accepted by *International Journal of Web Engineering and Technology (IJWET) 2003*

A Study of Securing Route Structures for Mobile Agents Dispatched in Parallel

Yan Wang, Kian-Lee Tan and Yue Wang

Department of Computer Science
National University of Singapore
3 Science Drive 2, Singapore 117543
E-mail: ywang@comp.nus.edu.sg

E-mail: tankl@comp.nus.edu.sg

E-mail: wangy@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 first present and discuss five secure route structures for mobile agents dispatched in parallel. These schemes aim to protect the route information against malicious hosts and facilitate efficient dispatching of a large number of agents, combining public-key encryption and signature schemes and exposing minimal route information to hosts. In term of security, they are improved one by one. In the 5th structure, nested route and signature structures are adopted in order to detect attacks as early as possible. Meanwhile, the feedback based mechanism is adopted that can enforce the dispatch order to be strictly followed. Additionally, a robustness mechanism with substitute routes is provided for skipping temporarily unreachable hosts to activate descendant agents. Finally, we evaluated our models both analytically and empirically.

Keywords: Mobile agent, secure route structure, parallel dispatch model, robust route structure

Biographical notes: Yan Wang received his Bachelor Degree of Engineering, Master Degree of Engineering and Doctorate Degree of Engineering on computer science and technology in 1988, 1991 and 1996 respectively from Harbin Institute of Technology (HIT), China. He ever worked at the Department of Computer Science, City University of Hong Kong as a Senior Research Assistant/Associate in 1997 and 1999. He is currently a Research Fellow of the Department of Computer Science, School of Computing, National University of Singapore (NUS). His research interests cover mobile agent, computer security and electronic commerce.

Kian-Lee Tan, received his Ph.D. in computer science in 1994. He is currently an Associate Professor in the Department of Computer Science, School of Computing, National University of Singapore (NUS). His current research interests include multimedia information retrieval, query processing and optimization in multiprocessor and distributed systems, and database performance, database security and genome databases. He has published numerous papers in conferences such as SIGMOD, VLDB, ICDE and EDBT and two books. Kian-Lee was a Visiting Scientist at IBM's Almaden Research Center, California (Jan 92 -- Jul 92), and CSIRO's Canberra Laboratory, Australia (Jun 94 -- Jun 95). He was a Senior Scientist at the Genome Institute of Singapore (Joint appointment (June 01 -- June 03). Kian-Lee is a member of ACM and an affiliate member of IEEE.

Yue Wang received her honorable degree on computer science in 2001 from the National University of Singapore. She was a Research Associate at the Department of Computer Science, School of Computing, National University of Singapore in 2002. She is currently a PhD student at the Department of Computer Science, Concordia University, Canada.

1. INTRODUCTION

Mobile agents can be employed in many applications [1,2], for example, in e-commerce, they can be used to visit a large number of e-shops to find the best price for certain products. While they can be efficiently dispatched in parallel to improve efficiency [3], there is a greater need to protect the agents against potential malicious hosts (e-shops) en route from tampering with the data/code they are carrying. Otherwise, a malicious host, say an e-shop, may prevent other e-shops from being visited so that its offer and service may become the best offer. Moreover, mechanisms should be provided so that a host is capable of verifying the validity of incoming agents and their carried routes.

While hardware-based mechanisms like tamper-poof devices [4] and secure coprocessors [5] can be used to protect mobile agents and hosts, they are limited in the types of protection they offer. Software-based approaches involve more work, such as encrypted functions [6, 7] and digital signatures with proxy certificates [8]. This paper adopts a software approach to protect the routes that are embedded within an agent. Secure route structure for an individual mobile agent has been discussed in [9,10] to protect the route of a serially migrating agent. However, as discussed in [11], the response time of a serial agent is unacceptable unless the number of hosts (e.g., e-shops) to visit is small. When the number of hosts is large, parallel dispatch is essential for performance reasons. Moreover, it is more essential and complicated to secure the route structures for parallel dispatch.

In this paper, we build on the binary dispatch model [3] to protect the route structures of agents dispatched in parallel. We present five route structures and discuss their security properties. Meanwhile, the confirmation feedback based mechanism is adopted in the dispatch protocol to ensure that the dispatch order is strictly followed, preventing any malicious host from breaking the dispatch sequence and hence worsening the dispatch performance. Thus, we preserve the efficiency of the hierarchical dispatch model while ensuring route security.

In addition, a robustness mechanism with substitute routes is provided. With this mechanism, when a predefined right child host is unreachable, a substitute agent can be dispatched to the substitute host so that the descendant agents in this branch can be activated. The strategy to select substitute hosts is presented to minimize the cost for generating substitute routes. A model for distributing the workload of decrypting multiple substitute routes is also presented.

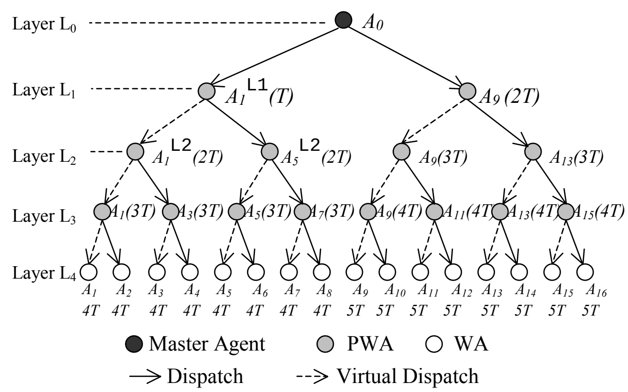

Figure 1 Dispatch tree with 16 mobile agents

In this paper, we employ well-known public-key cryptography scheme, signature generating scheme and X.509 authentication framework [12, 13]. 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.

The rest of this paper is organized as follows. Section 2 reviews the basic binary dispatch model. Section 3 presents five secure route structures for the binary dispatch model. A formal description on the 5th route structure and the corresponding dispatch protocol are presented in Section 4. Section 5 presents the robustness mechanism. In Section 6, after presenting two existing serial models, the complexities of dispatch/migration and route generation of both serial and parallel models are analyzed. Section 7 illustrates the results of experimental study. Finally, Section 8 concludes our work.

2. THE BASIC BINARY DISPATCH MODEL

In this section, we briefly review a typical parallel dispatch model [3] where each *parent agent* can dispatch two *child agents* resulting in a binary tree structure as shown in Figure 1.

We term a stationary agent as a *Master Agent* if it is created at the home host and is responsible for dispatching a pool of mobile agents to pre-found remote hosts. We assume that a home host is an authorized host just like the MASP (Mobile Agent Service Provider) in [3]. We call a mobile 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 (i.e. agent A_i is dispatched to host H_i and H_0 is the home host where A_0 resides). Now, 16 mobile agents are divided into 2 groups led by two PWAs respectively, 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 its *right child agent* A_5 and distribute 4 members to it. A_5 is a PWA responsible for activating its 4 members in binary. After dispatching A_5, A_1 will transfer to layer L_2, 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 the other 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.

