Proceedings of the 5th International Workshop on Mobility in Databases and Distributed Systems (Workshop of DEXA2002), pp 734-738, IEEE Computer Society Press, September 2002, Aix-en-Provence, France

Several Structures for Protecting the Routes of Mobile Agents Dispatched in Parallel

Yan Wang, Kian–Lee Tan, Xiaolin Pang
Department of Computer Science
National University of Singapore
3 Science Drive 2, Singapore 117543
{ywang, tankl, panxiao}@comp.nus.edu.sg

Abstract

For mobile agents to be effective in practice, they have to be securely and efficiently deployed. In this paper, we propose and discuss several route structures and methods for mobile agents that are dispatched in parallel. These schemes can protect the route information against malicious hosts and facilitate the dispatching of a large number of agents in parallel efficiently. Under non-collusion attacks, these methods expose minimal addresses to a host for dispatching other mobile agents. Moreover, they can detect possible attacks as early as possible and enforce the predefined dispatch order.

1. Introduction

The use of mobile agents for applications on the Internet is gaining increasing attention. Mobile agents are autonomous, mobile and flexible, and can facilitate parallel processing. In some scenarios, a mobile agent can act on behalf of its owner to migrate through the distributed network and send results back [l,2].

In many applications, the security and efficiency of the mobile agent approach are of great concern [3,4,5]. For example, in e-commerce, there are a lot of online e-shops that may provide the same kinds of goods. It is too time-consuming if one agent is dispatched to visit all or most of them. Parallel models are needed to improve the efficiency. Moreover, since all e-shops are competitors, the initial route information of agents should be protected against potential malicious hosts (e-shops) en route from tampering with the predefined route information. Otherwise, a malicious host may prevent other e-shops from being visited so that its offer and service may become the best for the customer to accept.

Tamper-poof device [6] is hardware-based mechanisms for protecting mobile agents and hosts. Software-based approaches involve encrypted functions [7,8] and digital signatures with proxy certificates [9] etc. Several secure route structures are presented in [10] for protecting a serially migrating agent.

This paper focuses on how to protect the route information for mobile agents dispatched in parallel. We present several route structures and discuss their security performances. Though only initial route information is protected, these structures can be easily extended to protect other information, such as the task, privilege etc. In this paper, we employ well-known public-key encryption and signature generating algorithms.

2. Backgrounds

2.1 Public-key cryptography, hash function and signature

Public-key cryptography uses two different keys [11]. One is the *Public Key* and the other is the *Secret Key*. The public key can be distributed publicly to other parties via digital certificates while the secret key should be kept by the owner. Suppose Alice wants to send a message m to Bob securely. Alice can use the public key of Bob, P_B, to encrypt m as $P_B[m]$ and send it to Bob. Upon receiving the ciphertext, Bob can use its secret key S_B to decrypt the message as $m=S_B[P_B[m]]$. RSA is one of the most famous public-key cryptographies [12].

Secret key can also be used to generate a digital signature [11]. If Bob wants to send Alice a document D, he can generate the signature as $S_B(D)$ and sends it to Alice with the document. With the signature, Alice can use Bob's public key P_B to check the data integrity of the document. Generally, when generating a signature on a long document, a one-way hash function, denoted *as* $H(x)$, can be used to save time, which can operate on an arbitrary-length long message m and returns a fixed-length hash value h, where $h = H(m)$. In this way, the signature, denoted as $sig= S_B(H(D))$, will be shorter.

2.2 Basic binary dispatch model

In this section, we briefly introduce the basic binary dispatch model. It is a typical parallel dispatch model where each *parent agent* can dispatch two *child agents* resulting in a binary tree structure as shown in Figure 1. Clearly, the model can be easily generalized to *m-branch* $(m \geq 2)$ parallel dispatch model as presented and discussed in [13].

