full mobility. The prototype implements the main structure and interaction between the agents, as well as security intended behaviour. This prototype is implemented in JDK 1.4.1_02 and uses the framework Aglets 2.0.2. The prototype has been distributed through a network of more than one hundred machines. One specific machine is used as a manager agent host. Let's recall that the prototype included for agent categories:

- *Manager Agent* : this is the scheduler of all the solutions. It uses a set of initialisation files describing the analysis-concerned domains, the IP addresses of the analysed hosts and also the agent platform domain signature.
- *Collector Agent* : collectors are for the moment composed of two categories. Event simulator agents and KCS sniffers [EKOBY03]. KCS sniffers are in fact implemented in C language and are not embedded in the agent platform because difficulties to capture network traffic in Java. The second kind of collector is, for the moment, an event generator used to simulate event generation through keyboard use as one example.
- *Correlator Agent*: Prototype correlators use empty rules in order to validate message exchanges between correlators and analysers.
- *Analysers Agent*: the prototype includes a signature-based analyser. A property propagation analyser is being developed as well. The initial

experimental results prove that the communication model seems to be well adapted to agent collaboration. In fact, by testing agent platform communication through more than 20 hosts, we didn't notice any information loss or delivery delay. Such results must also be validated by larger experimentation. The prototype was particularly efficient in defining an agent security policy. In fact, by creating some collector agent behind the authorised domain, we have been able to test the Aglet firewall functionality. Intrusion detection mechanisms can be not evaluated for the moment since analysis techniques are under development. We wish to have a complete evaluation platform in 3 or 4 months. Analysis models are based on compartmental analysis and statistical functions. The prototype includes an event generator and a set of rules to simulate correlator behaviour. During the actual experimentation, we notice that a set of conflicted rules can be a drain on correlator efficiency. In fact, we should add a set of filtering rules, as used in expert systems, in order to add a conflict solving phase when deciding which rule to be applied. For signature-based analysers, we believe that they will be no specific difficulties to define such rules that can totally rely on signature definitions. For property propagation, the use of abstract semantics as in [EK01] is ambiguous. Special care must be used to abstract parameter definitions. The compartmental analyser is more complicated. This is a work in progress and will be developed in a PHD project. For the moment we believe that we first must fix a set of "measurable" parameters that can be used as correct behaviour definition base. Opened communication port, segmentation messages, network charge are examples of useful parameters. The result obtained by the KCS project [EKOBY03] will be used as an initial set of input to analyse correct SSH/ SSL sessions in order to extrapolate a set of useful "measurable" parameters. The following figures (3, 4) illustrate the graphic user interfaces related to our IMA-IDS prototype :

- The Manager Agent Interface, it is the main console for controlling and using the prototype. The manger agent uses a set of parameters, described in input files, in order to create and distribute collector and correlator agents in the analysed domain.
- Reception of the events by the analyser Agent from the Correlator Agent, when a signalisation rule is verified, the correlator sends a message to the concerned analyser. The message includes the crucial information for the analysis. In further use, a specific secret communication channel will be created by adding a specific library to the aglet platform.

- Reception of an alert message by the Manager Agent from the Analyser Agent. When detecting an attack in an on line or post analysis model, the analyser sends a message to the manager that can substitute the network administrator by taking some corrective actions. The prototype did not include such abilities for the moment. We are working on defining a set of rules (in a kind of decision tree) to add such intelligent behaviour to our solution.

As we can see it, a specific console is used to show any agent action such as event signalisation or message exchange between analyser and manager.

6 CONCLUSION AND FUTURE WORK

Our purpose by implementing IMA-IDS prototype is to validate agent platform use for IDS. The chosen communication model, as proven by experimentation, offers a flexible and modular agent information exchange. After the validation of the global architecture, we are now working on event signalisation and correlation rules. For the moment, event collectors are based on specific C libraries because of their efficiency. We are studying a set of signature-based IDS to deduce a set of rules to be used by correlator agents. Future work deals mainly with Analyser agents. We aim to study a set of statistical and behaviour models in order to develop a new one for describing a "correct" and an "attack free" system behaviour. We believe that these models will be more efficient when coupled with other analyser such as signature-based systems.