To summarize, the tasks of A_1 is to act as a PWA and dispatch A_5, A_3 and A_2 in sequence. Then, it becomes a WA. The virtual dispatch/transfer implies the changes of different layer positions and corresponding states of a PWA only, without any time cost. But they will cause the changes of logical relationships between different agents. To describe the route more clearly, we may have to label the layer position for an agent and use A_i^{Lx} denoting the agent A_i at layer L_x. For instance, A_1^{L1} at layer L_1 is the *parent agent* of agent A_5^{L2} at layer L_2. After transferring to layer L_2, A_1^{L2} becomes the *left sibling agent* of A_5^{L2}.

Clearly, when there are *n* mobile agents the dispatch complexity is $O(log_2 n)$. In contrast, a serial migration model has a complexity of $O(n)$.

3. SECURE ROUTE STRUCTURES AND THEIR SECURITY PROPERTIES

Following the binary dispatch model, if no secure route structure is provided, the route information of an agent will be revealed to the host it visits. Attacks can be easily mounted without being detected. These attacks include

ATK1: route forging attack (forge a route)

ATK2: replay attack (dispatch a forged agent to a visited host)

ATK3: wrong dispatch attack (dispatch an agent to a wrong host)

ATK4: dispatch skip attack (skip a predefined dispatch)

ATK5: sub-route deleting attack (delete a unused sub-route), or

ATK6: dispatch disorder attack (break the predefined dispatch order)

The structure of an agent can be briefly described as follows [11]:

$$\{Cer_{H0}/id_{H0}, S, C, D\}$$

where Cer_{H0} is the certificate of its sender, say H_0, which should be a registered host in PKI (Public Key Infrastructure) environment. With it, a receiver could verify the ownership of an arriving agent. Without loss of generality, for simplicity, Cer_{H0} can be replaced by the unique *id*, say id_{H0}, of the sender since we assume the certificates can be obtained in advance. *S* is the state of an agent represented by a set of arguments. A route is part of it. *C* is the code of the agent and *D* is the results obtained after execution. *D* can be sent back to the query node through messages.

In the following context, we suppose agent A_i should be dispatched to host H_i where upon arriving, A_i should deploy its subsequent child agents if it is a PWA or access local data if it is a WA.

During the process of dispatching, a PWA resides at the same host, but its layer positions vary in a dispatch tree. As we mentioned in Section 2, the logical relations between agents are dynamically changed due to the changes of their layer positions. To describe the route more clearly, we may have to label the layer position for an agent and use A_i^{Lx} to denote the agent A_i at layer L_x. As shown in Figure 2, $A_{i\text{-}P}$ denotes the parent agent of A_i. If A_i resides at layer L_x, $A_{i\text{-}P}$ will reside at a higher layer $L_{x\text{-}1}$, denoted as $A_{i\text{-}P}^{Lx\text{-}1}$. But we can simply denote it as $A_{i\text{-}P}$ if it is known that A_i is residing at layer L_x.

Likewise, the left sibling agent of A_i^{Lx} residing at layer L_x is denoted as $A_{i\text{-}LS}$ while the right child agent of A_i^{Lx} is denoted as $A_{i\text{-}RC}$. So $r(A_i^{Lx})$ and $r(A_i^{Lx+1})$ are different because of the same agent with different layer positions. But for hosts, we do not need to label their layer position since their address and key do not change with the change of layer position. All terms and symbols used in our models are listed and explained in Table 1.

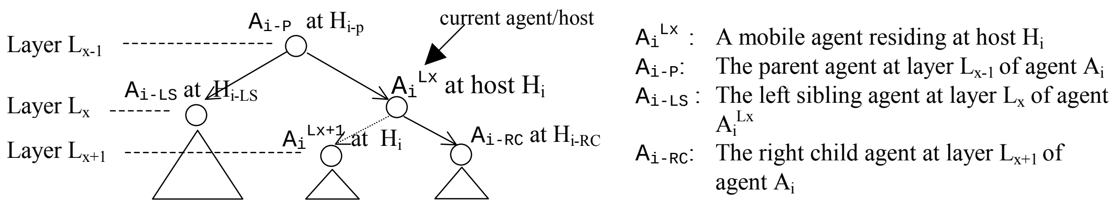

Figure 2 An Example of Symbols in a Dispatch Tree

Table 1 Terms and Symbols Used in Our Models

A_0	The *master agent* at home host H_0
A_i	A mobile agent residing at host H_i
$A_{i\text{-}LS}$	The left sibling agent of agent A_i^{Lx}
A_i^{Lx}	A mobile agent A_i at layer L_x
$A_{i\text{-}P}$	The parent agent of agent A_i
$A_{i\text{-}RC}$	The right child agent of agent A_i
$A_{i\text{-}RC'}$	The substitute right child agent
$Entity_{Ai}$	The whole entity of agent A_i received by host H_i including its code and data
Ħ(…)	the digest with fix-length returned by hash function Ħ;
H_0	Home host where master agent A_0 resides
H_i	A host where agent A_i should go
$H_{i\text{-}LS}$	The left sibling host of host H_i
$H_{i\text{-}P}$	The parent host of host H_i
$H_{i\text{-}RC}$	The right child host of host H_i
$H_{i\text{-}RC'}$	The substitute right child host where the substitute right child agent $A_{i\text{-}RC'}$ should go
$ip(H_i)$	The network address of host H_i
isPWA	A token denoting the current agent is a PWA
isWA	A token denoting the current agent is a WA
L_x	Layer position L_x
P_{Hi}	The public key of host H_i
$r(A_i^{Lx})$	The encrypted route for agent A_i^{Lx}
$r'(A_{i\text{-}RC'})$	The route for the substitute right child agent $A_{i\text{-}RC'}$; it is encrypted by the public key of APWA
S_{H0}	The secret (private) key of host H_0
S_{H0}(Ħ(…))	The signature generated at H_0
t	The time when a route is generated at H_0
t_{ir}	The time when agent A_i is received by host H_i
$t_{i\text{-}Result}$	The time when A_i gets result at host H_i

In all the proposed schemes presented in this section, the route is generated by A_0 at H_0 before any dispatch is performed. The route is encrypted using public keys of the corresponding hosts that will be visited. The encrypted route can be decrypted with the assistance of the current host. The agent can verify the validity of plaintext using the included signature.

In this paper, $P_B[m]$ denotes encrypting a message m using the public key of participant B while S_B denotes B's secret key. One-way hash function Ħ operated on message m is denoted as Ħ(m). S_B[Ħ(m)] denotes the signature generated by B.

