Protecting a Mobile Agent's Route against Collusions

Dirk Westhoff[1], Markus Schneider[2], Claus Unger[1], and Firoz Kaderali[2]

[1] FernUniversität Hagen, D-58084
Fachgebiet Praktische Informatik II
[2] FernUniversität Hagen, D-58084
Fachgebiet Kommunikationssysteme
{dirk.westhoff,mark.schneider,claus.unger,firoz.kaderali}@fernuni-hagen.de

Abstract. In the world of mobile agents, security aspects are extensively being discussed, with strong emphasis on how agents can be protected against malicious hosts and vice versa. This paper discusses a method for concealing an agent's route information from being misused by sites en route to collect profile information of the agent's owner. Furthermore, it is shown that the protected route resists attacks from a single malicious host and from colluding malicious hosts as well.

1 Introduction

Mobile agents are becoming more and more important for Internet based electronic markets. In many scenarios, mobile agents represent customers, salesmen or mediators for information, goods and services [1]. Agents are autonomous programs, which, following a route, migrate through a network of sites to accomplish tasks or take orders on behalf of their owners. Without any protection scheme, a visited site may read an agent's data and thus collect information about the agent's owner, e.g. its services, customers, service strategies, collected data, etc. To avoid such a situation, the amount of data accessible to a visited site has to be restricted as much as possible [2].

In this paper, we concentrate on an agent's route information, i.e. the address list of sites to be visited during a trip. The owner provides its agent with an initial route. On its travel, the agent works through the route stage by stage. Protecting the route guarantees that none of the visited stations can manipulate the route in a malicious way or can get an overview of other sites the agent's owner is contacting. To repulse malicious programs like Trojan horses, the identity of an agent becomes known to all visited sites [3], anonymous agents [4] are not dealt with in this paper. In the following we use the concept and terminology of the ALOHA[1]-software package [5].

[1] The ALOHA (Agent Local Help Application) environment allows its users to easily define, send, receive and evaluate agents.

2 Related Work

Methods that protect an agent against attacks can be categorized into those which prevent attacks and those which detect attacks.

Cryptographic traces [6] detect illegal modifications of an agent by a post-mortem analysis of data the agent collected during its journey. *State appraisal* [7] mechanisms protect an agent's dynamical components, especially its execution state, against modifications. When an agent reaches a new site, the appraisal function is evaluated passing as a parameter the agent's current state.

Tamper-proof devices [8] are hardware based and therefore not suitable in open systems. Software based approaches include the *computation of encrypted functions* [9] and *code scrambling* [10]. Unfortunately the first approach can only be applied to polynominal and rational functions. In [10] the code of an agent is re-arranged to disguise the agent's functionality.

Although *onion routing* [11] is not an agent specific approach, it uses partial encryption similar to our method. Onion routing allows anonymous connections and is used to protect a variety of Internet services against eavesdropping and traffic analysis attacks. In contrast, beside concealing the route, our solution detects attacks that modify an agent's route but allows legal route changes.

3 Framework

An agent is an autonomous program which acts on behalf of its owner. According to its route, it visits sites linked together via a communication network. An agent is created, sent, finally received and evaluated in its owner's *home context*. At a visited site, the agent is executed in a *working context*. To save costs, an agent usually does not return to its home context before it has worked off its route; thus during the agent's journey its home site has not to be connected to the communication network all the time. To forward an agent to its next site or to extend an agent's route, each visited site needs access to a certain part of the route. A site is not allowed to remove a not yet visited site from the initial route and thus, e.g., exclude sites from offering their services to the agent. When a visited site extends a route, all added sites must become aware of the fact that they have not been on the initial route, and thus may, e.g., restrict the agent's access rights and functions, e.g. for electronic cashing. When a site changes a route, the change must be uniquely be associated with the site. To avoid arousing suspicion, as soon as a site detects an attack against an agent, it has to send the agent back to its home context.

In the following, we present a concept which reduces the route information, that becomes visible to a visited site, to a minimum, and which protects the route against malicious changes. In addition the concept is flexible enough to handle route extensions during the agent's journey. Our concept carefully considers efficiency aspects like computational complexity, additional network traffic, etc. Because of the latter, the concept of a trustworthy center to be visited between each two consecutive sites, is not taken up. Each site is only given access to the address of its predecessor and its successor site.[2]

Additionally to the route, the agent includes other components like *profile*, *binary code*, *mobile data* and a *trip marker* per journey. Agents have to be protected against passive, reading attacks, and active attacks that modify an agent's functionality or even fully destroy it. This paper concentrates on the route and its protection against attacks performed by one single context as well as on attacks performed by collusions of cooperating malicious contexts.

