Design and Implementation of Security Mechanisms for a Hierarchical Community-Based Multi-Agent System

Kenichi Takahashi[1], Yoshiki Mitsuyuki[2], Tsunenori Mine[2], Kouichi Sakurai[1,2], and Makoto Amamiya[2]

[1] Institute of Systems & Information Technologies/KYUSHU,
2-1-22 Momochihama, Sawara-ku, Fukuoka 814-0001, Japan.
{takahashi, sakurai}@isit.or.jp
[2] Faculty of Information Science and Electrical Engineering, Kyushu University,
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan.
{mitsu, mine, amamiya}@al.is.kyushu-u.ac.jp

Abstract. Recently, several community-based systems have been developed; however, almost all such systems have been developed as Web-server-based systems. Thus, server administrator can easily eavesdrop on user communications, since they have to send/receive information through the server. Therefore, we propose multi-agent-based peer-to-peer (P2P) system wherein each peer manages his/her information and exchanges it with other peers directly. This, thus, resolves the problems posed by Web-server-based systems; however, we have to consider attacks from malicious third parties. This study designs and implements security protocols/mechanisms for a hierarchical community-based multi-agent system. Furthermore, if we consider a practical use case, we should be able to demonstrate that the proposed system can be implemented by combining it with existing security techniques for more reliable and rapid deployment.

1 Introduction

The evolution of the Internet has made it difficult to discover target information, further, the nature of the Internet, which allows everyone to easily contribute, has made it difficult to ensure the reliability of information. This imposes the inference of reliability of information on users. Therefore, community-based systems have been developed. In a community-based system, users with common interests and/or objectives organize a community. They can communicate efficiently on the topics within a community. Furthermore, these communities also facilitate the secure exchange of privacy-related information by restricting communications to within a community.

Almost community-based systems are developed as Web-server-based systems. These systems enforce security by employing public key infrastructure (PKI), https, access controls mechanisms, etc. However, since users have to send/receive information through a server, server administrator can eavesdrop

on their communications. Moreover, a vulnerability in the server may threaten the privacy of all users.

Therefore, we propose a multi-agentbbased peer-to-peer (P2P) system wherein each peer manages his/her information and exchanges it with other peers directly. P2P systems can be divided into pure P2P systems such as Gnutella and hybrid P2P systems such as Napster. Hybrid P2P systems employ one or more special peers who provide a special service such as resource discovery and/or indexing of peers, while pure P2P systems do not employ special peers. With regard to a community-based system, a hybrid P2P system is more suitable since special peers organize communities.

In the proposed system, the formation of a community depends on a special peer, but each peer in the community can exchange their information directly. This, thus, remedies the problems posed by Web-server-based systems; however, we have to consider attacks from malicious third parties, such as eavesdropping, message alteration, spoofing. In this study, we design and implement security protocols/mechanisms for a hierarchical community-based multi-agent system. Furthermore, if we consider a practical use case, we should be able to demonstrate that the proposed system can be implemented by combining it with existing security techniques for more reliable and rapid deployment.

The remainder of this paper is organized as follows; Section 2 presents related works. Section 3 highlights the security requirements of a hierarchical community-based multi-agent system. Section 4 describes designs of the security mechanisms satisfying these requirements. In section 5, we demonstrate an example application.

2 Related Work

In [11], an agent-community-based P2P information retrieval system is proposed; in this system, agent communities manage and look up information efficiently. In [15], a group-based reputation system is proposed, wherein peers evaluate the credibility of a given peer based on his/her local trust information or references from within the group. However, these do not consider security.

In [4, 13], the security functions for a few multi-agent-based applications have been discussed. In [7], SAgent is proposed for protecting the computations of mobile agent applications in a potentially hostile environment. Also, several researches [10, 16] have discussed mobile agent security. These, however, do not take a community structure into consideration.

