Security of Mobile Agents in Distributed Java Agent Development Framework (JADE) Platforms

Timo Bayer and Christoph Reich
University of Applied Science Furtwangen, Germany
Email: {timo.bayer, christoph.reich}@hs-furtwangen.de

***Abstract*—Mobile Software Agent has become increasingly interesting as a basic software technology in the field of Internet of Things, like Smart Cities, Smart Home, Industry 4.0, etc. since it offers dynamic adaption, autonomous actions, flexible maintenance, parallel processing, and is tolerant to network faults. In particular, it is very challenging to guarantee security because of characteristic features such as mobility and autonomy. This paper addresses specific security requirements for mobile software agents and possible threats for agent system operations in the context of Java Agent Development Framework (JADE) platform. The main objective of the paper is to show existing vulnerabilities and security gaps by analyzing the security of agent platform JADE, showing existing improvements of the confidentiality of software agents merging from one agent platform to another and introducing trusted agents and their implementation in JADE.**

***Keywords*-JADE security; mobile agent security; agent migration; agent confidentiality; agent trust**

I. Introduction

One of the big challenges for the design and the use of Multi-Agent Systems (MASs) is the complexity caused by their dynamic, autonomy, and high decentralization nature. MASs are often operated in different networks, that belong to various organizations, and consist of components with different responsibilities (Section III). Consequentially, there is a wide range of attack vectors and a number of difficulties arise during the integration of appropriate protective measures. A particular challenge is the transfer of software agents, which can merge from one platform to the other. Obviously it is necessary to ensure a secure migration of the mobile agents and to protect the underlying agent system against potential malicious agents. The agents may operate in different MAS platform providers. In order to protect the agent owner and the receiver platform, it has to be ensured their integrity and adequate trust level during the migration process. A multi provider distributed MAS approach for auditing data access policies between multiple hybrid cloud environments, for instance, has been proposed in Ruebsamen et.al [1]. There, for example, agents migrate from platform to platform to verify the compliance of correct data access along the provider chain. The objectives of the paper are to show existing vulnerabilities and security gaps by analyzing the existing security implementations of agent platform JADE, show existing improvements of the confidentiality of software agents merging from one agent platform to another and introducing trusted agents and their implementation in JADE. The paper is structured as follows: In Section II we presents the related work. In Section III-A, we discuss the requirement for multi-agent systems to have a security evaluation while considering the characteristics of mobile agents. This paper focuses on Java Agent Development Framework (JADE), its existing security implementations Jade Security (Section III-B), a security analysis with regard to the specific requirements of mobile agents, and to point out the security gaps of JADE (see Section III-C). Finally, additional security measures, especially in the field of confidentiality and trust are discussed and adapted for an appropriate use in the context of JADE in Section III-D and Section IV. Section V concludes the paper.

II. Related Work

A detailed discussion about agent technology, particularly with reference to the specific features, technical characteristics, possible applications and the arising potential of this technology can be found in [2]-[3]. To increase the field of application, big efforts have been made in designing and developing suitable security measures.

Ahila et al. reported in [4] about the security requirements for mobile agent systems and pointed out some existing security threats as well as general security enhancements. Particularly, security concepts protecting the confidentiality of mobile agents have been introduced. In this paper, some of the illustrated security concepts are being adapted to the applications of the framework JADE.

Bürkle et al. described in [5] existing security implementations in agent framework JADE and pointed out some vulnerabilities allowing attacks to reduce the availability of agents or the agent system by using targeted denial of service attacks. Furthermore, they pointed out some limitations in the security implementations of the agent framework JADE, including some restrictions on the permission model for mobile agents. The described vulnerabilities conform with the results shown in this work.

Vila et al. also evaluated in [6] the existing security implementations of JADE and reported some additional ameliorations concerning security, being adopted to a concrete application scenario. The analysis focussed on encrypting

 ISBN: 978-1-61208-547-0

the communication between agents and on ensuring the availability of the agent system.