3.1 Route Structure (I): Atomic Route and Atomic Signature

Suppose agent A_i is dispatched to current host H_i, its dispatch layers from H_i are $L_1, L_2, \ldots, L_m$ $(m \geq 1)$. The route of A_i is:

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

(ii) if $1\le x<m$, where A_i is a PWA,

$r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),t,S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),id_{H0},t))]$

(iii) if x=m, A_i should be a WA, $r(A_i^{Lm})=P_{Hi}[isWA,ip(H_0),t,S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_0),id_{H0},t))]$

where $H_{i\text{-}P}$ is the parent host of H_i; $H_{i\text{-}RC}$ is the right child host of H_i where the right child agent $A_{i\text{-}RC}$ should be dispatched to; H_0 denotes the home host where A_0 resides. isPWA is the token meaning the current agent is a PWA while isWA means the agent is a WA. t is the unique timestamp for this pool of mobile agents when all routes are generated at home host H_0. P_{Hi} is the public key of H_i; S_{H0} is the secret key of H_0. $S_{H0}(\text{Ħ}(\ldots))$ is the signature generated at H_0. It can be used to authenticate the owner, for this pool of mobile agents.

In this structure, the route of a PWA is the concatenation list of routes in different layers. Each route contains the network address of the right child host $H_{i\text{-}RC}$ (say $ip(H_{i\text{-}RC})$), the token (isPWA or isWA) showing the agent is a PWA or a WA, the route for the right child agent (say $r(A_{i\text{-}RC})$) and corresponding signature. Also the addresses of the parent host $H_{i\text{-}P}$, current host H_i and right child host $H_{i\text{-}RC}$ are included in the signature so that wrong dispatch attack (*ATK3*) can be detected by the destination host. With the signature, a forged route (*ATK1*) can be easily detected by the current agent. With t, any replay attack (*ATK2*) that should use a signature copy will not be successful since the destination host can verify the signature.

However, in structure (I), a sub-route can be deleted (*ATK5*) by the parent or current host since there is no dependence between routes. A dispatch skip attack (*ATK4*) can be performed by the current host. Dispatch disorder attack (*ATK6*) may be successful since routes at different layers are all encrypted by the public key of H_i, say P_{Hi}. The dispatch order can be arbitrarily controlled by H_i since it performs decryption and transfers the plaintext to the agent. It can pass routes to the agent in a reverse order so that the dispatch sequence is totally broken. As long as all dispatches are performed, it is not easy to detect this kind of attack but the whole dispatch performance is worsened. For example, H_1 can make A_1 dispatch A_2, A_3 and A_5 in sequence after having performed its local data-accessing task. In such a case, if each host makes the current agent access the local data before performing any dispatches, the total performance will be almost the same as a serial model.

3.2 Route Structure (II): Nested Route and Atomic Signature

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

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

(ii) if $1\le x<m$, A_i is a PWA, $r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),r(A_i^{Lx+1}),t,S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_{i\text{-}RC}),id_{H0},t))]$

(iii) if x=m, A_i is a WA, $r(A_i^{Lm})=P_{Hi}[isWA,ip(H_0),t,S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_0),id_{H0},t))]$

Adopting the combination of encryption and signature, structure (II) shares the security properties as structure (I) against attacks *ATK1* to *ATK3*. In structure (II), the route is in a nested structure. But the signature is an atomic structure. Some attacks cannot be detected earlier.

The subsequent route $r(A_{i\text{-}RC})$ can be obtained only after $r(A_i^{Lx})$ is decrypted. But $r(A_{i\text{-}RC})$ can be deleted without being detected *after* route $r(A_i^{Lx})$ is just decrypted since route $r(A_{i\text{-}RC})$ does not appear in the signature of $r(A_i^{Lx})$(*ATK5*). This will result in a successful dispatch skip attack (*ATK4*) that can only be detected by master agent A_0. Meanwhile, dispatch disorder attack (*ATK6*) can be successful if H_i decrypts all routes and gives A_i route information in an arbitrary sequence.

3.3 Route Structure (III): Atomic Route and Nested Signature

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

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

(ii) if $1 \le x < m$, A_i is a PWA, $r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),t, S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),r(A_i^{Lx+1}),\ldots,r(A_i^{Lm}), id_{H0},t))]$

(iii) if $x=m$, A_i is a WA,

$r(A_i^{Lm})=P_{Hi}[isWA,ip(H_0),t, S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_0),id_{H0},t))]$

Like structure (II), structure (III) can protect against attacks *ATK1* to *ATK4*. Since $r(A_{i\text{-}RC})$, $r(A_i^{Lx+1}),\ldots,r(A_i^{Lm})$ appear in the signature of $r(A_i^{Lx})$, deletion of these subsequent routes before use can be detected by agent A_i when verifying the signature. However, routes are in concatenation list. If $r(A_i^{Lx})$ is deleted *before* it is used or decrypted, the attack cannot be detected by the current agent. The reason is that an earlier route does not appear in the signature of subsequent routes (*ATK5*). Meanwhile, the dispatch disorder attack (*ATK6*) cannot be detected as long as all child agents are dispatched.

3.4 Route Structure (IV): Nested Route and Nested Signature

In structure (IV), both nested route and nested signature are used. Suppose agent A_i is dispatched to current host H_i, its dispatch layers from H_i are $L_1, L_2, \ldots, L_m$. The route of A_i is:

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

(ii) if $1 \le i < m$, A_i is a PWA, $r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),r(A_i^{Lx+1}),t, S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),r(A_i^{Lx+1}),id_{H0},t))]$

(iii) if $i=m$, A_i is a WA, $r^{Lm}(A_i)=P_{Hi}[isWA,ip(H_0),t,S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}), ip(H_i),ip(H_0),id_{H0},t))]$

Since both the route and the signature are in a nested structure, deleting subsequent routes before encryption will not be possible (*ATK5*). So the security property is improved in comparison with above-mentioned three structures. However, since all routes are encrypted by P_{Hi}, the dispatch disorder attack (*ATK6*) remains unsolved. Similarly, if host H_i passes to the current agent A_i only subsequent routes and deletes earlier routes *after* decrypting all routes (*ATK4*), A_i cannot find the attack since earlier routes cannot be included in subsequent routes. This attack can only be confirmed after the successful investigation conducted by A_0. This is the same as the above three structures.

3.5 Route Structure (V): A Secure Route Structure Ensuring Dispatch Order

Based on structure (IV), structure (V) can provide the capability based on confirmation feedback mechanism to restrict the dispatch order to be strictly followed while ensuring other security properties. The route for next right dispatch is included in the route for the current right child agent. Only after the right dispatch is successful, can current agent receive the route for the next dispatch from the confirmation feedback sent by the dispatched right child agent.

To make the protocol clearer, in this section we introduce the basic idea via an example. The formal description on route structure and dispatch protocol are presented in Section 4.

In Figure 3, master agent A_0 dispatches 2 PWAs respectively, namely A_1^{L1} and A_9^{L1}, encapsulating routes $r(A_1^{L1})$ and $r(A_9^{L1})$ to them. To simplify, let us look at the branch rooted by A_1^{L1} only.