We term an agent as a *Master Agent* if it is created at the home host and is responsible for dispatching a pool of mobile agents to remote hosts. We call an agent a *Worker Agent* (WA) if its sole responsibility is to perform simple tasks assigned to it, e.g. accessing local data. If a WA also dispatches other worker agents besides performing the task of local data accessing, it is called a *Primary Worker Agent* (PWA).

As shown in Figure 1, suppose master agent A_0 has to dispatch 16 agents to 16 hosts (e.g. agent A_i to host H_i). Now, 16 mobile agents can be divided into 2 groups led by two PWAs, say A_1 and A_9. When agents A_1 and A_9 are dispatched to H_1 and H_9 respectively, each of them has 8 members including itself. For A_1 at layer L_1, it will dispatch A_5 and distribute 4 members to it. After that A_1 will transit to the same layer (i.e., L_2) as A_5, which is called a *virtual dispatch* costing no time. Now A_1 has 4 members only. Following the same process, A_1 dispatches A_3 and A_2 successively. During all these processes, A_1 always resides at H_1 without any migration. At the same time when A_1 dispatches A_5, A_0 dispatches A_9 to H_9 to activate all agents in parallel in another branch. At last, after all dispatch tasks have been completed, A_1 becomes a WA and starts its local data-accessing task at H_1. The whole dispatch process can be illustrated by a *dispatch tree*, as shown in Figure 1. As a whole, the model benefits from the parallel dispatches by different PWAs at different hosts. When there are $n=2^h$ mobile agents and T is the average time for dispatching a mobile agent, $(h+1)T$ will be the time for dispatching n mobile agents. So, the dispatch complexity will be $O(log_2^n)$ [13]. In contrast, a serial migration model has a complexity of $O(n)$.

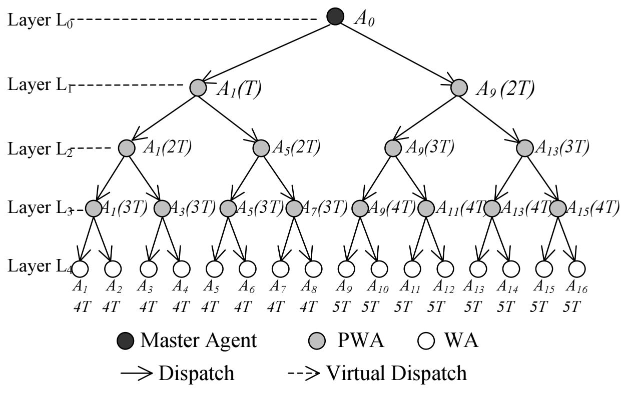

Figure 1 Dispatch tree with 16 mobile agents

3. Secure route structures

Following the binary dispatch model, if no secure route structure is provided, the route information of an agent will be revealed to the hosts it visits. Attacks can be easily mounted without being detected. These attacks include route insert attack, route delete attack, replay attack, wrong dispatch attack or dispatch skip attack. In the following sections, we adopt the combination of public-key encryption and signature schemes to propose several secure route structures. We will also illustrate the schemes and compare their security properties. In all the proposed schemes, the route is generated by A_0 at H_0 before any dispatch is performed. Of course the route cannot be encrypted by the public key of the dispatched agent since this will disclose the secret key of the agent to visited hosts. Thus, the route is encrypted using public keys of the corresponding hosts that will be visited. A carried encrypted route can be decrypted with the assistance of the current host that may help to dispatch child agents when an agent arrives there. The agent can verify the validity of plaintext using included signature. The host can delete a used route after the corresponding dispatch is successful. Some basic ideas for secure route structures of serially migrated agents were presented in [10]. Here we address secure route structures for binary dispatch, which are more complicated than serial migration. In the following, we assume that there exists a secure environment including the generation, certification and distribution of public keys and each host can know the authentic public key of other hosts.