Piette et al. described in [7] the use of mobile agent systems for configuration, deployment, and monitoring of distributed applications in the domain of Ambient Intelligence. They focus on the ability to enhance privacy by hiding information using the agent architecture. Such a scenario includes sensitive information and therefore requires the capacity to protect the agent system. The shown scenario clarifies the necessity to consider the particular security requirements of mobile agents.

Geetha and Jayakumar evaluated in [8], the general security requirements for mobile agent systems and existing security measures. Especially, they pointed out some weaknesses in the field of protecting the carried data of mobile agents. To mitigate this issue, they implemented a trust and reputation management to provide a secure path for mobile agent data protection. This work is similar to the approach presented in Section IV, but focused more on the data carried by the agent rather than their general trustability.

Dong et al. [9] argue that due to the open nature of communication channels in networked multi-agent systems, the network is vulnerable to various malicious cyber attacks. A specific edge-bound content modification cyber attack has been designed which compromises and destabilizes the multi-agent systems. The paper describes attack detection schemes and it proposes an attack mitigation scheme. Such an approach can also be used to enhance the general security threats shown in Section III-C.

Gengarajoo et al. introduced in [10] an approach to ensure trust of cooperating agents. The proposed trust evaluation model determins the trust of an agent, based on experience gained from interaction among agents. In addition to our approach presented in Section IV, it relies on further runtime information. Therefore, the two approaches can be combined to ensure trust among the whole agent life-cycle.

III. Security in Multi-Agent Systems

Multi-Agent Systems (MASs) provide an environment for multiple interacting intelligent agents, that cooperate to solve problems. There are several key characteristics, such as adaptation, scalability, autonomy, communication between agents, etc. (see [4]). Frameworks, like JADE [11], exist to support the agent system development, operation, and maintenance. Next to the MASs characteristics, JADE also supports, mobile agents, which merge from one platform to the other. Mobile agents, as stated in Gitter et al. [12], are autonomous software components, being able to change their execution environment in heterogeneous networks to perform predefined tasks in the name of its user [12]. Besides the general security requirements of agent technology, an extensive security analysis has to be done taking into account the typical MAS's features. In this section, we first determine the security requirements of a MAS (Section III-A), then introduce existing security implementations of JADE (Section III-B) and finally, compare them to determine the remaining security gaps (Section III-C).

A. Security requirements for mobile agents

To determine the MAS security requirements, it has to be considered: a) the agent specific security issues, like data access restriction, agent data protection, etc. b) the security protection of the platform and execution environment and c) the general security requirements, like ensuring the integrity, confidentiality and availability of the agent system. This includes security measures of agent specific access control, resource management usage, and the platform protection against malicious agents arriving from other locations. The following lists the security requirements of multi-agent systems. The bold text indicates threads listed in Fig. 1:

Network Security: Network security is essential to ensure the agent's integrity and confidentiality on transmission level. Existing security solutions like secure sockets or correspondingly configured firewalls may be used to guarantee a secure transmission between the system components and protect the system against **Denial of Service** attacks [13].

Confidentiality of Mobile Agents: The ability to change the execution environment autonomously, requires appropriate solutions to ensure the confidentiality of mobile agents. For many applications, a standard point-to-point encryption provides adequate protection against **eavesdropping the communication** and **injecting messages**. But there exist more complex applications requiring the consideration of additional security measures. Agents often collect and process information on different network locations. In case an agent migrates into multiple platforms in various areas of trust only selected platforms should be authorized to access the information collected by the agent. Therefore, a platform specific encryption is necessary to protect the agent against **unauthorized access to collected data**.