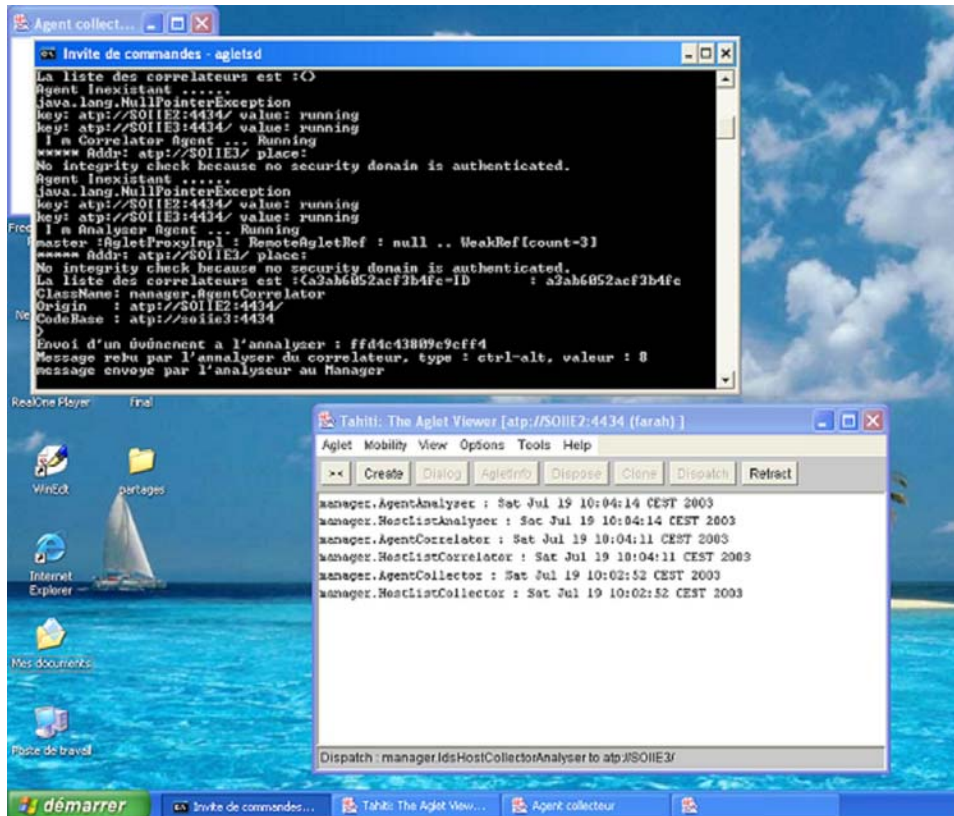


Figure 3: Reception of the events by the Analyser agent from the Correlator Agent

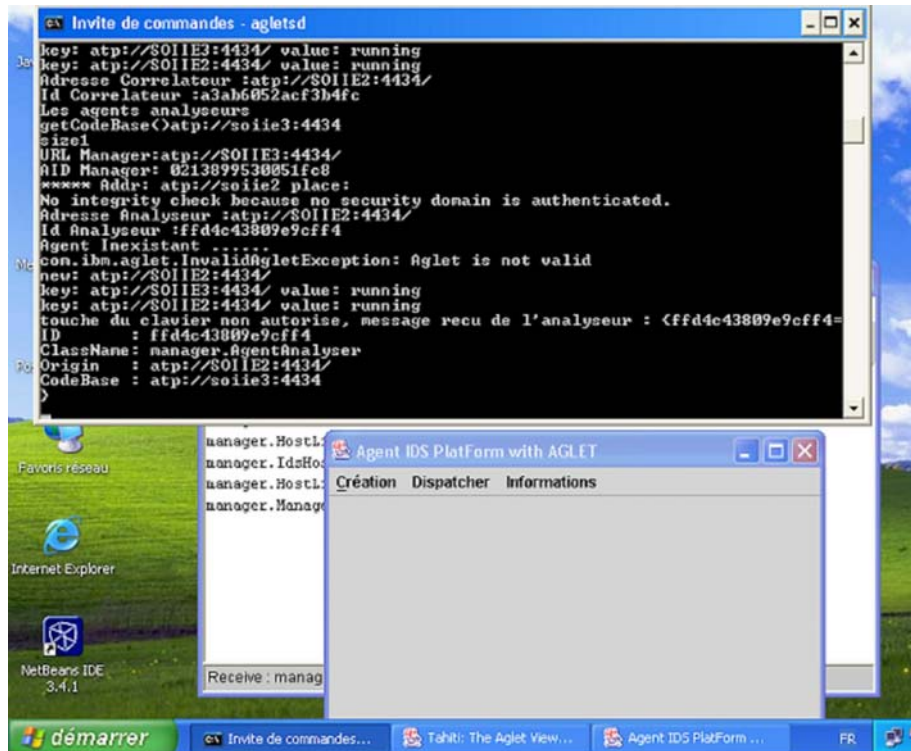


Figure 4: Reception of an alert message by the Manager Agent from the Analyser Agent

7 REFERENCES

- [AOTG99] M. Asaka, S. Okasawa, A. Taguchi, and S. Goto(1999), "A method of tracing intruders by use of mobile agents". In INET'99.
- [BDSM00] M. C. Bernardes and E. Dos Santos Moreira(2000), "Implementation of an intrusion detection system based on mobile agents". In International Symposium on Software Engineering for Parallel and Distributed Systems.
- [BGFI + 98] J. S. Balasubramaniam, J. O. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni(1998), "An architecture for intrusion detection using autonomous agents". In Proceedings of the 14th Annual Computer Security Applications Conference.
- [CLM + 99] R.H. Campbell, Z. Liu, M.D. Mickunas, P. Naldurg, and S. Yi. (November 1999), "seraphim: An active security architecture for active

network". Technical report, UIUCDCS-R-99-2167, UILU-ENG-99-1756, Urbana, IL 61801.

[DM98] B. L. Danny and O. Mitsuru (1998), "Programming and Deploying Java Mobile Agents with Aglets". Massachusetts, Addison Wesley, second edition, isbn 0-201-32582-9 edition, 225 p.

[DQDCP99] J. D. De Queiroz, L. F. R. Da Costa, and L. Pirmez. Micael(1999) "An autonomous mobile agent system to protect new generation net-worked application". In 2nd Annual Workshop on Recent Advances in Intrusion Detection.

[EK01] N. EL KADHI (2001) "Automatic verification of confidentiality properties of cryptographic programs".

[EKB01] N. EL Kadhi and P. Boury (2001) "Static analysis of java cryptographic applets". In Proceedings of ECOOP2001 (Budapest) Workshop on Java Formal Verification.

[EKBBGe03] N. EL Kadhi, F. A. Barika, E. Burstein, and K. Ghédira(2003) "Toward agent ids : Agents platforms security features study". In Proceedings of CSC 2003, Computer security Congress, Mexique.

[EKOBY03] N. EL Kadhi, J. Olivain, and J. Ben Younes(2003), "Kcs: A new ssh/ssl protocol analyser for key correlation system detection". In Proceedings of SCI.

[Ghe93] K. Ghédira. MASC (1993) "une approche Multi-Agents de problèmes de Satisfaction de Contraintes". PhD thesis.

[Gon97] L. Gong (July 1997), "java security architecture". Technical report, JavaSoft.