The route of A_1^{L1} is:

$r(A_1^{L1})=P_{H1}[isPWA,ip(H_5),r(A_5^{L2}),t,S_{H0}(\text{Ħ}(isPWA,\ldots,t))]$

Where $r(A_5^{L2})=P_{H5}[isPWA,ip(H_7),r(A_1^{L2}),r(A_7^{L3}),t,S_{H0}(\text{Ħ}(isPWA,\ldots,t))]$

And $r(A_1^{L2})=P_{H1}[isPWA,ip(H_3),r(A_3^{L3}),t,S_{H0}(\text{Ħ}(isWA,\ldots,t))]$

Note $r(A_1^{L1})$ has only one sub-route, say $r(A_5^{L2})$, that is prepared for the right child agent A_5^{L2}. $r(A_5^{L2})$ has two sub-routes. One is $r(A_7^{L3})$, the route for its right child agent A_7^{L3}. The other is $r(A_1^{L2})$, that should be returned to agent A_1. Thus, the main difference between route structure (V) and (IV) is that A_5^{L2} and A_1^{L2} are sibling agents but $r(A_1^{L2})$ is included in $r(A_5^{L2})$ in structure (V). Only after A_5^{L2} is successfully dispatched to H_5, can A_5 send $r(A_1^{L2})$ back to A_1 so that A_1 can know it should transfer to layer L_2 and dispatch A_3 to H_3. Note that $r(A_1^{L2})$ is encrypted by P_{H1} so A_5 and H_5 cannot decrypt it and H_1 cannot get it before dispatching A_5 to H_5. Hence, the dispatch order is ensured (*ATK6*). As all sub-routes appear in the signature, route deletion attack (*ATK5*) and dispatch skip attack (*ATK4*) can be detected by the current agent.

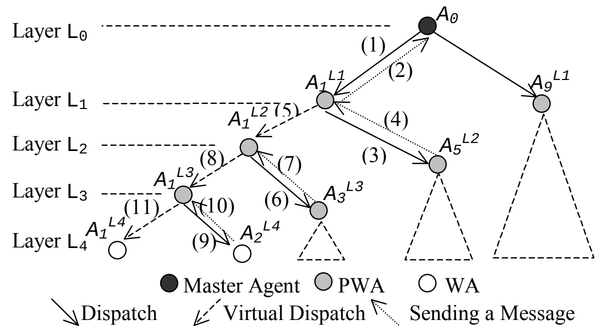

Figure 3 Dispatch Process by Agent A_1

As shown in Figure 3, the dispatch processes of A_1 are as follows:

Step 1: Master agent A_0 dispatches agent A_1 to H_1.

Step 2: When A_1 (i.e. A_1^{L1}) arrives host H_1, H_1 decrypts the route $r(A_1^{L1})$ and A_1 gets an decrypted route as

$R=S_{H1}[r(A_1^{L1})]=(isPWA,ip(H_5),r(A_5^{L2}),t,S_{H0}(\not{H}(isPWA,\dots,id_{H0},t))$. A_1 then sends a reply to A_0 confirming the successful dispatch:

$msg\_A_1^{L1}\_to\_A_0=P_{H1}[t_{1r},S_{H1}(\not{H}(ip(H_0),ip(H_1),Entity_{A1},t_{1r}))]$

where t_{1r} is the time when A_1^{L1} is received.

Step 3: From R, A_1^{L1} knows it is currently a PWA so it dispatches its right child agent, namely A_5^{L2}, to host H_5 encapsulating the route $r(A_5^{L2})$ to it.

$r(A_5^{L2})=P_{H5}[isPWA,ip(H_6),r(A_1^{L2}),r(A_6^{L3}),t,S_{H0}(\not{H}(\dots)]$

Step 4: Once having arrived host H_5 and having decrypted its route $r(A_5^{L2})$, A_5^{L2} will send a reply to A_1^{L1} including $r(A_1^{L2})$ to confirm the successful dispatch:

$msg\_A_5^{L2}\_to\_A_1^{L1}= P_{H1}[r(A_1^{L2}),t_{5r},S_{H5}(\not{H}(\dots,t_{5r}))]$

After that, A_5^{L2} deploys the branch including agents A_5, A_6, A_7 and A_8.

Step 5: From the above message, agent A_1^{L1} obtains $r(A_1^{L2})$, and transfers to layer L_2 hereafter becoming A_1^{L2}, where $r(A_1^{L2})=P_{H1}[isPWA,ip(H_3),r(A_3^{L3}),t,S_{H0}(\not{H}(\dots))]$

Step 6: From $r(A_1^{L2})$, A_1^{L2} knows it is still a PWA and should dispatch its new right child agent A_3 (i.e. A_3^{L3}) to host H_3. The route $r(A_3^{L3})$ can be found from route $r(A_1^{L2})$: $r(A_3^{L3})=P_{H3}[isPWA,ip(H_4),r(A_1^{L3}),r(A_4^{L4}),t,S_{H0}(isPWA,\dots,id_{H0},t)]$

Step 7: Similarly, the successful dispatch of agent A_3^{L3} returns A_1^{L2} a message $msg\_A_3^{L3}\_to\_A_1^{L2}$ including $r(A_1^{L3})$: $msg\_A_3^{L3}\_to\_A_1^{L2}=P_{H1}[r(A_1^{L3}),t_{3r}, S_{H3}(\not{H}(\dots,t_{3r}))]$

Step 8: From this message, A_1^{L2} obtains $r(A_1^{L3})$ and knows it should transfer to layer L_3 becoming A_1^{L3}, where $r(A_1^{L3})=P_{H1}[isPWA,ip(H_2),r(A_2^{L4}),S_{H0}(\dots)]$

Step 9: After decrypting $r(A_1^{L3})$, A_1^{L3} dispatches agent A_2^{L4} to host H_4 encapsulating route $r(A_2^{L4})$ to it, $r(A_2^{L4})=P_{H2}[isWA,ip(H_0),r(A_1^{L4}),t,S_{H0}(\ldots)]$

Step 10: After the successful dispatch, A_1^{L2} will get a message $msg\_A_2^{L4}\_to\_A_1^{L3}$ from A_2^{L4}.

$msg\_A_2^{L4}\_to\_A_1^{L3}=P_{H1}[r(A_1^{L4}),t_{2r}, S_{H2}(\overline{H}(\ldots,t_{2r}))]$

Step 11: From the above message, A_1^{L3} obtains route $r(A_1^{L4})$ and knows it is a WA after transferring to layer L_4 and hereafter it should complete its local data-accessing task at host H_1. The result should be sent to master agent A_0.

$r(A_1^{L4})=P_{H1}[isWA,ip(H_0),t,S_{H0}(\ldots)]$

Though the feedback may make the average dispatch time longer, the nature of parallel dispatch is not changed. In fact, in structure (IV), the feedback is essential too in an encrypted structure [11] but it is used for confirming successful dispatch only.

3.6 Discussion

To summarize, the above-proposed secure route structures ensure some basic security properties. They aim to expose only minimal addresses to a host to perform dispatches. Structures (I) to (III) have relatively simple structures, but their security properties are not satisfactory. For structure (IV), the dispatch skip attack (*ATK5*) can be detected by A_0. However, the dispatch disorder attack (*ATK6*) cannot be prevented though it can be considered as a benign attack. For structure (V), its security performance is the best. Table 2 summarizes the security properties of the different route structures.