3.1 Atomic route and atomic signature

Route Structure (I):

Suppose agent A_x is dispatched to current host CH, its dispatch layers from CH are $L_1, L_2, \ldots, L_m$ ($m\geq1$). The route of A_x is:

(i) $r= r^{L1}(CH)\|\ r^{L2}(CH)\|\ r^{L3}(CH)\| \ldots \|\ r^{Lm}(CH)$

(ii) if $1\leq i<m$, where A_x is a PWA, $r^{Li}(CH)=$ P_{CH}[isPWA, ip(RH), r(RH), t, S_{H0}(isPWA, ip(PH), ip(CH), ip(RH), r(RH), t)]

(iii) if i=m, A_x should be a WA, $r^{Lm}(CH)=P_{CH}$[isWA, ip(H_0), t, S_{H0}(isWA, ip(PH), ip(CH), ip(H_0), t)]

where the symbol || denotes the concatenation of data; PH is the parent host of CH; RH is the right child host of CH; H_0 denotes the home host where A_0 resides. isPWA is the token meaning the agent is a PWA while isWA is the token meaning the agent is a WA. P_{CH} is the public key of CH; S_{H0} is the secret key of H_0.

For A_1 in Figure 1, its route is

$r=r^{L1}(H_1)\|\ r^{L2}(H_1)\|\ r^{L3}(H_1)\|\ r^{L4}(H_1)$

where

$r^{L1}(H_1)=P_{H1}$[isPWA, ip(H_5), r(H_5), t, S_{H0}(isPWA, ip(H_0), ip(H_1), ip(H_5), r(H_5), t)];

$r^{L2}(H_1)=P_{H1}$[isPWA, ip(H_3), r(H_3), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_3), r(H_3), t)];

$r^{L3}(H_1)=P_{H1}$[isPWA, ip(H_2), r(H_2), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_2), r(H_2), t)];

$r^{L4}(H_1)=P_{H1}$[isWA, ip(H_0), t, S_{H0}(isWA, ip(H1), ip(H_1), ip(H_0), t)]

In this case, the route of a PWA is the concatenation list of routes in different layers while each route contains the IP address of the right child host RH (say ip(RH)), the token showing the agent is a PWA or a WA, the route for the right child agent (say r(RH)) and corresponding signature. Also the addresses of the parent host PH, current host CH and right child host RH are included in the signature. t is the unique timestamp when the route is generated at home host H_0. With t, any replay attack that should use a signature copy will not be successful since the destination host can verify the signature.

Since the signature is generated at host H_0 using its secret key, it is not possible to forge it. On this basis, route insert attack will not be successful. The addresses in the signature can prove the valid path of the agent for any visited hosts. So the attack to dispatch an agent to a wrong host can be easily detected by the destination host.

However, the route structure cannot help to prevent route deletion attack since all routes do not have any dependence of each other. Suppose $r^{L1}(H_1)$ is deleted *before* decryption, A_1 is not likely to find it with the rest of the routes. Thus, A_5 will not be dispatched. In such a case, it may cause investigation against H_1 since A_0 will not get any information from A_5 to A_8, which are in the same group led by A_5.

In addition, even if the route is not deleted, the dispatch skip attack can be partially successful since the dispatch process can be actually controlled by the host. In our model we suppose H_{x+1} should send a confirmation message as $S_{Hx+1}(H(Code_{Ax+1}, ip(H_x), ip(H_{x+1}), t_r))$ to H_x after A_{x+1} is successfully dispatched to H_{x+1} from H_x at time t_r. With this mechanism, dispatch skip attack can be detected after the investigation conducted by A_0 since a malicious host cannot show the confirmation message generated by the destination host.

Moreover, with structure (I), dispatch disorder attack may be successful since routes at different layers are all encrypted by the public key of CH, say P_{CH}. The dispatch order can be arbitrarily controlled by CH since it performs decryption and transfers the plaintext to the agent. As long as all dispatches are performed, it is not easy to detect this kind of attack but the whole dispatch performance will be affected (e.g., H_1 makes A_1 dispatch A_2, A_3 and A_5 in sequence after having performed its local data-accessing task).