Most security researches in P2P systems have focused on anonymity or trust relationships; however, very few have focused on security countermeasures against attackes such as eavesdropping message alteration, and spoofing [17]. In [1, 3] , these security concerns with regard to community-based system have been focused on. [3] allows a peer to change the security policy flexibly based on applications; however, it does not indicate what type of security techniques are required in particular application. [1] focuses on the security on information

sharing; however, in this system, a peer has to create a digital signature for each message.

JXTA [14] provides certain functions for the development of secure P2P systems; it requires developers to determine and combine security functions for an application. The Globus Project provides the Globus Security Infrastructure (GSI) [6], which is a security framework for grid computing applications. Groove [5] provides spontaneous shared spaces in which a small group of collaborators can securely exchange information. However, these frameworks do not take a hierarchical community structure into considerations.

3 A Hierarchical Community-Based Multi-Agent System

We represent a user as an agent managing his/her information and supporting activities. These agents who have common interests and objectives organize a community. In the proposed model, a community is also defined as one agent; thus, communities can also organize a community. Therefore, our model allows a hierarchical community structure (figure 1) [18]. This nature matches that of real societies. For example, each company consists of certain departments with each department comprising certain sections and each section comprises certain staff. As in real societies, a community has a representative, named portal agent. A portal agent is responsible for organizing a community and subsequently providing services for new agent participation, deregistration, and channels to actualize communication from outside the community to within and vice versa. However, the portal agent does not restrict the communications of agents within the community. Thus, agents can communicate with each other freely and directly. This nature protects agents within a community from risks of information leakage caused by a portal agent.

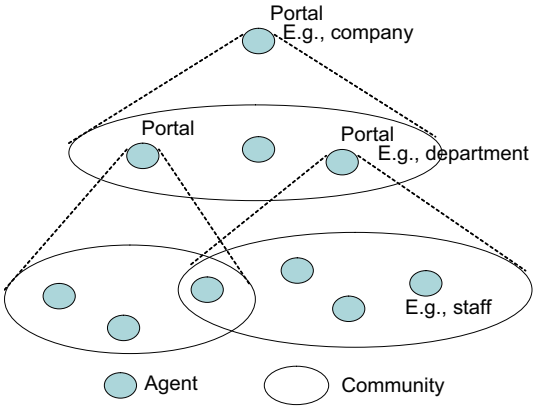

Fig. 1. Hierarchical community structure

3.1 Levels of Information Security

An agent manages not only non-privacy-related information but also privacy-related information to support his/her activities. Thus, each agent is responsible for releasing the appropriate information to only the appropriate partner in an appropriate manner. This implies that each agent should protect his/her information by appliying the appropriate cryptographic techniques according to levels of information security. With regard to a system with a hierarchical community structure, we can define four levels of information security:

Closed information, which cannot be released to any agent.
Partner-limited information, which can be released to only those agents who satisfy a specific condition.
Community-limited information, which can be released to only those agents in the same community.
Open information, which can be released to all agents.

We do not discuss which levels of information security should be applied to a particular information entity, since this depends on the application being considered. Information should be protected in an appropriate manner based on the level of information security.

3.2 Attacks and Corresponding Countermeasures

We have to consider the prevention of attacks from malicious third parties, such as eavesdropping, message alterations and spoofing (figure 2). Since closed information cannot be released to any agent, we do not need to consider preventive countermeasures for safequarding this type of information against these attacks. However, in the case of the other levels of information security, we have to consider to prevent these attacks.

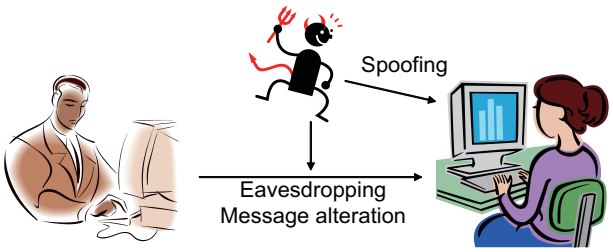

Fig. 2. Eavesdropping, message alteration, and spoofing attacks

With regard to eavesdropping, no preventive countermeasure are required for open informatio as it can be released to all agents. However, community-limited and partner-limited require preventive countermeasures. Community-limited information can be released to only agents in the same community; thus,

agents in the same community should be able to access the information but other agents should not. On the other hand, partner-limited information can be released to only those agents who satisfy a particular condition; thus, only an agent who satisfies a specific condition should be able to access the information. Therefore, we introduce the group and P2P keys for encryption/decryption of community-limited and partner-limited information, respectively.