4 Protecting the Initial Route

When an agent migrates from working context c_i to working context c_{i+1}, all its objects are encrypted and thus protected against passive attacks. Before starting an agent, (except in very special cases [9]) a working context has to decrypt parts of the agent and thus make the agent vulnerable against active or passive attacks. Thus, the agent's route should be protected as much as possible.

An unprotected route $r = ip(c_1) \mid\mid \ldots \mid\mid ip(c_n)$ is a concatenated list of Internet addresses $ip(c_i)$. To abort an agent's journey, each site has to know the Internet address $ip(h)$ of the home context h, which therefore is stored in plaintext separate from the protected route.
The home context h signs data relevant for each working context to be visited by means of a signature S and encrypts the agent's route using an asymmetrical encryption method E and the i-th working context's public key e_i for $i = 1, \ldots, n$.[3] Applying encryption in a manner as it is known from onion routing network [11] for achieving untraceable communication h composes

$$\begin{aligned}
r = \ & E_{e_1}\Big[h, ip(c_2), S_h\big(h, ip(c_1), ip(c_2), t, E_{e_2}[\ldots]\big), \\
& E_{e_2}\Big[ip(c_1), ip(c_3), S_h\big(ip(c_1), ip(c_2), ip(c_3), t, E_{e_3}[\ldots]\big), \\
& \vdots \\
& E_{e_{n-1}}\Big[ip(c_{n-2}), ip(c_n), S_h\big(ip(c_{n-2}), ip(c_{n-1}), ip(c_n), t, E_{e_n}[\ldots]\big), \\
& E_{e_n}\Big[ip(c_{n-1}), EoR, S_h\big(ip(c_{n-1}), ip(c_n), EoR, t\big)\Big]\ldots\Big]\Big]
\end{aligned} \tag{1}$$

[2] In general the communication protocol itself automatically provides a receiver with the address of the sender.

[3] The ALOHA-software package uses RSA as asymmetrical encryption E. The signature S is based on a combination of SHA and RSA. We suppose that problems according to generation of asymmetrical key-pairs, certification and distribution of public keys are solved.

Thereby, t denotes a trip marker which is unique for each agent's journey and the EoR entry indicates the end of the route. Thus, the route contains the encrypted addresses of all working contexts that shall be visited and the signatures to prove the integrity of the route.

During the agent's journey, a working context c_i decrypts the Internet address $ip(c_{i+1})$ of its successor and $ip(c_{i-1})$ of its predecessor, its relevant signature and all the remaining ciphertext by using its private key d_i that corresponds to the public key e_i. Then, each site removes the decrypted address and signature from the agent's route. All other route entries are hidden from the actual working context.

With the help of digital signatures, active attacks can be detected. The signature of the address which is signed by the agent's home context h and presented to the actual working context c_i, proves that the route entry for c_i has not been modified. The influence of Internet address $ip(c_i)$ and trip marker t on the signature guarantees the actual working context that itself is part of the initial route. The predecessor's address $ip(c_{i-1})$ is taken into account for the signature's computation in order to avoid a special collusion attack to be explained later.

The uniqueness of the trip marker t is necessary to prevent replay attacks. Otherwise, a malicious working context could replace the complete route of the actual agent with a copied route of an agent's earlier journey.

With the help of the EoR entry, working context c_n realizes that it itself is the final entry of the agent's route. Via EoR and t in the signature, working context c_n is able to check whether these data have been generated by h and whether it itself was really included into the initial route.

Signatures must be encrypted by the home context as well, otherwise, if the agent carried the signatures in plaintext, under certain circumstances an attacker, who knows t, would be able to reconstruct the complete route by arranging and testing combinations of possible addresses. Such an attack would be feasible if the number of relevant sites is small.

In the following, we discuss methods with regard to active attacks which modify the functionality of an agent's route. Such attacks can be classified into those where the attacker solely tries to cheat without the help of any other working context and those where the attacker are acting in collusion with other dishonest working contexts.

The onion-like signature of the route ensures that all the desired working contexts have to be visited in order that the further route information can be decrypted. The signatures which depend on all instantaneously existing route data allow the detection of attacks as early as possible.