3.2 Atomic route and nested signature

Route Structure (II):

Suppose agent A_x is dispatched to current host CH, its dispatch layers from CH are $L_1, L_2, \ldots, L_m$. The route of A_x is:

(i) $r= r^{L1}(CH)\|\, r^{L2}(CH)\|\, r^{L3}(CH)\|\, \ldots\|r^{Lm}(CH)$

(ii) if $1\le i<m$, A_x is a PWA, $r^{Li}(CH)= P_{CH}[isPWA, ip(RH), r(RH), t, S_{H0}(isPWA, ip(PH), ip(CH), ip(RH), r(RH), r^{Li+1}(CH), \ldots, r^{Lm}(CH), t)]$

(iii) if i=m, A_x is a WA, $r^{Lm}(CH)= P_{CH}[isWA, ip(H_0), t, S_{H0}(isWA, ip(PH), ip(CH), ip(H_0), t)]$

For A_1 in Figure 1, its route is $r=r^{L1}(H_1)\|\, r^{L2}(H_1)\|\, r^{L3}(H_1)\|\, r^{L4}(H_1)$

where $r^{L1}(H_1)=P_{H1}[isPWA, ip(H_5), r(H_5), t, S_{H0}(isPWA, ip(H_0), ip(H_1), ip(H_5), r(H_5), r^{L2}(H_1), r^{L3}(H_1), r^{L4}(H_1), t)]$

$r^{L2}(H_1)=P_{H1}[isPWA, ip(H_3), r(H_3), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_3), r(H_3), r^{L3}(H_1), r^{L4}(H_1), t)]$

$r^{L3}(H_1)=P_{H1}[isPWA, ip(H_2), r(H_2), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_2), r(H_2), r^{L4}(H_1), t)]$

$r^{L4}(H_1)=P_{H1}[isWA, ip(H_0), t, S_{H0}(isWA, ip(H_1), ip(H_1), ip(H_0), t)]$

In structure (II), route deletion attack can be partially detected since $r^{L2}(H_1)$, $r^{L3}(H_1)$ and $r^{L4}(H_1)$ appear in the signature of $r^{L1}(H_1)$, deletion of subsequent routes can be found by agent A_1 when verifying the signature of a front route. But if $r^{L1}(H_1)$ is deleted *before* it is used or decrypted, the attack cannot be found since the front route does not appear in the signature of subsequent routes. Particularly, similar to structure (I), since all routes are in the concatenation list, the deletion attack can be mounted by either the parent host or current host. This will make the investigation too complicated and difficult.

Meanwhile, the dispatch disorder attack cannot be detected as long as all child agents are dispatched.

3.3 Nested route and atomic signature

Route Structure (III):

Suppose agent A_x is at current host CH, the layers are $L_1, L_2, \ldots, L_m$

(i) $r(CH)= r^{L1}(CH)$

(ii) if $1\le i<m$, A_x is a PWA, $r^{Li}(CH)= P_{CH}[isPWA, ip(RH), r(RH), t, S_{H0}(isPWA, ip(PH), ip(CH), ip(RH), r(RH), t), r^{Li+1}(CH)]$

(iii) if i=m, A_x is a WA, $r^{Lm}(CH)= P_{CH}[isWA, ip(H_0), t, S_{H0}(isWA, ip(PH), ip(CH), ip(H_0), t)]$

In structure (III), the subsequent route $r^{Li+1}(CH)$ can be obtained only after route $r^{Li}(CH)$ is decrypted. Likewise, $r^{Li+2}(CH)$ is included in route $r^{Li+1}(CH)$. Before decryption, any changes with the route can be found. But $r^{Li+1}(CH)$ can be deleted *after* route $r^{Li}(CH)$ is decrypted since route $r^{Li+1}(CH)$ does not appear in the signature of $r^{Li}(CH)$. After decryption, the current route information in plaintext including the signature can also be deleted that cannot be found by the agent. This will cause dispatch skip attack that can only be detected by A_0. Meanwhile, dispatch disorder attack can be successful if H_x decrypts all routes and gives A_x route information in an arbitrary sequence.