In order to protect against message alteration and spoofing, all types of information require preventive countermeasures. We can use a digital signature for this purpose; however, the creation and verification of a digital signatures require high computational power. In this case, we beleive that it would suffice to detect the spoofing when an attack may occur. Therefore, we propose a mechanism wherein a sender attaches a hash value generated from the content of a message with secret information (e.g. random number) to the message. When the receiver wants to verify whether the message was spoofing or not, he/she requests the secret information (signed by a digital signature) from the sender. On receiving this secret information, if the hash value of the message matches that generated from the secret information, the message can be considered non-spoofed.

Digital signatures indicate that a particular information entity has been created by someone who possesses a key for its creation; however, it does not indicate who the key belong to. Usually, a certificate authority (CA) keeps a record of which belongs to who by offline communication, ID/password authentication, etc. In the proposed system, a portal agent assumes this role. Therefore, a portal agent specifies requirements for new agent participations. When an agent wishes to join a community, he/she has to provide information that satisfies the specified requirements. Then, the portal agent issues a digital certificate for the agent's key. The agent can thus prove that the key is recognized by the portal agent by displaying his/her digital certificate.

3.3 Necessary Protocols

We need to design and implement the following protocols for the prevention of attacks from malicious third parties:

Participation in a community It is necessary to prepare for activities of new agents wishing to join a community. Each community has specific requirements for participation, for example, ID/password, invitation from a friend (e.g., mixi [12]), etc. Therefore, a portal agent needs to check whether new agent satisfies the requirements, issue a digital certificate for the new agent's public key, and provides the group key. Note that the information for satisfying these requirements may also be sensitive.

Group key communication It provides a secure channel for communication among agents from the same community. The sender embeds secret information into a message as the evidence of his/her message. Thus, he/she can prove that the message belongs to him/her. Further, it is also necessary to encrypt and decrypt community-limited information using a group key.

Mutual authentication and P2P communication It provides for the mutual authentication and secure communication between two agents. Mutual authentication is achieved by verifying the digital certificate of each agent. Further, two agents need to share a common key (P2P key) for the encryption and decryption of partner-limited information. After sharing a P2P key, they can exchange partner-limited information securely. Moreover, they can detect whether the group key communication have been spoofed by exchangin secret information.

Leaving a community When an agent leaves a community, his/her digital certificate should be invalidated. Then, his/her digital signature should be recorded in the certificate revocation list (CRL).

Group key updation When community members change, updated group key is necessary for forward and backward security.

CRL Reference Some digital signature may be come invalid before their expiry period. Therefore, an agent may want to refer to Cthe RL periodically.

Although we have designed all of the above protocols, this paper presents only participation in a community, group key communication, and mutual authentication and P2P communication.

3.4 Security Techniques

We use the following techniques for the design and implementation of the proposed security.

Public key cryptosystem This algorithm utilizes a public-secret key pair. A message encrypted using the public key can only be decrypted using the secret key and vice versa. In this study, the public and secret keys of agent X are depicted as PK_x and SK_x, respectively. The encryption and decryption of a message M using a key K are represented as $E_K(M)$ and $D_K(M)$, respectively.

Common key cryptosystem This algorithm utilizes the same key (common key) for message encryption and decryption. Thus, the sender and receiver of a message have to share a common key in advance. Moreover, the algorithm is much faster than the public key cryptosystem. In this study, a common key between agents X and Y is represented as CK_{xy}.

Digital signature A digital signature is created using a secret key (of public key cryptosystem). Since only an agent who knows the secret key can create the digital signature for a message, other agents can confirm that the message is created by the agent who has the secret key. This study represents the digital signature of message M created using a secret key SK_x as $Sign_{SK_x}(M)$.