4. ROUTE STRUCTURE (V) AND ITS DISPATCH PROTOCOL

In this section, we present the formal description of route structure (V) and the corresponding secure dispatch protocol.

4.1 Secure Route Structure

Table 2 Security Properties of Different Route Structures

	Route Forging Attack (ATK1)	Replay Attack (ATK2)	Dispatch to Wrong Host H_w (ATK3)	Dispatch Skip Attack (ATK4)	Sub-Route Deletion Attack (ATK5)	Dispatch Disorder Attack (ATK6)
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	An included route can be deleted after decryption.	No.
III	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 before use can be found by current agent by checking the signature of its front route.	No.
IV	Yes, by current agent.	Yes, by destination host.	Yes, by H_w	Yes, by A_0	Deleting subsequent routes before encryption is not possible because of the nested structure. Deleting an included route can be detected 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.

For a PWA, there are 2 kinds of route structures. For example, in Figure 3, the routes for A_1^{L2} and A_5^{L2} have different structures since A_5^{L2} is a newly dispatched agent but A_1^{L2} is formerly A_1^{L1}. The route of A_5^{L2} should include a sub-route (i.e. $r(A_1^{L2})$) that should be returned to its parent agent A_1^{L1}. For A_1^{L2}, though its parent agent is A_1 too (i.e. A_1^{L1}), its route does not have such a sub-route that should be returned to parent agent. Meanwhile, for a WA, there are 2 kinds of route structures too. For example, A_1^{L4} and A_2^{L4} are 2 WAs (see Figure 3). But A_2^{L4} is a newly dispatched agent. As presented in Section

3.1.5, once A_2^{L4} is successfully dispatched to H_2, it should send A_1^{L3} a confirmation feedback including a route indicating A_1^{L3} to become a WA, namely A_1^{L4}. So $r(A_2^{L4})$ should has a sub-route.

The formal route structures are as follows.

Secure Route Structure (**V**):

(1) For a PWA A_i^{Lx} residing at layer L_x,

a) if A_i^{Lx} is virtually dispatched (e.g. A_1^{L2} at layer L_2 in Figure 3) or it is dispatched by master agent A_0 (e.g. A_1^{L1} or A_9^{L1} in Figure 1), the route of A_i^{Lx} has one sub-route for its right child agent:

$$r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),t,\ S_{H0}(\text{Ħ}(isPWA,ip(H_i),ip(H_{i\text{-}RC}),r(A_{i\text{-}RC}),id_{H0},t))] \qquad \textbf{(V-i)}$$

b) otherwise, A_i^{Lx} should be newly dispatched by its parent agent $A_{i\text{-}P}$ (e.g. A_5^{L2} in Figure 3). Hereby, besides the route for right child agent, namely $r(A_{i\text{-}RC})$, $r(A_i^{Lx})$ includes an extra sub-route for its left sibling agent-namely $r(A_{i\text{-}LS})$. Then the route for A_i^{Lx} is

$$r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),r(A_{i\text{-}LS}),r(A_{i\text{-}RC}),t,S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),\ ip(H_i),ip(H_{i\text{-}RC}),r(A_{i\text{-}LS}),r(A_{i\text{-}RC}),id_{H0},t))] \qquad \textbf{(V-ii)}$$

(2) For a WA A_i^{Lx}, residing at layer L_x,

a) if it is a newly dispatched agent (e.g. A_4^{L4} in Figure 3), the route has one sub-route for the left sibling agent:

$$r(A_i^{Lx})=P_{Hi}[isWA,ip(H_0),r(A_{i\text{-}LS}),t,S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_0),\ r(A_{i\text{-}LS}),id_{H0},t))] \qquad \textbf{(V-iii)}$$

b) otherwise, A_i^{Lx} (e.g. A_1^{L4} in Figure 3) is virtually dispatched by its parent agent which is itself at a higher layer $L_{x\text{-}1}$. Its route has no sub-route:

$$r(A_i^{Lx})=P_{Hi}[isWA,ip(H_0),t,S_{H0}(\text{Ħ}(isWA,ip(H_i),ip(H_0),id_{H0},t))] \qquad \textbf{(V-iv)}$$

4.2 The Feedback based Dispatch Protocol with Secure Routes

Algorithm 1: The feedback based binary dispatch protocol with secure routes (structure (V))

1: When an agent A_i^{Lx} is successfully dispatched to a host H_i, H_i will use its secret key S_{Hi} to decrypt the carried encrypted route $r(A_i^{Lx})$, getting the plain route R as:

$$R=S_{Hi}[r(A_i^{Lx})] \qquad \textbf{(1)}$$

2: **WHILE** A_i^{Lx} is a PWA **DO**

3: **IF** R has 2 sub-routes **THEN** //A_i^{Lx} is virtually dispatched

4: A_i^{Lx} first sends a message to its parent agent $A_{i\text{-}P}$ as follows:

$$msg\_A_i^{Lx}\_to\_A_{i\text{-}P}=P_{Hi\text{-}P}[r(A_{i\text{-}LS}),t_{ir},\ S_{Hi}(\text{Ħ}(ip(H_{i\text{-}P}),ip(H_i),Entity_{Ai},r(A_{i\text{-}LS}),t_{ir})] \qquad \textbf{(2)}$$

where $r(A_{i\text{-}LS})$ is originally included in route $r(A_i^{Lx})$; $Entity_{Ai}$ is the whole entity of agent A_i^{Lx} received by host H_i including its code and data; t_{ir} is the time when H_i received A_i^{Lx}.

(Note: Host $H_{i\text{-}P}$ will store this message in its dispatch record database that is useful for future investigation. With $r(A_{i\text{-}LS})$, the parent agent of A_i^{Lx} can transfer to layer L_x (virtual dispatch) and starts to dispatch its new right child agent after the route is decrypted by $H_{i\text{-}P}$.)

5: **ELSE** //A_i^{Lx} is newly dispatched and R has only 1 sub-route

6: With the assistance of H_i, A_i^{Lx} sends a reply to parent agent A_0 showing that the agent has been successfully dispatchd.

$$msg\_A_i^{Lx}\_to\_A_0= S_{Hi}(\text{Ħ}(ip(H_{i\text{-}P}),ip(H_i),Entity_{Ai},t_{ir})) \qquad \textbf{(3)}$$

7: **End IF**

8: A_i^{Lx} dispatches its right child agent $A_{i\text{-}RC}$ to host $H_{i\text{-}RC}$ at layer L_{x+1} encapsulating the route $r(A_{i\text{-}RC})$ to it ($r(A_{i\text{-}RC})$ can be obtained from the decrypted route R).

