THREATS TO THE TRUST MODEL OF MOBILE AGENT PLATFORMS

Michail Fragkakis and Nikolaos Alexandris
Department of Informatics, University of Piraeus, 80 Karaoli & Dimitriou St, 185 34, Greece

Keywords: Mobile Agents, Security, Threat Scenarios, Trust model.

Abstract: This paper intends to present the ways in which mobile agent architectures address important threats concerning their trust model, by comparing the behaviour of four major mobile agent platforms. The conclusions drawn are then used to point out deficiencies of current technology and highlight issues that need to be addressed by future research.

1 INTRODUCTION

Agent systems are a special category of software, which is designed to carry out a specific task on behalf of an entity. A mobile agent should be able to travel between different platforms, possibly carrying along all its data. As a research field, they are closely connected to artificial intelligence and distributed computing (Rempt 02, Vlassis 07). As a result, Multi-Agent Systems (MAS) bear the same security issues met at any mobile code systems application. Despite the employment of various security features by most systems, there is lack of a generally adopted security standard that covers all their operation aspects (Fragkakis 07 a, b, Roth 04).

This paper continues our previous research on mobile agent trust and security models (Fragkakis 07 a, b). In our present work we will focus on the trust model of the MASs previously examined. We will examine the behaviour of each system on specific threats involving their trust assumptions. Furthermore we will attempt to determine the suitability of each trust model for open environment operations. In contrast to our previous research, we will use a different method of analysis on the same set of agent systems. This way we will be able to support the validity of our past conclusions on security deficiencies. This would strengthen our position that there is a general lack of standardization on trust and security models. Finally, based on our findings, we will attempt to point out certain areas, where security technology may be insufficient and we will set goals for future research.

2 THE MOBILE AGENT SECURITY CONCEPT

Multi-agent systems involve communication between agents, as well as a certain degree of mobility. For certain tasks it may be vital that an agent is moved, along with its computations, across a wide area network. As a consequence, agents must be protected against malicious platforms, aiming to tamper with their operation or (possibly confidential) data. On the other hand, platforms providing agent hosting services must also be protected against malicious agents. Finally, it is possible for an agent to launch an attack against another agent in the same, or even, in a different platform.

Agent developers employ a number of security mechanisms in order to address issues, which include authentication, confidentiality, integrity and monitoring (Ameller 04, Rempt 02, Posland 02, Zhang 01). In addition to these, the fact that an agent may have to operate in a changing environment introduces the concept of trust. Trust resembles the social human behaviour for evaluating risk, driven by the need for cooperation through communication and interaction for the accomplishment of a specific purpose (Posland 02, English 02, Zhang 01). It involves assumptions about the security or even malice of several entities comprising a MAS. Although making trust assumptions is necessary, sometimes it may lead to mistakenly considering a party to be secure or legitimate, when it actually is not. In this case, using the latest and most advanced security mechanisms is pointless.

Fragkakis M. and Alexandris N. (2008).
THREATS TO THE TRUST MODEL OF MOBILE AGENT PLATFORMS.
In *Proceedings of the Third International Conference on Software and Data Technologies - PL/DPS/KE*, pages 175-180
DOI: 10.5220/0001884301750180

Although agent technology can be very useful, their adoption is considered to be slow (Roth 04). The fact that there is autonomous and intelligent software running on a foreign host without a standard security model may restrict their applications. As a result, current agent systems are usually implemented within a limited network, which is considered to be secure.

3 THREAT SCENARIOS

We will attempt to investigate the behaviour of mobile agent systems on threats concerning their trust model. Our purpose does not include generic Information Security issues, which also happen to concern mobile agents, like communications security, or access control violations. Rather, we will only focus on agent-specific threat scenarios, more specifically on scenarios of trust model insufficiency. As such, we will attempt to examine the behaviour of the agent systems in the following scenarios:

- Malicious behaviour of an authenticated/ trusted agent. The trust level attributed to an agent usually derives from the identity of its owner (Fragkakis 07 a, b). However, belonging to a legitimate user, does not prevent an agent from exhibiting malicious behaviour. This type of attack may be realised either intentionally by the so far trusted user or by a third party committing identity theft of the trusted user. Whichever the cause of this may be, we would like to investigate the response of the agent system, as well as the possible consequences on its operation.
- The threat of an agent platform being, or having become malicious. An execution platform is usually by default considered to be trusted by the agent (Fragkakis 07 a, b). However, it is possible for a platform to launch an attack on the executing agents. This can happen either because the integrity of initially legitimate platform has been compromised, or because the platform was malicious in the first place. We would like to determine the extent to which the trust and security models of the MASs under review are capable of addressing this kind of threat and assess possible consequences.
- Agent operation in an open environment. Although this may not be considered to be specific threat, we feel that it is important to examine the efficiency of the trust model for operation within environments, where there is no common security administration, like the Internet. We would like to determine the degree to which this kind of operation is supported.