Message authentication code (MAC) MAC is used for the detection of message alteration by using a hash function. The MAC for message M is represented as $MAC(M)$.

Digital certificate A digital certificate binds a public key with an identity. It is issued by the CA (a portal agent in our system). In this study, the digital certificate created by a portal agent P (who possesses the secret key SK_p) for a public key PK_x is represented as $Cert_{SK_p}(PK_x)$.

Group key The group key is shared among the agents belonging to the same community and enables them to encrypt and decrypt a message. In this study, the group key of community C is represented as GK_c.

Hash chain A hash chain is a sequence of hash values computed by adapting the hash function n times. In this study, the hash chain with seed S is represented as $H^1(S), H^2(S), ..., H^n(S)$. Even if $H^i(S), where i \leq n$, is known to other agents, they cannot compute any $H^j(S), where j < i$.

4 Design of Security Protocols

We designed the protocols for participation in a community, group key communication, and mutual authentication and P2P communication.

4.1 Participation in a Community

With regard to participation in a community, it is necessary to check whether new agent satisfies the requirements for community participation or not, issue a digital certificate for the new agent's public key, and provides the group key. Therefore, we designed the protocol showin in figure 3.

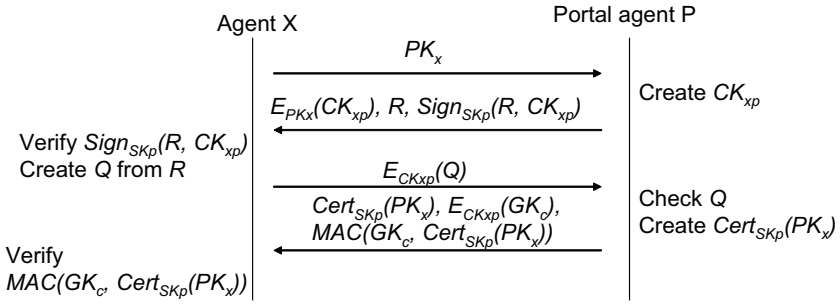

Fig. 3. Protocol for participation in a community

1. When an agent X wishes to participate in a community C, X sends PK_x to the portal agent P of C.
2. P creates a common key CK_{xp} and responds by sending $E_{PK_x}(CK_{xp})$, requirements for participation R, and $Sign_{SK_p}(R, CK_{xp})$.
3. X decrypts $E_{PK_x}(CK_{xp})$ using SK_x and verifies the integrity of CK_{xp} and R using $Sign_{SK_p}(R, CK_{xp})$. Next, X creates a qualification Q for R and sends $E_{CK_{xp}}(Q)$ to P.

4. P decrypts $E_{CK_{xp}}(Q)$ and checks whether Q satisfies R. If it is satisfied, P creates $Cert_{SK_p}(PK_x)$ and responds by sending $Cert_{SK_p}(PK_x)$, $E_{CK_{xp}}(GK_c)$, and $MAC(GK_c, Cert_{SK_p}(PK_x))$.
5. X decrypts $E_{CK_{xp}}(GK_c)$ and verifies the possibility of message alteration using $MAC(GK_c, Cert_{SK_p}(PK_x))$. Also, X creates a hash chain $H^1(S_x), H^2(S_x)$, ..., $H^n(S_x)$.

Thus, only those agents who satisfy requirements for community participation can obtain the digital certificate and group key.

4.2 Group Key Communication

With regard to group key communication, message encryption using a group key and preventive countermeasures against spoofing are necessary. The proposed system provides two countermeasures against spoofing: One simply involves using a digital signature; however, it requires much amount of computations in order to create a digital signature for each message. The second involves the usage of a hash chain, a sender attaches $MAC(M, H^{n-m}(S))$ to the m-th message M. Subsequently, the receiver stores the $MAC(M, H^{n-m}(S))$ and M pair. When the receiver needs to verify whether the message has been spoofed, he/she can do it by obtaining $H^{n-m}(S)$ from the sender (using P2P communication).