For A_1 in Figure 1, its route is

$r= P_{H1}[isPWA, ip(H_5), r(H_5), t, S_{H0}(isPWA, ip(H_0), ip(H_1), ip(H_5), r(H_5), t), P_{H1}[isPWA, ip(H_3), r(H_3), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_3), r(H_3), t), P_{H1}[isPWA, ip(H_2), r(H_2), t, S_{H0}(isPWA, ip(H_1),$

$ip(H_1), ip(H_2), r(H_2), t), P_{H1}[isWA, ip(H_0), t, S_{H0}(isWA, ip(H_1), ip(H_1), ip(H_0), t)]\ ]\ ]\ ]$

3.4 Nested route and nested signature (I)

Route Structure (IV):

Suppose agent A_x is dispatched to current host CH, its dispatch layers from CH are $L_1, L_2, \ldots, L_m$. The route of A_x is:

(i) $r= r^{L1}(CH)$

(ii) if $1\le i<m$, A_x is a PWA, $r^{Li}(CH)= P_{CH}[isPWA, ip(RH), r(RH), r^{Li+1}(CH), t, S_{H0}(isPWA, ip(PH), ip(CH), ip(RH), r(RH), r^{Li+1}(CH), t)]$

(iii) if i=m, A_x is a WA, $r^{Lm}(CH)= P_{CH}[isWA, ip(H_0), t, S_{H0}(isWA, ip(PH), ip(CH), ip(H_0), t)]$

In structure (IV), since both the route and signature are in nested structure, any changes with the subsequent route before decryption can be found. But since all routes are encrypted by P_{H1}, the dispatch disorder attack remains unsolved. Similarly, if host CH only transfers to the agent subsequent routes and deletes front routes *after* decrypting all routes, the agent cannot find it since front routes cannot be included in subsequent routes. This attack can only be confirmed after the investigation conducted by A_0.

As an example, for A_1 in Figure 1, its route is

$r=P_{H1}[isPWA, ip(H_5), r(H_5), t, S_{H0}(isPWA, ip(H_0), ip(H_1), ip(H_5), r(H_5), P_{H1}[isPWA, ip(H_3), r(H_3), \ldots], t), P_{H1}[isPWA, ip(H_3), r(H_3), S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_3), r(H_3), P_{H1}[isPWA, ip(H_2), r(H_2), \ldots], t), P_{H1}[isPWA, ip(H_2), r(H_2), t, S_{H0}(isPWA, ip(H_1), ip(H_1), ip(H_2), r(H_2), P_{H1}[isWA, ip(H_0), \ldots], t), P_{H1}[isWA, ip(H_0), t, S_{H0}(isWA, ip(H_1), ip(H_1), ip(H_0), t)]\ ]\ ]\ ]$

3.5 Nested route and nested signature (II)

Based on structure (IV), structure (V) can provide the capability to restrict the dispatch order to be strictly followed while ensuring other security properties. It makes the subsequent route for the subsequent right dispatch included in front right route. Only after a dispatch is successful, the host and agent can get a route for the next dispatch. The idea is introduced by an example as follows.

For instance, the route of A_1 in structure (V) is:

$r=P_{H1}[isPWA, ip(H_5), r(H_5), t, S_{H0}(isPWA, ip(H_0), ip(H_1), ip(H_5), r(H_5), t)]$

where $r(H_5)=P_{H5}[isPWA, ip(H_7), r(H_7), P_{H1}[isPWA, ip(H_3), r(H_3), t, S_{H0}(is(H_1), ip(H_3), r(H_3), t)], t, S_{H0}(isPWA, ip(H_1), ip(H_5), ip(H_7), r(H_7), P_{H1}[ip(H_3), r(H_3), t, S_{H0}(ip(H_3), r(H_3), t)], t)]$