4 MOBILE AGENTS BEHAVIOUR

Using the proposed threat scenarios we will attempt to determine the adequacy of the trust models of four mobile agent platforms: Grasshopper, Cougaar, Aglets and Havana. These systems were selected because they have been developed taking into account the security needs of mobile agent technology, which becomes apparent from the fact that they employ a wide range of security features (Fragkakis 07 a, b).

In particular, Grasshopper and Aglets are general purpose platforms whose trust and security models have drawn considerable attention in the past (Altman 01, Fischmeister 01, Giang 02, Baumer 99, Kadhi 03, Vigna 02). Cougaar is a special purpose agent system, initially funded by DARPA, which was designed for the high risk environment of warfare conditions. Finally, Havana is a special purpose shopping agent system designed to provide a totally trusted environment ruled by a business contract.

Another reason for which the particular systems were selected, is because they are diverse enough to represent different types of applications of mobile agent technology. Furthermore, they are among the MAS projects whose internal operation and security mechanisms are sufficiently documented.

In the following sections, we will examine the behaviour of the four MAS systems in the afore-mentioned threats.

4.1 Grasshopper

The Grasshopper agent system is developed by GMD FOKUS, distributed by IKV++ and is intended for e-commerce applications, information retrieval, telecommunication services and mobile computing.

Internal security relies on mechanisms provided by the Java Virtual Machine (JVM) (Giang 02). It aims to protect the platform and its resources from malicious agents, as well as to provide protection at agent-to-agent interactions. Grasshopper agents inherit their owner's access rights. As such, if a user

is trusted, their agents are also considered to be trusted (Fischmeister 01, Giang 02, Altman 01). In the case of a trusted agent behaving maliciously Grasshopper relies on the security features of the JVM. Although this may protect to a degree other agents running on the same platform, the platform itself is exposed. The malicious agent, having already been authenticated, has been granted access to system resources. Depending on its programming, an agent may try to compromise the integrity of the platform, access sensitive data or strain system resources.

On the other hand the JVM is not designed to protect an agent from a malicious platform. In this case the integrity of the agent and its data can be easily compromised and even manipulated, since Grasshopper's security model doesn't address this type of threat (Fischmeister 01, Giang 02).

Grasshopper defines a domain (Region) within which agents operate (Giang 02, Baumer 99). The security level in all platforms is globally defined by the Region Registry in a centralised way. Despite the fact that a Grasshopper Region is trusted by entities within its boundaries, it cannot be regarded as an open environment in any case. This is apparent, as operation among foreign Regions is practically not supported, restricting its efficiency on open environments.

As far as the overall security level provided, successful attacks have been achieved against the Grasshopper MAS (Fischmeister 01). These involve trusted code base attacks, graphic user interface attacks, system properties attacks and policy system attacks, which make use of trusted Java classes and incomplete or unsecured methods.

4.2 Cougaar

Cognitive Agent Architecture (Cougaar) was developed by Cougaar Software Inc as part of the DARPA ALP and UltraLog programs. It is designed to meet the high security, robustness and scalability standards needed in times of war (Feiertag 04a). Cougaar, as well, is based on Java.

The dominant characteristic of Cougaar is its Security Adaptive Engine. This engine initially identifies the important assets of each application. It then creates scenarios of potential attacks and their purpose and identifies vulnerabilities of the applications. Finally, it produces a list of suitable countermeasures and their cost and enforces a set of countermeasures, which balance the level of security with the total cost. The Security Adaptive Engine is considered to be necessary, because enforcing the maximum level of security on all applications has a serious impact on the performance of the platform (Cougaar 04, Feiertag 04b).

Depending on the type of application built on top of Cougaar, security features may be enabled or not. The trust model is determined by the authentication requirements set by each application. In general, an authenticated agent is considered to be trusted and the platform is considered to be trusted by the agents. Cougaar includes a Monitor and Response framework, which collects and analyses data from various entities, in order to detect possible attacks and dynamically adjust the level of security (Rosset 04).

As a result of this approach, the Cougaar hosting platform is protected from malicious agents by the Java Sandbox, as well as this intrusion detection mechanism. Depending on the security policy set and the behaviour of the agent, measures can be taken to ensure the integrity of the platform (Feiertag 04b).