4.3 Mutual Authentication and P2P Communication

A protocol for mutual authentication between agents X and Y is shown in figure 4.

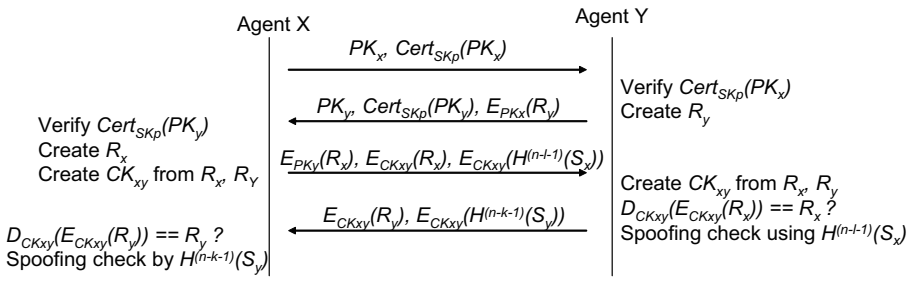

Fig. 4. Protocol for mutual authentication and P2P communication

1. Agent X sends PK_x and $Cert_{SK_p}(PK_x)$, which is the digital certificate provided by a portal agent P to agent Y.
2. Y verifies $Cert_{SK_p}(PK_x)$ using PK_p. Here, only when Y trusts P, the result of verification evaluates to true. Next, Y generates a random number R_y and responds by sending PK_y, $Cert_{SK_p}(PK_y)$, and $E_{PK_x}(R_y)$.

3. X verifies $Cert_{SK_p}(PK_y)$ and generates a random number R_x. Next, the agent X decrypts $E_{PK_x}(R_y)$ using SK_x. Then, X can create a P2P key (common key) CK_{xy} using R_x and R_y. Finally, X sends $E_{PK_y}(R_x)$, $E_{CK_{xy}}(R_x)$, and $E_{CK_{xy}}(H^{n-l-1}(S_x))$ to Y, where l is the number of messages already sent by X as group key communication. Here, $E_{CK_{xy}}(H^{n-l-1}(S_x))$ is optional. It is required only when Y wishes to verify the messages in group key communication have been spoofed.
4. Y decrypts $E_{PK_y}(R_x)$ using SK_y and creates a P2P key CK_{xy} using R_x and R_y. Subsequently, CK_{xy} is surely shared if and only if the decryption of $E_{CK_{xy}}(R_x)$ yields R_x. Finally, Y responds by sending $E_{CK_{xy}}(R_y)$ and $E_{CK_{xy}}(H^{n-k-1}(S_y))$, where k is the number of messages already sent by Y as group key communication. $E_{CK_{xy}}(H^{n-k-1}(S_y))$ is also optional. Moreover, Y can verify whether the messages in group key communication have been spoofed using $MAC(M, H^{n-i-1}(S_x))$ generated using $H^{n-i-1}(S_x), where i > l$.
5. X decrypts $E_{CK_{xy}}(R_y)$. Subsequently, CK_{xy} is surely shared if and only if the decryption of $E_{CK_{xy}}(R_y)$ yields R_y. X can also verify whether the messages in group key communication have been spoofed.

Thus, X and Y share the P2P key CK_{xy} and can exchange partner-limited information securely between them in P2P fashion.

4.4 Considerations of a Hierarchical Community Structure

There are two types of a community structure; upper communities handle important information and lower communities handle important information. For example, the hierarchical community structure representing a company organization handles important information in executive meetings than one in the meeting in each section. However, the community comprising friends (e.g., mixi) handles important information in the lower communities. Thus, security requirements differ with the type of community structure.

A portal agent manages this type of security requirement as a policy. When an upper community handles important information, the portal agent prohibits to forward group key communication messages from the upper community to the lower communities. On the other hand, when lower communities handle important information, the portal agent prohibits to forward group key communication messagess from the lower community to the upper community. Note that if an agent expricitly specifies that message forwarding to an upper/lower community should be allowed, the policy will be neglected.

