A secure protocol to protect ambient agents group

Nardjes BOUCHEMAL (1, 2) Ramdane MAAMRI (2)

University Centre of Mila, Algeria (1), LIRE Laboratory, Mentouri University of Constantine, Algeria (2)

Abstract -In a future vision of Ambient Intelligence – or AmI – our surrounding environment will integrate a pervasive, interconnected network of devices equipped with sensors, actuators and ambient agents. One important concern in AmI is the exact manner in which security and privacy are managed by a system formed mostly of agents with limited capabilities and resources. The goal of this paper is to present a cooperative approach to protect the ambient agents group. The key idea is based on inspection of new agents, guaranty, privileges and encryption keys. Each group has a specific agent called the representative. On the arrival of a new agent, members check it by sending test code and trivial data, and then the representative averages the privileges and confidence degrees awarded by the members. After integration, members must inspect the new agent, with possible revocation if they detect malicious behavior. Furthermore, we discuss voluntary and involuntary departure, where an agent can be stolen with encryption keys and sensitive data.

Keywords: Ambient Intelligence, Ambient Agent Group, Security, Privileges, Inspection.

Received August 15, 2012; accepted September 13, 2012

1. Introduction

Agents [3] have a set of characteristics which allow them to cover several needs for AmI environments, such as autonomy, reasoning, reactivity, social abilities, pro-activity, mobility, organization, etc. We talk about ambient agents [15].

In the design of ambient agents, inspiration was also taken from the human behavior and thinking. Ambient agent represents a user or an object and designed to be embedded on an electronic device. The set of ambient agents makes a group.

Ambient agents will operate on portable electronic devices with little autonomy energy and limited computing and storage capabilities that vary from one device to another. Any such restrictions should therefore be considered in order to guarantee the proper functioning of the group.

Ambient agent and multi-agent systems have been successfully applied to several social AmI scenarios such as education, culture, entertainment, medical domain, robotics and home domain [1, 9, and 10]. However, generally the design of a social system where the user is a key element, and where personal data can be manipulated, reveals ethical issues, security and privacy.

These constraints are even more problematic in ambient intelligence. Precisely, ambient agents systems present a risk if there is no control of different participants, or agents who require communicating [5]. The crucial question is how to ensure any confidentiality, integrity and privacy of exchanged messages and data if, at the outset, we are not sure that we communicate with the correct entity?

This article addresses security issues and presents an approach to guaranty that only trusted agents can communicate with ambient agents group. In section 2 we present some related work on protection of ordinary multi agent systems (MAS) and we discuss the need of new robust protection approaches for ambient agents systems or groups. Section 3 presents our proposal based on cryptography concepts, privileges and inspection of new agents. In section 4 we propose a scenario using the protocol. Section 5 presents some experimentation issues using Jade for ambient agents and AES algorithms with Java for encryptions keys. Finally, a conclusion summarizes the paper and future works.

2. Related Work

We discuss in this section the need of more powerful mechanisms which response to AmI requirements. But first, we present some approaches proposed to protect classical multi agent systems.

Mana and al in [6], proposed an approach to protect a society of collaborating agents, by making every agent collaborate with one or more remote agents running in different hosts. These agents act as secure co-processors for the first one. Likewise, these agents are in turn protected by other agents.

The Static Mutual Protection strategy can be successfully applied to many different scenarios. However, there will be scenarios where (i) it is not possible to foresee the possible interactions between the agents at development time, (ii) where the agents are generated by different parts.
Mana and al [7], proposed a new strategy called Dynamic Mutual Protection where each agent is able to execute arbitrary code sections on behalf of other agents in the society. Each agent includes a public part, an encrypted private part and a specific virtual machine.

Trusted Computing Platforms take the advantage of the use of a hardware element in order to provide a secure environment. These hardware elements are called TPM (Trusted Platform Modules) [8]. A TPM is a microprocessor with some special security features.

The Trusted Computing technology can be applied to protect agent systems. However, it is important to take into account the limitations of this solution. One of the main limitations is that this solution is designed for TPM-enabled devices. Because of this, it is not easy to apply this solution in ubiquitous environments, where there is a high level of device heterogeneity due to the different physical requirements of these devices.

The most interesting question to ask is: “in what way does AmI make a difference to security in this setting?” [13]. Where the computational infrastructure is composed of a very large number of computing devices with heterogeneous capabilities and under the control of different owners. Moreover, devices are highly mobile, so physical security is less provided that for static systems.

We have to keep AmI challenges in mind: energy devices management, devices physical theft or alteration, and identity stealing. Furthermore, users move to a free and arbitrary leaving or joining the group at any time.

Ambient agents will operate on portable electronic devices with little autonomy energy, limited computing and storage capabilities that vary from one device to another. So we must take into account ambient agents’ limitations and we mustn’t endow agents with complex cryptography concepts or historic data. This is why our proposal was based on cooperation and division of security tasks between agents of the group.