As for the threat of a malicious platform, Cougaar makes no specific provision. Most security features address agent based threats or external threats. However, features like mutual authentication, as well as operation in the closed environment of military applications, reduce the likelihood of the specific risk.

As a military application, Cougaar features centralized administration in a domain (Society) through the Society Authority, which is responsible for adjusting the security policy in each server (Node). As a result the overall level of security increases, since agent operations happen in a controlled environment. However, in a similar manner to the case of Grasshopper, the trust model is unsuitable for open environment applications.

4.3 Aglets

IBM Aglets Workbench was developed initially at the IBM Tokyo Research Laboratory (Ferrari 04), and currently is an open source project. It is a general-purpose mobile agent platform designed to provide an easy programming model, reliable communications and adequate security features. It is intended for use over the Internet and it is implemented in Java (Ferrari 04, Oshima 98).

Aglets attempts to ensure platform integrity from malicious agents in two ways: the JVM native mechanisms, as well as an intrusion detection tool. This tool relies on audit information analysis to ensure platform integrity (Vigna 02, Ferrari 04, Fischmeister 01). The use of the intrusion detection

mechanism mitigates the insufficiency of the JVM standard mechanisms against attacks by authenticated/trusted agents.

In Aglets all agent system components within a domain are considered to be trusted and use a shared secret key for authentication and communications integrity (Fischmeister 01, Karjoth 97). Bearing this common key, the platform is de facto considered to be trusted by the agent, irrespective of the possibility of being malicious. As a self-protection mechanism, each agent employs a proxy, which isolates the agent itself from the other entities in the system. This security feature, although useful for interactions between agents, may not be effective in the case of a malicious platform, since the proxy itself runs on the platform.

The Aglets MAS is designed for use over the Internet as a medium, but this should not be mistaken for operation over an open environment. In fact, Aglets entities operate only within a well defined Domain. In particular, they share the same Domain-wide security policy and the afore-mentioned secret key, both issued by the central Domain Authority. This common security administration, as well as the existence of a single shared key, imply that Aglets is oriented for operation among trusted entities, sharing some type of common background. As such, operation on an open environment seems to be outside of its scope.

4.4 Havana

Havana is a mobile agent system developed by the University of Guelph, Canada. It aims to provide the execution platform, as well as a business model for integrating mobile agent technology to existing web servers (Mahmoud 04, 05, 06). Havana proposes a closed architecture where all entities are bound to each other with a profitable business contract (Mahmoud 06). The agents are in fact shopping agents, which are introduced to the world through a Gateway. Their migration takes place between the Gateway and the Business Servers of various online retailers.

A dominant concept in Havana is the trust model. Any malicious behaviour during interaction results in the breaking of the business contract. As Havana is based on Java, the platform is protected by malicious or malfunctioning agents by the Java Security Manager, without implementing any additional mechanisms. However, the real protection Havana offers is the business contract between the entities. Consequently, a malicious agent may compromise the platform integrity, but this will have direct impact on its owner.

The trust model of Havana considers an authenticated platform to be trusted. Of course, it is possible for a previously legitimate platform to become malicious. In this case, it is technically possible to compromise an agent's integrity, and even manipulate its shopping activity. However, Havana offers strong non-repudiation services. As soon as an agent completes its operations on a server, it transmits the results back to the Gateway. This is done to detect any alterations from future hosts, which would again result in the breaking of the contract. Again, in this threat the true protection comes from the real world consequences on the entity that broke the business contract.

Havana is suitable for operations across open networks, with the various entities involved having independent security administration (Mahmoud 04, 06). However, the Gateway is responsible for the registration and authorization of shopping users, agents and online stores, allowing their interaction. Therefore, it can be considered to be a central security enforcer which renders the Havana world a closed environment. Accordingly, the Havana world is considered to be trusted, creating in this way a beneficial relationship where no unauthorized entity may enter.

5 COMPARISON RESULTS

Having reviewed the behaviour of the trust models of Grasshopper, Cougaar, Aglets and Havana MASs, we will proceed to their comparison. Since all systems examined are based on Java, they display a number of similarities on their security features. However, our comparison will focus on the behaviour of the systems in the threat scenarios listed in Section 3. A summary of the comparison is shown in Table 1.