Note route $r'= P_{H1}[isPWA, ip(H_3), r(H_3), t, S_{H0}(is(H_1), ip(H_3), r(H_3), t)]$ is included in $r(H_5)$. It is used for dispatching A_3 from A_1 and it can only be obtained after A_5 is successfully dispatched. At H_1, after decrypting route r, A_1 can dispatch agent A_5 to H_5 encapsulating route $r(H_5)$ to it. When H_5 has successfully received A_5, it will send a message to H_1 confirming that the dispatch is successful and returning the route r' to H_1 for A_1's next dispatch. r' is a ciphertext that cannot be decrypted by H_5. The message from H_5 to H_1 is:

$msg\_5\_1= P_{H1}[isPWA, ip(H_3), r(H_3), S_{H0}(ip(H_1), ip(H_3), r(H_3), t)]||\ S_{H5}(H(P_{H1}[isPWA, ip(H_3), r(H_3), S_{H0}(ip(H_1), ip(H_3), r(H_3), t)], Code_{A5}, t_2))$

where t_2 is the time when H_5 receives A_5; H is the hash function; encrypted route $P_{H1}[isPWA, ip(H_3), r(H_3), S_{H0}(ip(H_1), ip(H_3), r(H_3), t)]$ is originally included in route $r(H_5)$.

From msg_5_1, A_1 knows after decryption it should be a PWA and dispatch A_3 to H_3 encapsulating $r(H_3)$ to it. Likewise, after A_3 is successfully dispatched, A_1 will know it should dispatch A_2 to H_2 after decrypting the message msg_3_1 from H_3. The rest routes are as follows:

$r(H_3)=P_{H3}[isPWA, ip(H_4), r(H_4), P_{H1}[isPWA, ip(H_2), r(H_2), t, S_{H0}(ip(H_1), ip(H_2), r(H_2), t)], t, S_{H0}(isPWA, ip(H_1), ip(H_3), ip(H_4), r(H_4), P_{H1}[isPWA, ip(H_2), r(H_2), t, S_{H0}(ip(H_1), ip(H_2), r(H_2), t)], t)]$

$msg\_3\_1= P_{H1}[isPWA, ip(H_2), r(H_2), S_{H0}(ip(H_1), ip(H_2), r(H_2), t)]||S_{H5}(H(P_{H1}[isPWA, ip(H_2), r(H_2), S_{H0}(ip(H_1), ip(H_2), r(H_2), t)], Code_{A3}, t_3))$

$r(H_2)=P_{H2}[isWA, ip(H_0), P_{H1}[isWA, ip(H_0), S_{H0}(ip(H_1), ip(H_0), t)], S_{H0}(isWA, ip(H_1), ip(H_2), ip(H_0), P_{H1}[isWA, ip(H_0), S_{H0}(ip(H_1), ip(H_0), t)], t)]$

$msg\_2\_1= P_{H1}[isWA, ip(H_0), S_{H0}(ip(H_1), ip(H_0), t)]||\ S_{H5}(H(P_{H1}[isWA, ip(H_0), S_{H0}(ip(H_1), ip(H_0), t)], Code_{A2}, t_4))$

After A_2 is successfully dispatched, H_2 will send the route $P_{H1}[isWA, ip(H_0), S_{H0}(ip(H_1), ip(H_0), t)]$ via msg_2_1 to H_1 and A_1 who will hereby become a WA.

Consequently, from structure (V), the dispatch sequence will be strictly followed in a non-collusion environment. If current host wants to hide any information, it will not succeed since it can be found by current agent instead of A_0.

4. Conclusions

The above-proposed secure route structures ensure some basic security properties. They expose only minimal addresses to a host to perform dispatches. With the improvement of security performances, the computational overhead for route generation may increase too. However, with respect to security, which is the most important issue for mobile agents, the sacrifice on performance is worthy while the dispatch complexity remains $O(log_2^{\,n})$. Moreover, we think the security levels of structure (IV) and (V) are similar. For structure (IV), the dispatch skip attack can be found by A_0 and the dispatch disorder attack can be considered as benign. In addition, it is possible to extend it to include substitute routes so as to make dispatches robust [14]. For structure (V), though the dispatch sequence can be strictly followed, it is not robust since a failed dispatch will cause the failure of all subsequent dispatches. For future work, we will provide robustness mechanism for structure (V). Table 1 summarizes the security properties of the different route structures.