We have to also consider group key and P2P communications between agents belonging to different communities. These communications are kept secure by formulating an agreement between the communities. In the process of agreement, portal agents P and Q of each community exchange each public key PK_p and PK_q, and change their policy to allow message forwarding from the opposite community. Consequently, agents belonging to different communities are able to authenticate each other and share a P2P key. They can, thus, do P2P

communications securely (left panel in figure 5). Moreover, when lower communities handle important information, group key communication messages between agents belonging to different communities should be protected. Therefore, the portal agents have to share a P2P key. Then, the messages exchanged between the communities are protected by encryption using the P2P key (right panel in figure 5). Thus, messages exchanged between agents belonging to different communities are protected.

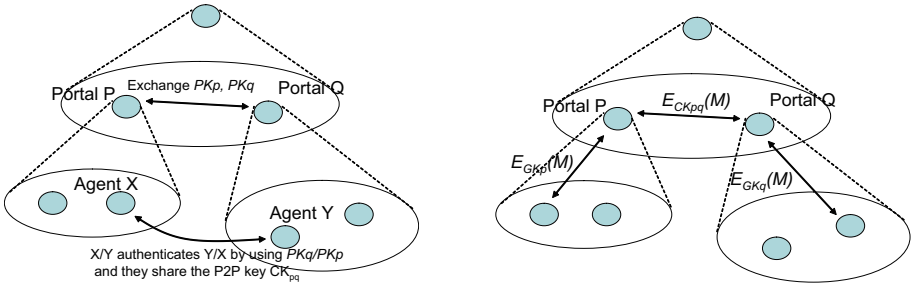

Fig. 5. Mutual authentication and P2P key sharing between agents beloing to different communities (left), and group key communication between agents belonging to different communities (right)

5 An Application Example

Nowadays, various business matching services are being actively used and gaining popularity [2, 9]. Since these services are developed as Web-server-based systems, all information is stored on the Web site. Thus, companies may not be willing to use sensitive information for business matching. Therefore, a P2P-based system is more suitable for business matching services. Therefore, we implemented the proposed security mechanisms using Java and developed a business matching system for small-and-medium-sized companies (http://www-al.is.kyushu-u.ac.jp/Kodama/ACN/index.html). This system are working well now.

In the developed business matching system, the requirements for participation in a community are ID and a password pair check. A company can obtain the ID and password by closing a contract with the administrator of the community. The top community comprises two sub-communities, software and semiconductor community; semiconductor community consists of 12 small communities (figure 6).

An agent participates in the community, exchanges information, and advances business matching semi-automatically. There are two steps in business matching: The first step involves finding candidate business partners from within the community; The second step involves agreeing on terms and contracting the