As far as the threat of a malicious agent is concerned, all systems make use of the Java Sandbox, which is designed for untrusted code execution (Giang 02). The isolated execution environment it provides can protect the platform or other agents from establishing direct contact with the malicious entity - at least to some degree. However, a trusted agent exhibiting malicious behaviour poses a serious threat, since it is granted access to system resources and information, even with the use of Java Sandbox. Cougaar, Aglets and Havana employ additional features to counter this threat. In particular, Aglets and Cougaar use built-in intrusion

detection mechanisms which monitor the agents for suspicious behaviour. Havana, on the other hand, uses its non-repudiation Services, offered by the Gateway, to allocate liability for malicious behaviour. Although this approach deviates from the ones adopted by the other systems, it can be very effective for the trade oriented environment of Havana, where all entities are bound by a real-world business contract.

Table 1: Comparison results.

	Malicious Agent	Malicious Platform	Open Environ-ment
Grass-hopper	Java Sandbox	Agent is exposed	Operates within a defined Region
Cougaar	Intrusion detection (Monitor and Response), Java Sandbox	Attack is possible but unlikely to occur due to contained environment	Operates within a defined Society. Initially not intended for non military applications
Aglets	Intrusion detection, Java Sandbox	Proxy (limited effectiveness)	Operates within a defined Domain
Havana	Real world consequenc es, due to business contract, Strong non repudiation Services by the Gateway, Java Sandbox	Attack is possible but brings real world consequen-ces, due to business contract. Strong non repudiation Services by the Gateway	Operates in a closed environmen t for trade operations only. All entities are bound by a real world business contract.

The threat of a malicious platform is dealt with varying ways among the systems. Although all MASs dictate that an authenticated platform is trusted by all the agents of the same domain, they may employ additional mechanisms to protect the integrity of the agents. In particular, the trust model of Grasshopper relies only on this assumption to ensure the integrity of an agent, which can be very risky if this threat is actually realised. Likewise, Cougaar relies on the same assumption, so an attack may be possible. However, the risk is mitigated by the closed nature of its network environment. Since Cougaar was developed for military applications, it relies on the use of mutual authentication between entities, closed networks, tamper-resistant hardware and restricted communication channels. All these features leave minimal margin for the existence of a malicious platform. Aglets on the other hand, attempts to address this threat through the use of a proxy, built inside the agent. The effectiveness of this approach may be limited, since it is possible for the platform to tamper with the proxy. Finally, Havana does not employ any security mechanism to protect an agent from a malicious platform, but assumes that the platform can be trusted. On the other hand, the business model of Havana ensures that there is no gain for a platform attacking an agent, even if there is such an intention. Even if this attack is realised, the non-repudiation services, in conjunction with the real world contract that binds all entities, ensure the amendment of the abused party.

As far as the third scenario is concerned, all systems follow a similar approach. All MASs operate within a defined domain, with some sort of central security administration. Specifically, in the case of Havana, besides the security control applied by the Gateway, the domain entities are also bound by a real-world business contract. Although the systems may be deployed over an open network, their trust models require a central security administration, closed to foreign entities. We cannot say that any one of the systems examined supports practical operation in an open environment, where no common security administration exists.

6 CONCLUSIONS AND FUTURE WORK

Based on our comparison of the four systems, we were able to draw some general conclusions about the trust models of MASs. Although the architectures and the use of the selected systems differ, we identified important similarities in their trust models.

In all systems the trust model assumes that an authenticated agent that belongs to a trusted user is considered to be trusted by the platform. Furthermore, the platform is by default considered to be trusted by the agent. We argued that these two assumptions can potentially lead to serious security breaches, especially in the case of a malicious platform. Although it is possible to employ self-protection mechanisms within an agent (Ametller 04), the threat of a malicious platform is very difficult to overcome. We have also argued (Fragkakis 07 a, b) that mobile agent security issues

are more difficult to resolve because there is no commonly accepted framework covering all the aspects of a trust and security model. These threats, along with the lack of a standardized way to address them, are possible reasons for which MASs rely on domains with central security administration for their operation. In such environments the risks concerning malicious entities are mitigated.

On the other hand, we have already stressed that this kind of usage can be very restricting and hinders the adoption of mobile agents, especially for large-scale applications. It is true that, in order to operate securely, an agent system requires a trusted environment. This is achieved either by operating in a completely closed environment, or by employing a separate trust authority to guarantee the legitimacy of the entities in a MAS.

Havana displays the interesting concept of merging the trust-granting authority with the real-world contracts. This combination ensures that in the case of malicious behaviour there will be real-world repercussions on the party behind the malicious entity.