Integrity of Mobile Agents: An illegal **manipulation of mobile agents** may take place on transport level or triggered by destructive system components or malicious remote agent platforms on the migration path. Particular attention is also necessary to protect the agent against destructive platforms may attacking the availability of mobile agents (**agent paralyzation**). After the agent reaches its destination platform, the platform executes the agent actions. Thus, the agent is entirely managed and operated by the underlying platform. Consequently, a malicious agent platform is able to manipulate the agent. This risk leads to an additional security requirement, comprising the agent protection against unauthorized manipulations during the migration process, respectively to find a way to detect such a manipulation.

Malicious Agent Protection: The ability to change the execution environment autonomously not only results in higher agent security requirements but also requires an additional consideration of the agent platform protection against

 ISBN: 978-1-61208-547-0

unauthorized resource access and **platform paralyzation**. One additional security requirement in the context of agent platform protection is to be able to comprehend the whole migration path of a mobile agent accessing the platform. The migration of a mobile agent may be triggered by several different components and therefore the agent platform is not able to follow the migration path precisely. The information on which platforms an agent was executed previously is helpful to determine the general trust of an incoming agent and its memorized data. Therefore, the ability to follow the migration path is an essential security requirement to protect the platform against malicious agents.

Trust Level of Merged Agents: The execution of a mobile agent on a remote platform represents a code injection because an unknown program code which is only partly controllable by the platform will be executed. The question is: “Can the agent be trusted?” To answer this question we need a metric to verify the trust level of an incoming agent. At present, this is not possible. There is a risk that malicious agents perform actions that will cause severe damage to the underlying platform (**inject malicious agents**). For example, DoS attacks against the availability of services, extraction of confidential information, etc. Therefore, it is necessary to be able to verify the trust level of the incoming agent before it is accepted. Otherwise, the platform has to prevent the execution of actions by the agent.

B. Security implementations in JADE

The basic version of JADE provides no specific security implementations to protect the agent system. Nevertheless, to achieve a basic security level, the framework has to be extended with security plugins provided by the JADE manufacturer. The plugin design allows flexibility and the ability to include only the necessary plugins, which are needed for the application-specific security level.

The most important security expansion, providing the basic security measures, is called *JADE Security* [14]. The *JADE Security* extension is based on the *Java Security Model* and consists of several security implementations, including the authentication of system components, the corresponding permission allocation, as well as message encryption and digital signature service [14]. To provide these security measures, the containers and agents have to be configured with additional information, including certificates and permission stores. As the signature and encryption service only secure the transmitted messages but not the agent transfer, additional security plugins like JADE Public Key Infrastruktur (*JADE-PKI*) [15] and Instant Message Transfer Protocol (IMTP) over Secure Sockets Layer (SSL) (*IMTPoverSSL*) [16] are necessary providing an encoded communication channel between the containers. Further information concerning available security measures is to be found in [5][6][14].

C. Missing Security in JADE

Assuming the use of the previous described security expansions, it is possible to reduce some general security threats. This includes the **eavesdropping of the communication**, the **injection of messages**, **agent and platform paralyzation**, and the **unauthorized access to resources** of the platform. Nevertheless, some security aspects were left unconsidered. Figure 1 shows an overview of existing security threats as well as the respective security implementations of agent platform JADE.

Threats \ Security implementations	Security	Permission	Signature	Encryption	IMTP o. SSL	JADE-PKI
Agent platform paralyzation		■				
Unauthorized resource access		■				
Eavesdropping the communication				■	■	■
Inject messages			■		■	■
Inject malicous agents						
Agent paralyzation		■				
Unauthorized access to collected data						
Manipulation of agents					■	■
Denial of service						

■ Security implementation mitigates the threat ■ Positive impact but no solution

Figure 1. Security threats and their respective security implementations

Here, we focus on the ”yellow” and ”grey” marked boxes, which show the missing JADE implementations.

Agent and Platform Paralyzation: The agents and platforms have to be protected against misbehaviour like overloading agents with spam messages caused by faulty or malicious system components. To be able to reduce this threat extensive permission restrictions are necessary.