If context c_i receives an agent, then c_i is the only one that can reveal the successor's address; if the signature check is positive, c_i can be sure that

- it was included in the initial route,
- it received the agent from the correct predecessor,
- the successor's address is correct,
- all further data contained in the route are not compromised.

The home context knows the last entry c_n of the agent's route, i.e. the context from which it finally expects the agent. This prevents any other working context in the initial route from returning the agent too early.

If a malicious working context c_i intends to cheat without the help of any other working context, e.g. by deleting entries from the route, by replacing entries in the route or by adding new entries to the route, the next honest working context will detect such an attack immediately.

If c_i deletes ciphertext whose corresponding plaintext refers to a working context c_j with $j > i + 1$, working context c_{i+1} will detect immediately by signature check that the route was compromised. If c_i tries to skip c_{i+1}, it would not be able to reveal the address $ip(c_{i+2})$. So the malicious context c_i is only able to forward the agent to a randomly chosen address.

If c_i adds new addresses or if c_i replaces entries either in plaintext or in ciphertext, it would never be able to use the right signature key, and so the attack would become obvious as early as possible.

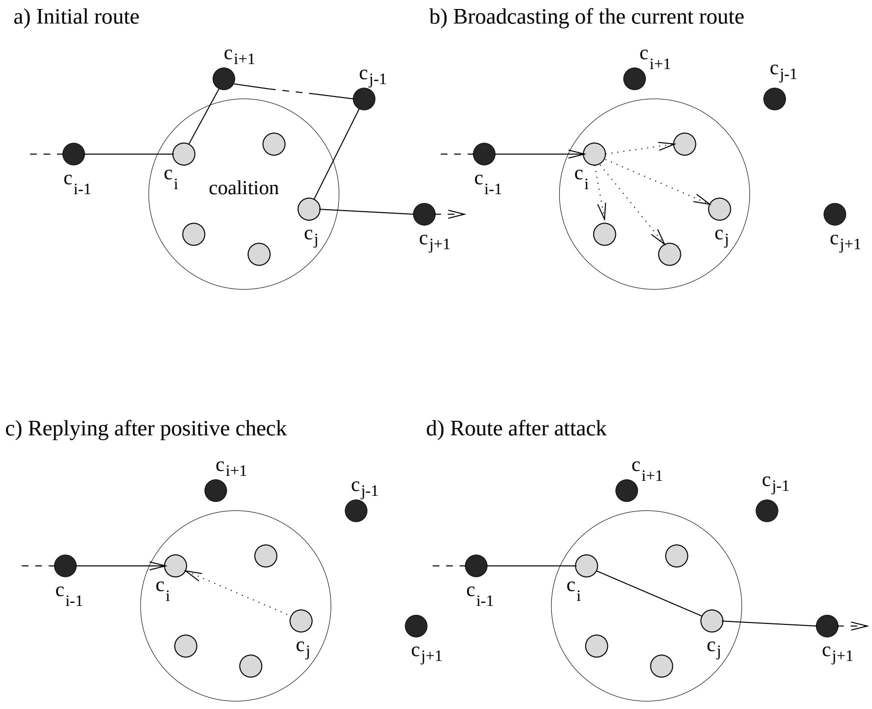

Fig. 1. Attack on an atomic encrypted route under collusion.

In all previous attacks, it was assumed that a malicious working context does not exploit the help of another dishonest working context. In general, a home context does not know if there exist collusions, and which working contexts are acting in collusions.

The performance of the secured route, presented in expression (1), becomes clear when considering collusions. If the route entries are encrypted in an atomic way instead of an onion-like structure and if the initial route contains at least two members c_i,c_j of a colluding group, which are not adjacent $(j > i+1)$, the first visited dishonest context c_i could act in the following way: having obtained the agent, c_i broadcasts copies of the current route to its accomplices. Afterwards, the accomplices try to reveal by sequential decryption if they are members of the initial route. If accomplice c_j obtains a reasonable result after decryption of the presented ciphertexts, c_j found itself as a valid member of the initial route. Now, c_j informs c_i that it is also member of the agent's route and c_i forwards the agent to c_j by skipping all those route entries in the initial route lying in between c_i and c_j [see fig. 1]. No other honest working context c_k, $j < k < n$, would ever be able to detect this attack.