In this section we will present an approach to protect a group of ambient agents embedded in mobile devices and represents an object and/or a user. Each member of the group has a set of characteristics:

- **Id_Agent**: A unique identification within the group.
- **Priv**: Privileges within the group (set of permitted actions).

- **LiD**: The list of identifications agents of the group.
- **TrustDeg**: A degree of trust within the group: doubtful, malicious or trust
- **TmpSej**: The estimated staying time within the group.
- **DevRep**: Representation of the device resources: energy, memory and processor.

1) Each group has a specific and trust agent called representative with maximum of privileges, powerful device and with longer time to stay within the group.

2) Representative of the group has the instantly list of all group agents with their privileges, trust degrees and behavior reports.

3) To ensure the proper conduct of the group in case of attacks against the representative or an involuntary departure, we proposed two vice representatives having most privileges, residence time and powerful device.

4) When representative makes updates on agents status, new agents list, etc., It must send its changes report to vice representatives.

5) Members of the same group know each other and cooperate to complete various tasks. They share a Common Public Key (CPK) known only by group members and frequently changed by the representative.

6) Agents of the group have communication keys (Cki for an agent i) to establish secure channels with the representative. Cki are frequently changed by the representative.

### 3.1 Representative Roles

The representative supervises the group and cooperates with members to provide the proper functioning of the group and ensure a high level of security in case of new agents joining and/or leaving, moving elsewhere then rejoining the same group. Moreover, the representative verifies agents trust and inspects new integrated agents.

We will address following management functions: initialization of the group, arrival of a new agent and how to know that it is not malicious?
departure of an agent and how to avoid the case where it discloses the key communications, the common key or sensible data. we also adress inspection of new agents after integration in the group with the possibility of revocation in case of malicious behaviors. We will detail all these points in what follow.

a) Initialization of the group
To start the protocol, agents and devices must be set by a human administrator who furnishes an identity, a set of privileges and estimates residence time to each agent. He has to select the representative and tow vice representative to provide the case of representative departure.

Afterwards, representative establishes a secure channel with agents of the group, by sending a Communication Key to each one, (Cki to agent i). Then, representative sends Cki (CPK) to all members, where CPK is the Common Public Key of the group. Note that all agents are trusted and approved by the human administrator at the beginning of the protocol.

b) Arrival of a new agent
When the representative group receives an integration demand from an external agent, it sends its identity to group members. We study two cases before responding:

• First case: external agent is known by at least one agent from the group (guarantor): this is the case when both agents belongs to another group (see fig.1). The guarantor sends a confirmation message to representative who will delegates a third agent to test the external agent (see verification of an agent) and then avarages privileges and degrees of trust approved by the guarantor and the third agent.

• Second case: external agent is unidentified by any agent of the group; representative delegates a committee to inspect the external agent. The committee is a subset of agents with most privileges, resources, residence time and trust degree. Committee members check it using sandbox technique [4], then attribute a degree of trust: truthful, doubtful or malicious. The representative shall receive reports from committee members and calculates the average degree of confidence according to which privileges granted the external agent.

If the trust degree is malicious, the demand is rejected. Else, if the degree is doubtful, it is integrated and must be inspected by the other members.

We summarize the arrival of new agent in the following pseudo code.

c) Departure of an agent
Our goal in this section is not to provide fault tolerance solutions, but to treat the case where the agent is stolen, or divulges keys to malicious agents. In this case it presents a real danger to the group. For this, we consider two cases, as shown in leaving agent algorithm:

Voluntary departure. If the agent task is done, it sends
insufficient memory or no communication field (WiFi, GSM, etc.). In this case, the agent sends a standby message to representative (see voluntary algorithm departure).

Note that agents of the group have representation of the device such as energy resources, calculating and storage capabilities.

*Involuntary departure.* We treat in this section the case where the device is broken or stolen. We can conclude that if the agent does not respond to message, its residence time has not expired, and it did not send a message of standby.

The first agent who detects this anomaly sends a message to representative informing him that the agent is not responding. The representative must check the anomaly.

Upon confirmation that the agent stops responding, the first thing to do is to change keys CK(i) and CPK, as shown in involuntary departure algorithm.

An intruder cannot access the new key because it does not know the old group key neither communication key of the new member. Furthermore, renewal of keys is frequent.

d) *Departure of the representative*  
Before the representative leaves the group, the agent who replaces it must be trusted, powerful and having the right to be representative.

For this, we proposed two vice representatives chosen initially by the designer.  
So, representative delegates its vice representative, since it has all information needed to manage the group (the representative sent each time an update to the vice representatives). Before leaving the group, the old representative inform the other members about the new representative who changes keys and chooses its vice representatives among agents of the committee, since they have much capacities.

We also treat sudden departure of the representative due to an anomaly (failure or theft). The first member who detects that representative doesn’t respond, sends a message to the vice representative who confirms the anomaly, changes passwords and informs members.

e) *Inspection and update of member status*  
Members cooperate to keep track of new integrated agents in order to confirm their intentions.  
They send reports to the representative, who can make changes in status (fig.2), and privileges. At the end it sends all updates to members and establishes a *behavior certificate* that contains all behaviors of the agent.  
This certificate is known only by the representative and its vice representatives.