Unauthorized Access to Collected Data & Manipulation of Agents: Another security issue is the lack of protection against **unauthorized access to collected data**. This issue needs to be discussed with reference to two different aspects. First, the unauthorized access to mobile agent’s information during the transmission process and secondly the information access by unauthorized agent platforms. In case one of the expansions *JADE-PKI* or *IMTPoverSSL* is used it is possible to ensure the agent’s confidentiality during the transmission process and also avoid illegal **manipulation of agents**. However, the access by unauthorized agent platforms is far more complicated. Due to the fact that the security expansion does not have an authorization model that fits for the mobile agent’s needs, and therefore no location based access rights can be granted, the expansion does not offer a sufficient protection [5]. Therefore, additional security mechanisms are required protecting the agent’s confidential data, specific to its current execution environment.

Injecting Malicious Agents: The available security extensions focus on the operations of static agent systems. The

 ISBN: 978-1-61208-547-0

specific security requirements for the agent mobility [14] is not covered sufficiently. There are no effective mechanisms available to verify that a mobile agent is not malicious but trustable, before an agent is accepted and actions processed on the destination platform. A minimum protection is already guaranteed by the provided authentication of particular components, but this only applies to the agent platforms and containers. No platform-specific authentication model is available for mobile agents. As soon as a container or agent platform is authenticated in the system, it is allowed to migrate all its containing agents to a remote execution environment. After the migration of the agents to the destination component, the trust of incoming agents cannot be verified any more. Depending on the application scenario the worst security issues, including **unauthorized resource access** and the execution of a malicious program code (**injecting malicious agents**) causing severe damage to the remote computer system, may result from this security restriction.

Denial of Service (DoS): A general security issue is the **Denial of Service (DoS)** attacks. Although it is already possible to mitigate the impact of DoS attacks, by a targeted restriction of the communication, specific characteristics of DoS attacks will remain a threat. These attacks often take advantage of specific vulnerabilities of the implementation or specific features of the agent framework. Relating to JADE, this usually includes the recursive cloning of agents, the overloading of agents via spam messages as well as tailored restrictions on the availability of the *Agent Management System (AMS)* component [5]. By means of a strong limitation of the component's privileges and the use of appropriate communication rules, it is possible to mitigate the impact of these attacks.

The described vulnerabilities refer mostly to agent systems comprising several areas of trust and use the ability to migrate agents to remote locations. For static agent systems an appropriate security level is reached by the existing security expansion. The following sections introduce some additional security measures to protect the system against the uncovered threats.

D. Confidentiality of Mobile Agent Data

Mobile agents often migrate across multiple platforms and collect sensitive data at each site. In order to be able to protect the collected information, additional security measures are necessary to restrict the data access to selected platforms. One approach to achieve this protection is the use of an environmental key generation as described in [4].

The primary objective of the environmental key generation scenario is to hide the data collected by the agent until a particular environmental condition is reached on the destination platform. A possible environmental condition could be, for example, to find a certain environmental key or the availability of predefined resources or the existence of cooperating agents. The described security objective is of particular importance in case an agent has to cross multiple platforms to perform its predefined task. In such a scenario, the platforms should only be able to access selected information. The sensitive information is encrypted by using a key being partly known by the agent and the information only may be decrypted by the agent itself. After the agent reaches the destination platform, the predefined environmental condition is checked. When the condition is met, an activation key is generated to decrypt the enciphered information carried by the agent. To generate the activation key the part of the key known by the agent will be combined with the condition matching result. Therefore, without meeting the environmental condition, neither the agent nor the executing platform are able to decrypt the information.