Skipping honest contexts exploiting the power of collusions can be avoided if the route is secured like in expression (1). In an onion-like protection scheme, a malicious working context c_i is only able to reveal more than one succeeding route entries $c_{i+1}, \ldots, c_m$ if all these adjacent contexts belong to the colluding group. In this case, a copy of the route can be forwarded context by context as long as the succeeding context is member of the colluding group.[4] Then, the last member of the colluding group c_m found in that way is able to order the agent from c_i and can forward it to the honest context c_{m+1} which is not able to detect that contexts $c_{i+1}, \ldots, c_{m-1}$ have been skipped.

Even if such an attack is possible, accomplices $c_i, \ldots, c_{m-1}$ are not motivated to act in the described way. In contrast to the atomic encrypted approach, no honest context can become the victim of such an attack.

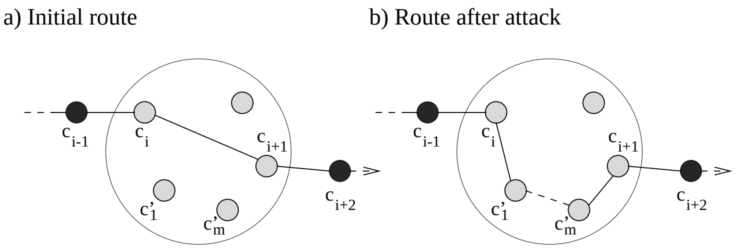

Fig. 2. Attack on onion-like protected route under collusion.

[4] The probability for a malicious context to find one of its accomplices as successor in the initial route depends on the number of members in the colluding group, the total number of contexts in the whole agent system and, when using the atomic encrypting approach, the current length of the route. In most cases, with an onion-like protected route, this probability is significantly smaller than in case of atomic encryption.

Instead of skipping dishonest contexts, malicious contexts may try to add new dishonest contexts to the initial route. If there are two adjacent malicious contexts c_i and c_{i+1} in the initial route, c_i can forward the agent via new dishonest contexts $c'_1, \ldots, c'_m$ to c_{i+1} [see fig. 2]. This attack can not be detected by the following honest context c_{i+2}. But again, no honest context can become the victim of such an attack.

The influence of the predecessor's address becomes obvious if one considers a similar attack. Because of the dependence of the signature from the predecessor's address, a malicious context c_i is not able to forward the agent via accomplices $c'_1, \ldots, c'_m$ to an honest context c_{i+1} [see fig. 3].

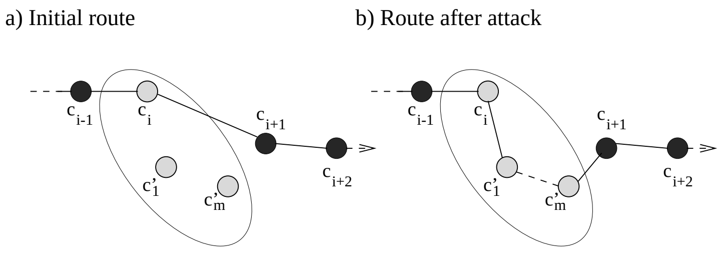

Fig. 3. Detectable attack on onion-like protected route under collusion.

To sum it up and to illustrate the strength of the protection scheme, neither

– skipping of honest working contexts, nor
– replacing honest working contexts by new contexts, nor
– adding honest contexts

is feasible without being detected afterwards, either by an honest working context or by the home context.

Of course, visited working contexts are able to exchange information about agents (predecessors, successors) *a posteriori* to perform route analysis and to obtain information about the visited contexts. For example, a malicious working context c_i can broadcast to all its accomplices that it was visited by a concrete agent with predecessor c_{i-1} and successor c_{i+1}. If the agent later visits a collaborating context c_j, this context can reply its predecessor c_{j-1} and successor c_{j+1} to context c_i. The larger the number of members in the colluding group, the higher the probability becomes that at least one accomplice is member of the initial route. But with growing probability this attack becomes more and more costly, because of the increasing number of candidates to which the agent information has to be forwarded. Without hiding the agent's identity such an attack can never be prevented by means of cryptography.

To summarize the properties of the presented scheme; onion-like encryption provides concealment of route entries at a maximum degree. Without collusions, every attack can be detected as early as possible by the next visited working context. But even under collusions, no honest context can be skipped and no maliciously added or replaced honest working contexts can be visited without being noticed. If the home context receives its agent without an error message, it can be sure that its agent visited all honest contexts in the initial route, and as many addresses as possible are kept secret. Furthermore, every visited honest working context can verify if it is a member of the initial route.