9: Then A_i^{Lx} will get a reply from the right child agent $A_{i\text{-}RC}$ which includes an encrypted route, say R_{new}, for its next action;

10: A_i^{Lx} virtually dispatches/transfers itself to layer L_{x+1} still residing at host H_i. Let $x=x+1$ and $r(A_i^{Lx})=R_{new}$

10: A_i^{Lx} get the decrypted route $R=S_{Hi}[r(A_i^{Lx})]$;

11: **END WHILE** //(now A_i^{Lx} is a WA)

12: **IF** R has 1 sub-route **THEN** // A_i^{Lx} is a newly dispatched WA

13: A_i^{Lx} sends a message to its parent agent $A_{i\text{-}P}$ as follows:

$$msg\_A_i^{Lx}\_to\_A_{i\text{-}P}=P_{Hi\text{-}P}[r(A_{i\text{-}LS}),t_{ir}, S_{Hi}(\not{H}(ip(H_{i\text{-}P}),ip(H_i),Entity_{Ai},r(A_{i\text{-}LS}),t_{ir})] \quad \mathbf{(4)}$$

where $r(A_{i\text{-}LS})$ is originally included in the encrypted route $r(A_i^{Lx})$.

14: **END IF**

15: A_i^{Lx} starts its task for local data accessing at host H_i;

16: When the data access task is completed, A_i^{Lx} will dispose after having successfully sent a message to agent A_0,

$$msg\_A_i^{Lx}\_to\_A_0=P_{H0}[ip(H_{i\text{-}P}),ip(H_i),Result_{Hi}, S_{H0}(\not{H}(isWA,\ldots,t)),t_{i\text{-}Result},S_{Hi}(\not{H}(ip(H_{i\text{-}P}),ip(H_i),Result_{Hi},t_{i\text{-}Result}, S_{H0}(\not{H}(isWA,\ldots,t)),t_{i\text{-}Result}))] \quad \mathbf{(5)}$$

where $S_{H0}(\not{H}(isWA,\ldots,t))$ is the signature by H_0 which is included in the route of A_i^{Lx}. Here it is used to show the identification of the agent A_i^{Lx}. $S_{Hi}(\not{H}(\ldots,S_{H0}(\not{H}(isWA,\ldots,t)), t_{Result}))$ is the signature generated by current host H_i. $Result_{Hi}$ is the result obtained at H_i. $t_{i\text{-}Result}$ is the time when getting the result.

17: **STOP**.

5 ROBUSTNESS ENHANCED EXTENTION

So far we have presented a security enhanced dispatch protocol for mobile agents based on route structure (V). However, at a certain layer, each PWA only knows the right child host $H_{i\text{-}RC}$ where its right child agent is to be dispatched. As such, should the right child host be unreachable, the right dispatch branch cannot be deployed and all the grouped members will thereby not be activated.

A straightforward solution is for a PWA to have a substitute route for dispatching its right child agent. In this way, if the predefined right child agent cannot be successfully dispatched due to some reasons from the destination host, the PWA can have another route for the right dispatch. In this section, we extend secure route structure (V) to handle unreachable hosts.

In the robust route structure, besides the route for the right child agent, a substitute route r' is included in the route of a PWA. But it is encrypted by the public key of the first PWA of another branch. For instance, in Figure 1, any substitute routes for the branch rooted by A_1 are encrypted by the public key of A_9, which is the root of the other branch. Likewise, any substitute routes in the branch rooted by A_9 are encrypted by the public key of A_1. Here A_1 and A_9 are called *Assistant PWA* (APWA). Once a destination host is unreachable, the current host can send the substitute route to its APWA. After decryption, the current host will know the substitute host (SH), where it can send an agent to continue to activate the right branch. The route for a WA is the same as structure (V).

Without the loss of generality, we suppose the addresses of APWAs are public.

If we were to provide one substitute route, secure route structure (V) presented in Section 4 can be extended as follows:

Robustness Enhanced Secure Route Structure (**VI**):

(1) For a PWA A_i^{Lx} residing at layer L_x,

a) if A_i^{Lx} is virtually dispatched by its parent agent or it is dispatched by master agent A_0, the route for A_i^{Lx} is $r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),ip(H_{i\text{-}RC'}),r(A_{i\text{-}RC}),r'(A_{i\text{-}RC'}),t,S_{H0}(\text{Ħ}(isPWA,ip(H_i),ip(H_{i\text{-}RC}),ip(H_{i\text{-}RC'}),r(A_{i\text{-}RC}),r'(A_{i\text{-}RC'}),id_{H0},t))]$ **(VI-i)**

where

- $H_{i\text{-}RC'}$ is the substitute right child host where the substitute right child agent $A_{i\text{-}RC'}$ should go;
- $r'(A_{i\text{-}RC'})$ is the substitute route for agent $A_{i\text{-}RC'}$ encrypted by the public key of the APWA;
- $ip(H_{i\text{-}RC'})$ and $r'(A_{i\text{-}RC'})$ appears in the signature of H_0 after being hashed;
- the rest is similar to route structure (V-i).

b) otherwise, A_i^{Lx} should be dispatched by its parent agent $A_{i\text{-}P}$ at layer L_{x-1}. Then the route for A_i^{Lx} is

$$r(A_i^{Lx})=P_{Hi}[isPWA,ip(H_{i\text{-}RC}),ip(H_{i\text{-}RC'}),r(A_{i\text{-}LS}),r(A_{i\text{-}RC}),r'(A_{i\text{-}RC'}),t, S_{H0}(\text{Ħ}(isPWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_{i\text{-}RC}),ip(H_{i\text{-}RC'}),r(A_{i\text{-}LS}),r(A_{i\text{-}RC}), r'(A_{i\text{-}RC'}),id_{H0},t))] \quad \textbf{(VI-ii)}$$

where

- $H_{i\text{-}RC'}$ is the substitute right child host where the substitute right child agent $A_{i\text{-}RC'}$ should go;
- $r'(A_{i\text{-}RC'})$ is the substitute route for agent $A_{i\text{-}RC'}$ encrypted by the public key of the APWA;
- $ip(H_{i\text{-}RC'})$ and $r'(A_{i\text{-}RC'})$ appears in the signature of H_0 after being hashed;
- the rest is similar to route structure (V-i).

(2) For a WA A_i^{Lx}, residing at layer L_x,

a) if it is a newly dispatched agent, the route is $r(A_i^{Lx})=P_{Hi}[isWA,ip(H_0),r(A_{i\text{-}LS}), t,S_{H0}(\text{Ħ}(isWA,ip(H_{i\text{-}P}),ip(H_i),ip(H_0),r(A_{i\text{-}LS}),id_{H0},t))]$ **(VI-iii)**

b) otherwise, A_i^{Lx} is virtually dispatched by its parent agent which is itself at a higher layer L_{x-1}. Its route is $r(A_i^{Lx})=P_{Hi}[isWA,ip(H_0),S_{H0}(\text{Ħ}(isWA,ip(H_i),ip(H_0),id_{H0},t))]$ **(VI-iv)**

Now suppose A_1 is the first PWA in the left dispatch sub-tree; A_9 is the right first PWA as shown in Figure 4(a). $r'(A_{1\text{-}RC'})$ is encrypted by the public key of H_9, say P_{H9}.