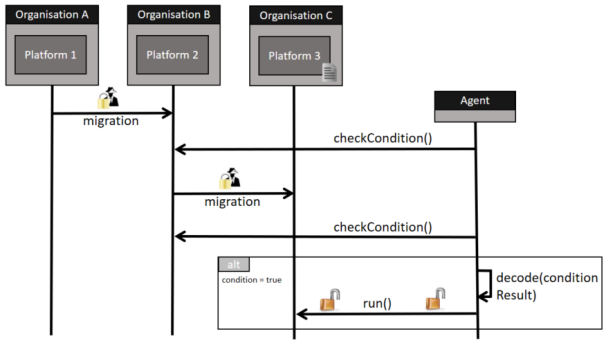

Figure 2. Environmental key generation scenario

Figure 2 shows three agent platforms being operated by different organizations. In Figure 2, the environmental condition is met when a platform contains a configuration file with a certain configuration entry, such as the use of appropriate encryption channels like HTTP for web servers. As soon as the first platform is reached, the encrypted agent starts to search for the condition matching file. In case the current platform does not meet the condition, the agent is not able to decrypt the carried information and therefore the platform is not able to access the data. Nevertheless, the agent is able to perform all kinds of tasks at its current location except the use of the encrypted information. After another migration takes place, the agent reaches the final destination. There, the agent will find the specified configuration file and finally decrypt the carried information by using the already partly known key in combination with a predefined entry in the configuration file. Thus, the agent as well as the executing platform are able to access the information.

To implement the described security measure, the agents have to be able to encrypt and decrypt specific data parts and to check the environmental condition. The security measure has been integrated into the agent framework JADE without the necessity to deploy customized execution environments. Besides the described environmental key generation, additional security measures exist to enhance the confidentiality of mobile agents. An algorithm for the generation of an

 ISBN: 978-1-61208-547-0

encrypted function (see [13]) is implemented into the basic platform of the agent. The encrypted function is integrated into the agent after the agent has been instantiated. After the agent reaches its destination platform, the function may be executed by using location-specific arguments. The platforms of the agent's migration path will be able to execute the agent but no further information about the implemented functionality or the containing data is accessible. In case a platform comprises the required key to decrypt the function, the platform is able to access the information.

The security measures mentioned provide an extensive confidentiality protection for mobile agents. However, this kind of mechanisms may lead to a considerable danger with reference to the security of the agent platforms. The platforms have to execute an incoming agent without any additional information about its level of trust. The following section introduces some solutions to provide the ability to check the level of trust of incoming agents.

IV. Trustworthy Mobile Agents

In order to achieve a comprehensive protection of the agent system, appropriate security measures to verify the overall trust level of a merging mobile agent have to be realized. A minimal protection is already provided by the use of encrypted and signed communication between the different locations. A signed communication only ensures the integrity and the origin of an arriving agent but does not provide the ability to make assumptions about the general trust level, additional security considerations are necessary.

There are two possibilities to increase the trust level of an mobile agent: a) Registering verified agents at a trusted authority repository (third party) described in more detail in this section. b) Tracking of the migration path. This mainly comprises recording of previously passed locations of the agent and therefore helps to achieve the traceability to other system components (see [17]).

A. Repository of Trustworthy Agents

One way to mitigate is to enhance the agent system with a trusted authority, comprising the verified and considered trustworthy agents. Before the agents are stored in the trusted authority, the authority analyzes the program code by examining the defined behaviour. When an agent is entering a new platform, the platform is in a position to check whether the agent store contains the agent, using the unique identifier. If the trusted authority comprises the incoming agent, additional information about the agent may be checked. Such information may include information about the defined behaviour as well as the corresponding hash value. Further analysis, based on the behaviour, the requested hash value may be used to verify whether the incoming agent matches with the expected and authorized agent. In case the incoming agent is not to be found in the agent store or the hash values are different, the agent may be rejected by the platform. Figure 3 visualizes the architecture and shows a high-level process overview.