In the following we will examine cases in which routes are legally extended during an agent's journey.

5 Extending the Initial Route

If for providing its service a context c_i on the agent's initial route needs the cooperation with other contexts $c_1^X, \dots, c_m^X$, then working context c_i should be allowed to extend the initial route in a legal way. Of course c_i should not be allowed to delete unvisited entries from the initial route.

To protect the home context's interests, the new entries do not include any confidential information.[5] On the other hand, c_i may be interested in protecting its new entries. Let $c_1^X, \dots, c_m^X$ be these new entries. Like a home context, c_i encrypts the new route extension and includes the route extension as a prefix to the current route r:

$$\begin{aligned} r^X = \; & E_{e_{X1}}\Big[ip(c_i), ip(c_2^X), S_i\big(ip(c_i), ip(c_1^X), ip(c_2^X), t, E_{e_{X2}}[\dots], r\big), \\ & E_{e_{X2}}\Big[ip(c_1^X), ip(c_3^X), S_i\big(ip(c_1^X), ip(c_2^X), ip(c_3^X), t, E_{e_{X3}}[\dots], r\big), \\ & \vdots \\ & E_{e_{Xm-1}}\Big[ip(c_{m-2}^X), ip(c_m^X), S_i\big(ip(c_{m-2}^X), ip(c_{m-1}^X), ip(c_m^X), t, E_{e_{Xm}}[\dots], r\big), \\ & E_{e_{Xm}}\Big[ip(c_{m-1}^X), EoX, S_i\big(ip(c_{m-1}^X), ip(c_m^X), EoX, t, r\big)\Big] \dots \Big]\Big] \; || \; r \end{aligned} \qquad (2)$$

In contrast to the protection scheme of the initial route, the signature in the extension of the route depends on the extension and on the current initial route r. The new parameter EoX indicates the end of extension. If c_i extends a route it must not delete its successor and the corresponding signature from the current initial route.

Having visited all sites $c_1^X, \dots, c_m^X$ of a route extension, the presented concept provides that the agent returns to c_i [see fig. 4].

[5] The new contexts get just informed that c_i is on the initial route.

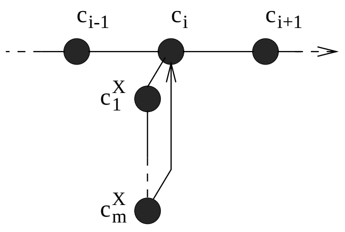

Fig. 4. Extending a route.

When the agent returns to c_i, the context obtains its original part of the initial route and can now decrypt its successor's address and check the signature. If it does not receive the agent from the expected context c_m^X, then it knows that something illegal happened during the agent's journey at the extended route.

The route extension inherits the protection properties of the scheme described in section 4. All signatures in the route extension stem from c_i and the route extension ends with an *EoX* entry. If a context receives such an *EoX* entry it sends the agent back to c_i. To detect attacks as early as possible, the remaining ciphertexts of the initial route are included into c_i's signature. Thus, when the agent returns from c_m^X to c_i, working context c_i can be sure, that all honest sites of the route extension have been visited.

The concept of route extension may be extended: each context of a route extension may be allowed to extend the route extension itself as well. In all these cases the presented scheme guarantees that either all honest sites in the initial route as well as all allowed route extensions are visited, or the journey is explicitly aborted because of an attack or a not accessible site.

6 Unreachable Working Contexts

In many systems, the gain of technical data protection conforms with a loss of functionality and flexibility. If a working context c_{i+1} is unavailable, because it terminated regularly/irregularly or the network connection was interrupted, then context c_i, that can decrypt only the Internet address $ip(c_{i+1})$, may not just skip over context c_{i+1}. In such a case, context c_i has to choose one of the following strategies:

- As long as c_{i+1} is unreachable, the agent waits in c_i.
- The agent aborts its journey and migrates back to its home context.
- Having tried to reach c_{i+1} for a certain time, the agent migrates back to its home context.

When an agent aborts its journey, the reason should become obvious to its home context. When the probability to reach a context falls below a certain level, the route protection mechanism should be changed to become more flexible, though from the data protection point of view this may be disadvantageous.