If the dispatch failure occurs when A_1 is dispatching its right child agent $A_{1\text{-}RC}$ to the right child host $H_{1\text{-}RC}$, A_1 should report it to A_9 attaching the substitute route $r'(A_{1\text{-}RC'})$.

In this way by robust route structure, a PWA will have a substitute route for the dispatch of its right child agent. Once the original dispatch is not successful, with the assistance of its APWA, it can have another destination to perform the right dispatch.

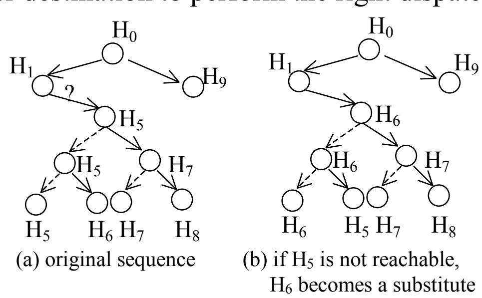

Figure 4 Examples of Substitute Routes

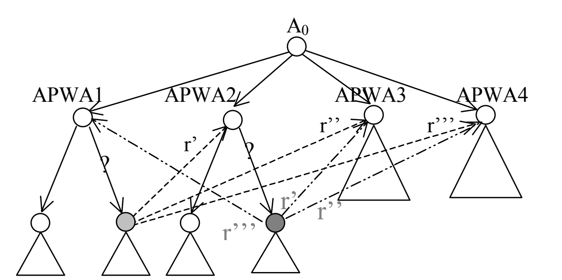

Figure 5 A Model with 3 Substitute Routes and 4 Branches

What we should address is that the substitute host is originally included in the members for the right dispatch branch. Taking the dispatch tree in Figure 4 as an example, if the dispatch failure occurs when A_1 at host H_1 is dispatching A_5 to H_5, A_1 can get a substitute route with the assistance PWA A_9 at H_9. To generate the substitute route, choosing H_6 to be the substitute host is better. By exchanging the positions of H_5 and H_6 as shown in Figure 4(b), H_6, which is originally a pure leaf node (i.e. the corresponding agent is a Worker Agent only), becomes the root of the branch with H_5 to H_8. Most sub-branches under H_6 are kept unchanged. Thus, we can reduce the complexity to generate a new substitute route.

Following the same idea, the second substitute route can be generated. H_8 instead of H_7 can be the second substitute. An originally unreachable host should be put to be a leaf node so that the failure of the second dispatch attempt can be made without increasing more load of the APWA for route decryption.

Figure 5 presents an extension providing 3 substitute routes and all agents are equally distributed into 4 APWAs, termed as APWA1, APWA2, APWA3 and APWA4. In each branch following an APWA, the dispatch is performed in binary way. Each substitute route is encrypted by different public keys of hosts where different APWAs reside. For instance, when a dispatch failure occurs in the first branch, the first substitute route is sent to APWA2 for decryption. If the substitute host is not yet reachable, the second substitute route will be sent to APWA3. Likewise the 3rd substitute route can only be decrypted by APWA4. Similarly, the first substitute route in the branch led by APWA2 should be sent to APWA3 for decryption and so on. In this way, the workloads of decryption for APWAs are partitioned and the whole dispatch efficiency is not significantly decreased while the robustness is enhanced.

6. COMPLEXITY ANALYSIS

In this section, we analyze the complexity of the proposed structure (V). As comparison, we shall use two existing serial models as reference.

6.1 Two Related Serial Models

The model by Westhoff et al in [10] adopted a fully serial migration providing a secure route structure without any robustness mechanism. An agent visits a set of hosts one by one. Suppose the visited hosts are $H_1, H_2, \ldots, H_n$, the route is:

$$\begin{cases} r(H_i)=P_{Hi}[ip(H_{i+1}),r(H_{i+1}),t,S_{H0}(\not{H}(ip(H_i),ip(H_{i+1}),r(H_{i+1}),t))] \quad (1 \le i < n) \\ r(H_n)=P_{Hn}[EoR,t,S_{H0}(\not{H}(ip(H_{n-1}),ip(H_n),t)] \end{cases} \quad \textbf{(Serial I)}$$

where S_{H0} is the secret key of home host H_0 and EoR is the token meaning the end of the route.

Obviously the migration complexity is *O(n)* if there are *n* hosts to be visited one by one.

In [13] Li et al proposed a robust route structure for serial migrating agents and the route robustness is enhanced by equally dividing a route into two parts. They are distributed to two agents A_1 and A_2 respectively. A_1 and A_2 are in partner relationship. Each agent residing at any host en route knows the addresses of the next destination and an alternative host. But the latter is encrypted by the public key of its partner agent. In case the migration cannot be performed, the encrypted address will be sent to the partner agent for decrypting. With its assistance, the agent can continue its migration.

In Li's model, as the addresses of *n* hosts are equally distributed to two agents, say $\{ip(H_1),\ldots,ip(H_m)\}$ and $\{ip(H_{m+1}),\ldots,ip(H_n)\}$. The nested route structure is:

$$r(H_i)=P_{Hi}[ip(H_{i+1}),r(H_{i+1}),r(H_i)',t,S_{H0}(\not{H}(ip(H_{i+1}),r(H_{i+1}),r(H_i)',t))] \quad \textbf{(Serial II)}$$

where $r(H_i)'=P_{PA}[ip(H_{i+2}),r(H_{i+2}),r(H_{i+2})',t,S_{H0}(\not{H}(ip(H_{i+1}),r(H_{i+2}),r(H_{i+2})',t))]$ is the substitute route where H_{i+2} is the new destination if H_{i+1} is not reachable. P_{PA} is the public key of the partner agent.

The problem of this model is that since both A_1 and A_2 are dynamically migrating. So when one needs the other's assistance, locating each other will be costly for both time and system resources. Meanwhile, the model is serial so it is neither efficient nor suitable for large-scale mobile agents. The whole migration time can be theoretically half of Westhoff's model. However the time complexity remains *O(n)*.

6.2 Complexity Analysis

In this section, we'd like to have some rough idea about the complexities of serial models and parallel models. To simplify, we first assume the time for encrypting an arbitrary-length message is a constant. Empirical studies are illustrated in Section 7.

As analyzed in Section 6.1, we can know the migration complexity of two serial models.

Theorem 1: Neglecting the time spent on local data access, the time complexity of migration of Westhoff's model and Li's model for visiting *n* hosts is $O(n)$.

As analyzed in our previous work [3], if n $(n\geq 2)$ WAs are dispatched by binary dispatch model, $h=\log_2 n$ $(h\geq 1)$ is an integer and the height of the dispatch tree, Δt is the time for dispatching a PWA or a WA, then the total dispatch time for *n* WAs is $T=(h+1)\Delta t$. So we have

Theorem 2: If n $(n\geq 2)$ mobile agents are dispatched by binary dispatch model, the dispatch complexity is $O(\log_2 n)$.