As a target of our future research, it would be useful to take this concept outside the business scope of Havana and create MAS-independent trust granting authority, expanding the trust and security models in the real world. Another interesting idea for this trusted third party would be to offer non-repudiation services in combination with insurance services to registered members. The existence of this service could help overcome the lack of trust in open environment like the Internet, and could be incorporated in a complete trust and security model for the operation of mobile agents.

REFERENCES

Fragkakis, M., Alexandris, N., April 2007. Comparing the Trust and Security Models of Four Mobile Agent Platforms, *RCIS'07* (a).

Fragkakis, M., Alexandris, N., 29 - 31 August 2007. Comparing the Trust and Security Models of Mobile Agents, *IAS 07, Manchester, UK* (b).

Ametller, J., Robles, S., Ortega-Ruiz, J. A., July 19-23, 04. Self-Protected Mobile Agents, *AAMAS'04, N. York.*

Rempt, B., Mertz, D., July 2002. Distributing Computing - Cooperative Computing with Mobile Agents *Intel Developer Services*., Available: http://gnosis.cx/publish/programming/dc4.pdf

Vlassis, N., 2007. *A Concise Introduction to MAS and Distributed AI*, pp 1-6.

Poslad, S., Calisti, M., Charlton, P., 2002. Specifying Standard Security Mechanisms in Multi-Agent Systems", *AMAS 2002, Bologna*, pages 122–127.

Zhang, M., Karmough, A., Impey, R., 2001. Adding Security Features to FIPA Agent Platforms, Available: www.elec.qmul.ac.uk/staffinfo/stefan/fipa-security/rfi-responses/Karmouch-FIPA-Security-Journal.pdf.

English, C., Nixon, P., Terzis, S., McGettrick, A., Lowe, H., 2002. Dynamic Trust Models for Ubiquitous Computing Environments, *UBICOMP `02.*

Roth, V., July-2004. Obstacles to the adoption of mobile agents, *MDM'04.*

Giang N.T., Tung, D.T., Jun 2002. Agent Platform Evaluation and Comparison, *II-SAS, Pellucid EU 5FP IST-2001-34519 RTD.*

Ferrari, L., Oct 2004. The Aglets 2.0.2 User's Manual, Available: http://aglets.sourceforge.net/.

Oshima, M., Karjoth, G., Ono K., 1998. Aglets Specification 1.1 Draft, http://www.trl.ibm.com/aglets/

Fischmeister, S., Vigna, G., Kemmerer, R.A., Dec 2001. Evaluating the Security Of Three Java-Based Mobile Agent Systems, *MA 2001*, 31-41 LNCS 2240, Springer.

Karjoth, G., Lange, D.B., Oshima, M., Jul/Aug 1997. A security model for Aglets *IBM Res. Div., Zurich, IEEE Internet Computing, Vol 1, Issue: 4, pp 68-77.*

Vigna, G., Cassell, B., Fayram, D., 2002. An Intrusion Detection System for Aglets, *I. Conference on Mobile Agents*.

Mahmoud, Q.H., Yu, L., 2005. An Architecture and Business Model for Making Software Agents Commercially Viable, *HICSS 2005,* Track 3, Vol 03.

Mahmoud, Q.H., Yu, L., May 2004. Havana: a mobile agent platform for seamless integration with the existing Web infrastructure, *Canadian Conference on Electrical and Computer Engineering*, Vol 3, 2-5 pp 1257 – 1261.

Mahmoud, Q.H., Yu, L., 2006. Havana agents for comparison shopping and location-aware advertising in wireless mobile environments, *ECRA 06*, 5(3): 220-228.

Kadhi, N., Burstein, E., Barika, F., Ghedira, K., March 2003. Towards Agent IDS: agent platform security features study, *Congreso de Seguridad 03*.

Altmann, J., Gruber, F., Klug, L., Stockner, W., Weippl, E., 2001. Using Mobile Agents in Real World: A Survey and Evaluation of Agent Platforms, *W. on Infrastructure for Agents, MAS, and Scalable MAS at Autonomous Agents `01*.

Baumer, C., Breugst, M., Choy, S., Magedanz, T., October 1999. Grasshopper – A Universal Agent Platform Based on OMG MASIF and FIPA Standards, *Ottawa MATA'99*, World Scientific Publishing, , pp. 1-18.

Feiertag, R., Rho, J., Rosset, S., 2004. Engineering Security in a Multi-Agent System, *Cougaar Software*.

Feiertag R., Rho, J., Rosset, S., 2004. Using Security Mechanisms in Cougaar, *Cougaar Software*.

Cougaar version 11.4, 23 December 2004.

Rosset, S., 2004. Cougaar Security Services, *Cougaar Software*.