3.2 *The member role*  
In this section, we describe some tasks that members cooperate to achieve.

a) *Identification between Agents*  
Let’s consider two groups: Group1 and Group2. Agent1 belongs to both groups, so it has CPKs and identity list of both groups (CPK1, CPK2, LiD1 and LiD2).

Agent2’ from Group1 has id.Agent2’ as identity, LiD1(agents list of Group1) and CPK1 (Common Public Key of Group1). To identify each other, Agent1 and Agent2’ proceed to the following protocol. Note that Encrypt(ID.Agent1).CPK1 is the encryption of Agent1 identity using CPK1.

![Fig.2 Possible change of integrated Agent's status](image-url)
1. Agent1 sends Encrypt(Id_Agent1)_CPK1 to Agent2’.
2. Agent2’ calculates Decrypt(Id_Agent1)_CPK1 and finds that id_Agent1 belongs to LiD2.
3. Agent2’ sends Encrypt(Id_Agent2’)_CPK1 to Agent1. Agent1 calculates Decrypt(Id_Agent2’) _CPK1 and finds id_Agent2’ belongs to LiD1.
4. Agent2’ has already made unsuccessful demands to group4).

b) Examination of an unknown agent
The external agent is not tested by all agents in the group, but by the committee (chosen by representative) using sandbox technique [4]. The principle of sandbox is that committee agents send trivial code and data to external agent. Each one sends different code and data. After execution, external agent resends result, code and data. Agents of the committee check the code, data and results and then send their report with the trust degree and privileges to the representative who calculates averages (see Agent Verification Algorithm).

Algorithm Agent Verification; // by committee agents
Ext_Agent: New agent;
Begin
    Send (Code, Data) to Ext_Agent;
    Wait (Code, Data, results);
    If Code and Data not modified then
        Begin
            //Behavior analysis
            If 3 Doubtful Behavior then
                TrustDeg (New_Agent) = Doubtful;
                End if;
            If 3 Malicious Behavior and
                TrustDeg (New_Agent) = Malicious;
                End if;
            If all Behavior trusted then
                TrustDeg (New_Agent) = Trust;
                End if;
            Else //code or data modified;
                TrustDeg (New_Agent) = Malicious;
                End if;
            Send(Report) to representative ;
        End.
End.

4. Experimentation
In this section we describe some experimentation issues using JADE platform (Java Agent DEvelopment framework) [2].

4.1 Experimentation concerns
Jade is a platform fully coded in Java. For that reason we choose to implement the proposed encryption keys (CPK, Cki) also in Java, using the AES (Advanced Encryption Standard) encryption algorithm [11, 12]. AES is a specification for the encryption of electronic data. It has been adopted by the U.S. government and is now used worldwide.

We use basic AES encryption/decryption functions in Java, because of limited capacities of agents.
To implement complex actions in JADE, we use behaviour composition by building simple behaviours. The basis for this feature is provided by the CompositeBehaviour class included in the jade.core.behaviours package.

Messages exchanged by agents are instances of the jade.lang.acl from ACLMessage class, which represents ACL messages. It contains a set of attributes as defined by the FIPA specifications (fig.4) [14].

4.2 Agents and Containers
In what follow, we describe some principles of the proposed scenario agent’s experimentation:
1. We create two containers, each one represents a group: main container comprises AgentChefMed, AgentMed1, AgentMed2, AgentNurs and AgentDriv. Container1 has AgentChef, Agent1, Agent2 and AgentMed1 (Fig3).
2. Each container has a representative agent (AgentChefMed for Main container, and AgentChef for Container1).
3. We suggest that AgentMed1 is a mobile agent in order to have an occurrence in Main container and in container1. So, Main container and container1 have AgentMed1 as a common agent. If AgentChef from container1 wants to communicate with AgentMed1 from Main container: In this case, AgentChefMed, the representative of Main container asks all agents of the container. A positive response comes from AgentMed1, which is the guarantor of AgentChefMed.
5. Conclusion

We addressed in this paper an approach to protect ambient agents group, representing dynamic devices, objects or users.

If an external agent requests communication with agents of this group, at least one agent belonging to the group must know it. Otherwise, representative and members cooperate to verify if this agent is malicious or not.

Even after integration, members inspect the new agent with possible changes in status or revocation in case of malicious behavior. Our goal is to guaranty that only trusted and known agents can be integrated to the group.

We have detailed the protocol and introduced a set of encryption keys: CPK (Common Public Key) shared known by all members of a group, Cki (Communication Key for Agent i) sent by the representative to an agent i. Finally, we presented an experimenta-

tion using JADE framework to represent Agents, and Java AES algorithm to generate different keys.

Our ongoing work is to simulate our approach with an AmI platform and apply it in real society fields like health or homecare domain.

REFERENCE


[14].www.fipa.org/repository/aclspecs.html