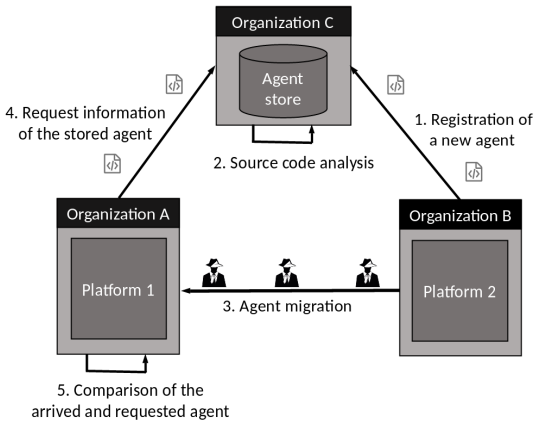

Figure 3. Overview: Repository of Trustworthy Agents

A challenge with reference to the hash-based check of identity is to generate comparable hash values. This challenge primarily originates from the fact that agent platforms append additional information to the agents during the instantiation time. For example, this includes the agent identifier and some information about the current execution environment. The local information leads to different hash values, depending on the current location. Another problem is that the involved components have different data formats representing the same agent.

The trusted authority's data format is the program code to verify the agent's behaviour, whereas the platform's data format, representing the migrated agent, is an instance of the class *Agent*. Nevertheless, to generate meaningful and comparable hash values, the values have to be created at class level instead of instance level. This kind of hash values guarantees that the incoming agent's behaviour matches with the stored and verified program code. Information about the current execution status as well as the collected data at runtime were left unconsidered in this approach.

B. Integration of Repository into JADE

JADE provides a function *serve(HorizontalCommand cmd)*, being activated automatically when a new agent reaches the platform. In order to be able to use the described security measure, this function has to be expanded by an additional application logic. Figure 4 shows the process flow when the expanded function *serve(..)* is executed.

After de-serializing the agent, after it has reached the platform, the agent's unique identifier is extracted by the *getAID()* function defined by the superclass *Agent*. The identifier is used to check whether the trusted authority is able to identify the requested agent. In case the agent is known by the agent store, the trusted authority responses with the corresponding hash value. Afterwards, the platform

 ISBN: 978-1-61208-547-0

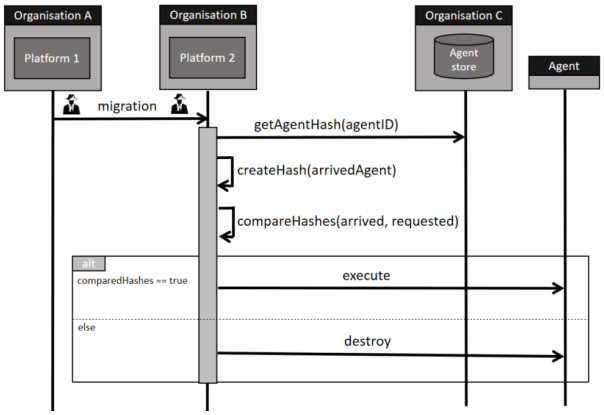

Figure 4. Process flow of extended serve(..) function

generates a hash value for the incoming agent, which subsequently is compared with the requested value. Assuming that the result of the comparison is positive the default implementation of the *serve(..)* function will be continued and the agent is accepted. In case the requested agent will not be found in the trusted authority or the compared hash values diverge, the incoming agent will be rejected and the platform prevents its execution.

If the described security measure is applied, the agent system provides the ability to restrict the authorized agents previously and thus the system is protected against malicious agents. The potential arises especially in large systems consisting of several independent organizations. By means of the use of the agent store a situation of mutual trust between the different organizations may be created. A proof of concept implementation can be found in [18].

V. Conclusion

The existing security expansions (JADE Security [14]) for JADE already reduces some of the general security threats. Reflecting on the specific security requirements for mobile agents, it will be realized that some security issues remain unconsidered. This paper gives a summary of some possible security measures, the missing security measures by JADE, and describes how to achieve the confidentiality of mobile agent data and how to protect the platform against malicious agents. However, further efforts are necessary to achieve an extensive security level for the whole agent system. A future work will be to develop easily integrable security plugins to increase the practicability of the described solutions.