Table 1 Security properties of different route structures

	Route Insert Attack	Replay Attack	Dispatch to Wrong Host H_w	Dispatch Skip Attack	Route Deletion Attack	Dispatch Disorder Attack
I	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by A_0	Any route can be deleted by parent host or current host before decryption. Only A_0 may detect it.	No.
II	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by A_0	Front routes can be deleted by parent host or current host before decryption. Only A_0 may detect it. Deleting a subsequent route can be found by current agent by checking the signature of its front route.	No.
III	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by A_0	An included route can be deleted after decryption.	No.
IV	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by A_0	Yes, by current agent.	No.
V	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by current agent.	Yes, by current agent.	Dispatch order is strictly followed.

"Yes": The attack can be detected.
"No": The attack cannot be detected.

5. Acknowledgement

This work is supported by the NSTB/MOE funded project on Strategic Program on Computer Security (R-252-000-015-112/303).

References:

[1] D. Lange, and M. Oshima, *Programming and Deploying Java Mobile Agents with Aglets*, Addison-Wesley Press, Massachusetts, USA, 1998

[2] S. Papastavrou, G. Samaras. and E. Pitoura, "Mobile Agents for World Wide Web Distributed Database access", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 12, Issue 5 , Sept.-Oct. 2000, pp. 802 –820

[3] T.D. Rodrigo and A. Stanski, "The Evolving Future of Agent-based Electronic Commerce", in *Electronic Commerce: Opportunity and Challenges* (Edited by S. M. Rahman and M.S. Raisinghani), Idea Group Publishing, Hershey, USA, 2000, pp. 337-351

[4] C. Panayiotou, G. Samaras, E. Pitoura and P. Evripidou, "Parallel Computing Using Java Mobile Agents", *Proceedings of 25th Euromicro Conference Special session on Network Computing*, September 1999

[5] V. Varadharajan, Security Enhanced Mobile Agents, *Proceedings of the 7th ACM conference on Computer and communications security***,** November 1 - 4, 2000, Athens, Greece, pp. 200 – 209

[6] U.G. Wilhelm, Cryptographically Protected Objects, Technical Report, Ecole Polytechnique Federale de Lausanne, Switzerland, 1997

[7] T. Sander and C.F. Tschdin, Protecting Mobile Agents Against Malicious Hosts, *Mobile Agents and Security,* LNCS Vol. 1419, Springer-Verlag, 1998, pp44-60

[8] P. Kotzanikolaou, M. Burmester and V. Chrissikopoulos, Secure Transactions with Mobile Agents in Hostile Environments, *ACISP 2000*, LNCS 1841, 2000, pp.289-297

[9] A. Romao and M.M. Sliva, Secure Mobile Agent Digital Signatures with Proxy Certificates, *E-Commerce Agents*, LNAI 2033, 2001, pp.206-220

[10] D. Westhoff, M. Schneider, C. Unger and F. Kenderali, Methods for Protecting a Mobile Agent's Route, *Proceedings of the Second International Information Security Workshop (ISW'99)*, 1999, Springer Verlag, LNCS 1729, pp. 57-71

[11] A. Menezes, P. Oorschot and S.Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996

[12] R.L. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems", *Communications of the ACM*, 1978.

[13] Y. Wang, Dispatching Multiple Mobile Agents in Parallel for Visiting E-Shops, *Proc. of 3rd International Conference on Mobile Data Management (MDM2002)*, IEEE Computer Society Press, Jan. 8-11 2002, Singapore, pp. 61-68

[14] Y. Wang and K.L Tan, A Secure Model for the Parallel Dispatch of Mobile Agents, *Proc. of Third International Conference on Information and Communications Security (ICICS2001)*, Springer-Verlag, LNCS Vol. 2229, Xi'an, China, 13-16 November, 2001, pp. 386-397