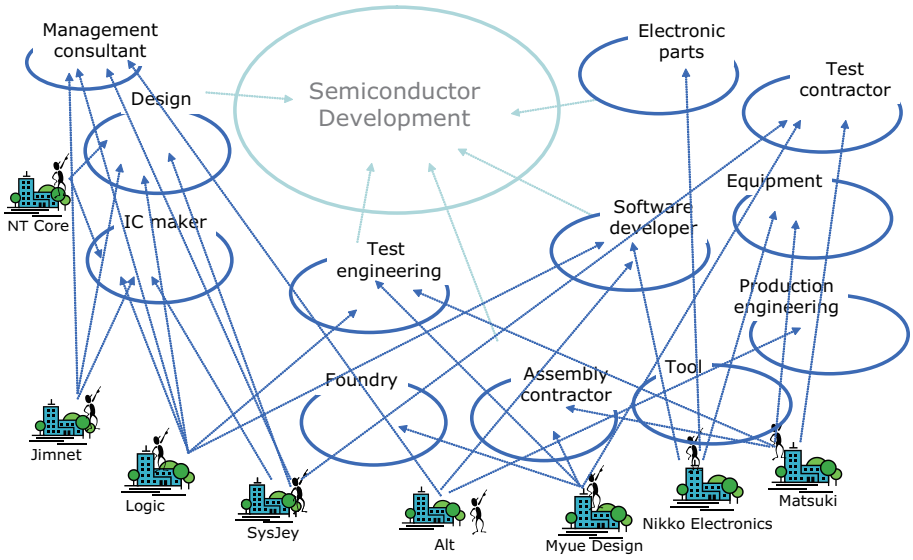

Fig. 6. The community structure for the business matching system

business. The first step requires group key communications for restricting information access to within the community. In the second step, more important information related to the contract is exchanged, therefore, this information should be protected from others by P2P communication. Thus, a company can advance business matching while protecting information.

6 Conclusion

In this study, we have designed and implemented security mechanism for a hierarchical community-based multi-agent system. The proposed system realizes the following: (1) protection of information based on four levels of information security, (2) each community can define unique requirements for participation, and (3) two anti-spoofing methods. Furthermore, we have considered a practical use case, and our security mechanisms can be implemented using Java classes. We are doing the performance tests now. The result of the performance tests will be also shown in this paper, if accepted.

Acknowledgments

This research was supported by the Strategic International Cooperative Program, Japan Science and Technology Agency (JST), and the Strategic Information and Communications R&D Promotion Programme under grant 052310008.

References

1. K.Berket, A. Essiari, A. Muratas. PKI-Based Security for Peer-to-Peer Information Sharing. Prof. of 4th International Conference on Peer-to-Peer Computing, pp. 45–52, 2004.
2. Business Mall. http://www.b-mall.ne.jp/.
3. A. Detsch, L. Gaspary, M. Barcellos, G. Cavalheiro. Towards a Flexible Security Framework for Peer-to-Peer based Grid Computing. Proc. of the 2nd workshop on Middleware for grid computing, pp. 52–56, 2004.
4. FIPA MAS Security white paper. http://www.fipa.org/index.html.
5. Groove. http://www.groove.net/.
6. GT Security (GSI). http://www.globus.org/toolkit/security/.
7. V. Gunupudi, S R. Tate. SAgent: A Security Framework for JADE. Prof. of 5th International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1116–1118, 2006.
8. Java Cryptography Extension (JCE). http://java.sun.com/products/jc.
9. Jetro TTPP. https://www3.jetro.go.jp/ttppoas/indexj.html.
10. T. McDonald, A. Yasinsac. Application Security Models for Mobile Agent Systems. Prof. of International Workshop on Security and Trust Management, pp. 38–53, 2005.
11. T. Mine, D. Matsuno, A. Kogo, M. Amamiya. Design and Implementation of Agent Community Based Peer-to-Peer Information Retrieval Method. Cooperative Information Agents VIII, LNAI 3191, pp. 31–46, 2004.
12. mixi, http://mixi.jp/.
13. S. Poslad, M. Calisti, P. Charlton. Specifying Standard Security Mechanisms in Multi-Agent Systems. Trust, Reputation and Security: Theories and Practice, LNCS 2631, pp. 227–237, 2003.
14. Project JXTA. http://www.jxta.org/.
15. H. Tian, S. Zou, W. Wang, S. Cheng. A Group Based Reputation System for P2P Networks. The 3rd International Conference on Autonomic and Trusted Computing, LNCS 4158, pp. 342–351, 2006.
16. J. Tsai, L. Ma. Security Modeling of Mobile Agent Systems, J. of Ubiquitous Computing and Intelligence, Vol.1, 73–85, 2007.
17. D. Wallach. A Survey of Peer-to-Peer Security Issues. Software Security - Theories and Systems, LNCS 2609, pp. 253–258, 2003.
18. G. Zhong, S. Amamiya, K. Takahashi, T. Mine, M. Amamiya. The Design and Implementation of KODAMA System. IEICE Transactions INF.& SYST., Vol.E85-D, No.4, pp. 637–646, 2002. e/.