References

[1] T. Ruebsamen and C. Reich, "Supporting cloud accountability by collecting evidence using audit agents," in 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, vol. 1, 2013, pp. 185–190.

[2] Y. S. . K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press, 2009, ISBN: 978-0521899437.

[3] G. Jezic, Y.-H. J. Chen-Burger, R. J. Howlett, and L. C. Jain, Eds., Agent and Multi-Agent Systems: Technology and Applications. Springer International Publishing, 2016, ISBN: 9783319398822.

[4] S. Ahila and K. Shunmuganathan, "Overview of mobile agent security issues - solutions," in Information Communication and Embedded Systems. Institute of Electrical and Electronics Engineers, 2014, pp. 1–6, ISBN: 9781479936984.

[5] A. Bürkle, A. Hertel, W. Müller, and M. Wieser, "Evaluating the security of mobile agent platforms," Autonomous Agents and Multi-Agent Systems, vol. 18, no. 2, 2009, pp. 295–311, ISSN: 1387-2532.

[6] A. R. X. Villa, A. Schuster, "Security for a multi-agent system based on jade," Computers & Security, vol. 26, 2007, pp. 391 – 400.

[7] F. Piette, C. Caval, A. El Fallah Seghrouchni, P. Taillibert, and C. Dinont, "A multi-agent system for resource privacy: Deployment of ambient applications in smart environments (extended abstract)," in Proceedings of the 2016 International Conference on Autonomous Agents, ser. AAMAS '16. International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 1–2, ISBN: 9781450342391.

[8] G. Geetha and C. Jayakumar, "Implementation of trust and reputation management for free-roaming mobile agent security," IEEE Systems Journal, vol. 9, no. 2, June 2015, pp. 556–566, ISSN: 1932-8184.

[9] Y. Dong, N. Gupta, and N. Chopra, "Content modification attacks on consensus seeking multi-agent system with double-integrator dynamics," Chaos: An Interdisciplinary Journal of Nonlinear Science, vol. 26, no. 11, 2016, p. 116305. [Online]. Available: http://dx.doi.org/10.1063/1.4965034

[10] R. Gengarajoo, P. S.G., and C. K. Loo, Evaluating Trust in Multi-Agents System Through Temporal Difference Leaning. Springer International Publishing, 2016, pp. 513–524, ISBN: 9783319435060.

[11] T. I. Lab, "Java agent development framework," November 2016, URL: http://jade.tilab.com/ [accessed: 2017-03-10].

[12] R. Gitter, V. Lotz, and U. Pinsdorf, Security and legal force for mobile agents. Deutscher Universitäts-Verlag, 2007, ISBN: 9783824421732.

[13] O. Paracha, A Security Framework for Mobile Agent Systems, 2006, ISBN: 9781599427249.

[14] "Jade security guide," 2016, URL: http://www.jade.tilab.com/doc/tutorials/JADE_Security.pdf [accessed: 2017-03-10].

[15] A. P. Zonowski, "Jade-pki 1.0 manuel," November 2016, URL: http://jade.tilab.com/doc/tutorials/PKI_Guide.pdf [accessed: 2017-03-10].

[16] G. Vitaglione, "Mutual-authenticated ssl imtp connections," Dezember 2015, URL: http://jade.tilab.com/doc/tutorials/SSL-IMTP/SSL-IMTP.doc [accessed: 2017-03-10].

[17] C. Mitchell, Security for Mobility. Institution of Engineering and Technology, 2004, ISBN: 9780863413377.

[18] T. Bayer and C. Reich, "Proof of concept implementation of trusted agents," November 2016, URL: http://en.hs-furtwangen.de/fileadmin/userupload/IfCCITS/Dokumente/Publications/Theses/JADETrustedAgents.rar [accessed: 2017-03-10].

 ISBN: 978-1-61208-547-0