7 Conclusions

In this paper we presented a method for the concealment of an agent's route information. Furthermore, we showed that our method resists all presented attacks performed by a single malicious context. But even under attacks of malicious contexts acting in collusion no honest working context can become a victim. Thus, when receiving its agent without an error message the home context can be sure that all honest contexts have been visited. All working contexts, and naturally also the honest ones, can detect if the home context intended its agent to visit them.

Additionally, in cases a working context needs the cooperation of other contexts for providing a service, the method allows legal route extensions. If a context extends an agent's initial route, having visited all sites of the route extension the agent returns to the context that has extended the route, and proceeds on its initial route. Following this pattern, one can develop several levels of route extension and use the described method for protecting the agent's route against attacks.

8 Future Work

To precisely identify that contexts which ran the attack, more in-depth research is needed. We are also extending our route protection mechanisms to other agent components. Mobile data can be handled similar to routes: at least parts of them must only become accessible to specific contexts. In contrast, the agent's binary code has fully to be decrypted at each site. By using checksums and a kind of third party protocol [13],[14], attacks can be detected and afterwards malicious contexts can be forced to forward correct binary code. Nevertheless such a protocol can not verify if a working context really started an agent. Maybe detection objects [15] ensure this.

References

1. F. Mattern: 'Mobile Agenten', it + ti - Oldenbourg Verlag, 1998, 4, pp.12-17.
2. W. Ernestus, D. Ermer, M. Hube, M. Köhntopp, M. Knorr, G. Quiring-Kock, U. Schläger, G. Schulz: 'Datenschutzfreundliche Technologien', DuD 21, 1997, 12, pp.709-715.
3. S. Berkovits, J. Guttman, V. Swarup: 'Authentication for Mobile Agents', in 'Mobile Agents and Security', Proceedings, Springer Verlag, LNCS 1419, 1998, pp.114-136.
4. D. Chess: 'Security Issues in Mobile Code Systems', in 'Mobile Agents and Security', Proceedings, Springer Verlag, LNCS 1419, 1998, pp.1-14.
5. D. Westhoff: 'AAPI: an Agent Application Programming Interface', Informatikbericht 247-12/1998, FernUniversität Gesamthochschule in Hagen 1998.
6. G. Vigna: 'Cryptographic Traces for Mobile Agents', in 'Mobile Agents and Security', Proceedings, Springer Verlag, LNCS 1419, 1998, pp.138-153.

7. W.M. Farmer, J. Guttman, V. Swarup: 'Security for Mobile Agents: Authentication and State Appraisal', in 'Proc. of the 4th European Symp. on Research in Computer Security', Springer Verlag, LNCS 1146, 1996, pp.118-130.
8. U.G. Wilhelm: 'Cryptographically protected Objects'. Technical report, Ecole Polytechnique Federale de Lausanne, Switzerland, 1997.
9. T. Sander, C. Tschudin: 'Protecting Mobile Agents Against Malicious Hosts', in 'Mobile Agents and Security', Proceedings, Springer Verlag, LNCS 1419, 1998, pp.44-60.
10. F. Hohl: 'Time Limited Blackbox Security: Protecting Mobile Agents from Malicious Hosts', in 'Mobile Agents and Security', Proceedings, Springer Verlag, LNCS 1419, 1998, pp.92-113.
11. M.G. Reed, P.F. Syverson, D.M. Goldschlag : 'Anonymous Connections and Onion Routing', in 'IEEE Journal on Selected Areas in Communication - Special Issue on Copyright and Privacy Protection', Vol. 16, No. 4, 1998, pp.482-494.
12. R. Rivest, A. Shamir, L. Adleman: 'A Method for Obtaining Digital Signatures and Public-Key Cryptosystems' in 'Communication of ACM', Volume 21, Number 2, February 1978, pp.120-126.
13. N. Asokan, V. Shoup, M. Waidner: 'Asynchronous Protocols for Optimistical Fair Exchange', 1998 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, Los Alamitos 1998, pp.86-99.
14. Y. Han: 'Investigation of Non-repudiation Protocols', in 'Information Security and Privacy', Proceedings of ACISP'96, Springer Verlag, LNCS 1172, 1996, pp.38-47.
15. C. Meadows: 'Detecting Attacks on Mobile Agents', Center for High Assurance Computing Systems, Naval Research Laboratory, Washington DC, 1997.