For Westhoff's model, the route with *n* addresses can be generated after the route with *n-1* addresses has been generated. So, the complexity $T(n)$ can be calculated from the follows

$$\begin{cases} T(n)=T(n-1)+C \\ T(1)=C,\ C \text{ is a constant and the time for encrypting a route} \end{cases}$$

From $T(n)=T(n-1)+C$, we have $T(n)=nC$. So $T(n)$ is $O(n)$.

Theorem 3: Assuming that the time to encrypt a route is a constant, the time complexity for generating a route with *n* addresses in Westhoff's model is $O(n)$.

In [11], the route generation complexity of structure (IV) is analyzed. It is $O(n)$ where $T(n)=2T(n/2)$ $(n=2^k)$, $T(i)=2T(i/2)+C$ $(i=2^h,\ 2^{k-1}\leq i\leq 2)$ and $T(1)=C$.

For route structure (V), the route of a PWA with *i* addresses consists of the sub-route of its right child agent, which has *i/2* address and includes the route of its left sibling agent with another *i/2* addresses. So, for our model, the complexity for generating routes without substitute route is $O(n)$, where

$$\begin{cases} T(n)=2T(n/2)\quad (n=2^k) & \text{// 2 routes are generated for left branch and right branch, each has } n/2 \text{ addresses} \\ T(i)=2T(i/2)+2C\ (i=2^h,\ 2^{k-1}\leq i\leq 2) & \text{// if } r(A_i^{Lx}) \text{ has } i \text{ addresses, each sub-route of its left sub-branch and right sub-branch has } i/2 \text{ addresses} \\ T(1)=C \end{cases}$$

Theorem 4: In the secure binary dispatch model (V), the complexity for generating routes without substitute route is $O(n)$.

Table 3 summarizes and compares the features of Westhoff's model and binary dispatch model.

Table 3 Comparison of Models without Substitute Routes

Features / Models	Nested Secure Route	Dispatch/ Migration Complexity	Route Generating Complexity
Westhoff's Model	*Yes*	$O(n)$	$O(n)$
Binary Dispatch with Secure Routes (IV)	*Yes*	$O(\log_2 n)$	$O(n)$
Binary Dispatch with Secure Routes (V)	*Yes*	$O(\log_2 n)$	$O(n)$

Theorem 5: Assuming that the time to encrypt a route is a constant, the time complexity for generating a route with 1 substitute route of Li's model is $O(n)$.

In Li's model, suppose the hosts in predefined sequence are $\{H_1,\dots,H_i,H_{i+1},H_{i+2},\dots,H_m\}$, if host H_{i+1} is not reachable, H_{i+2} will become the next destination from H_i and H_{i+1} will never be visited for this journey. Consequently, from the route structure (Serial II), when generating $r(H_i)$, both $r(H_{i+1})$ and $r(H_i)'$ should be generated first. $r(H_i)'=P_{PA}[ip(H_{i+2}),r(H_{i+2}),r(H_{i+2})', S_{H0}(ip(H_{i+1}),r(H_{i+2}),r(H_{i+2})',t)]$, it is a substitute route with the addresses of

$H_{i+2}, H_{i+3}, \ldots, H_n$ in sequence. Note $r(H_{i+1})=P_{Hi+1}[ip(H_{i+2}), r(H_{i+2}), r(H_{i+2})', t, S_{H0}(ip(H_{i+1}), r(H_{i+2}), r(H_{i+2})', t)]$. The difference of $r(H_{i+1})$ and $r(H_i)'$ is that they are encrypted by different public keys. Therefore when generating $r(H_i)'$, $r(H_{i+2})$, $r(H_{i+2})'$ exist already and the cost for generating $r(H_i)'$ is a constant *C* only. Hereby the route generation complexity with 1 substitute route is

$$\begin{cases} T(m)=T(m-1)+2C & \text{// each of the 2 dispatched agents has } m \text{ addresses, } 2m=n \\ T(1)=C \end{cases}$$

From $T(m)=T(m-1)+2C$ we have $T(m)=2mC-C$. So $T(n)$ is $O(n)$.

However, if a failed host is used for a second attempt in Li's model, the complexity for generating routes will become extremely bad since the sequence of hosts in a substitute route has been changed and the route should be generated and encrypted again.

In Li's model, if host H_{i+1} is not reachable from H_i, and H_{i+1} is put as the last destination for the second attempt, the host sequence in the substitute route will be $\{H_{i+2}, H_{i+3}, \ldots, H_n, H_{i+1}\}$. In such a case, when a route includes 1 substitute route, since the substitute route should be re-generated, the time complexity will be $T(n)=2T(n-1)+C$ and $T(n)$ is $O(2^n)$. Likewise, when there are 3 substitute routes, the time complexity will be $T(n)=4T(n-1)+C$ and $T(n)$ is $O(4^n)$. So we have Theorem 6.

Table 4 Comparison of Models with Substitute Routes

Models \ Features	Nested Secure Route	Dispatch/ Migration Complexity	Route Generating Complexity With 1 or 3 Substitute Routes	Try Failed Hosts Latter
Li's Model	*Yes*	$O(n)$	$O(n)$ or $O(2^n)$ / $O(n)$ or $O(4^n)$	*No*
Binary Dispatch (VI) with 1 or 3 Substitute Routes	*Yes*	$O(\log_2 n)$	$O(n\log_2 n)$	*Yes*

Theorem 6: The time complexity for generating routes with 1 substitute route or 3 substitute routes of Li's model making the 2nd attempt to failed hosts are $O(2^n)$ or $O(4^n)$.

In robust binary dispatch model, when generating the first substitute route for a branch, only a few steps should be taken in the left sub-branch of this branch. Considering the case in Figure 4(b), when H_{17} and H_{18} are exchanged, the branches with the root of H_{19}, H_{21} and H_{25} are all not changed. The number of the steps is the height *h* of the sub-branch. And hereby $T(n)$ is $O(n\log_2 n)$, where $T(n)=2T(n/2)+C$, $T(i) \leq 2T(i/2)+2(h+1)C$ ($n=2^k$, $i=2^{h+1}$, $2^{k-1} \leq i \leq 2$) and $T(1)=C$.

Similarly, the step numbers for generating the second substitute route and the third one are all $(2h-1)$. The time complexity for generating a route with 3 substitute routes and 4 branches is $O(n\log_2 n)$, where $T(n)=4T(n/4)+C$, and $T(i) \leq 2T(i/2)+(5h-1)C$ ($n=2^k$, $i=2^{h+1}$, $2 \leq i \leq 2^{k-2}$) and $T(1)=C$. So we have Theorem 7.

Theorem 7: In the secure binary dispatch model, the complexity for generating routes with 1 or 3 substitute routes is $O(n\log_2 n)$.

Table 4 summarizes and compares the features of the 2 models with substitute routes. The proofs of theorems can be found in [17].

7. EXPERIMENTS

In Section 6, for simplicity, the analysis is based on the assumption that the encryption time of a message of any length is a constant. But it does not hold in real application since encrypting a longer message needs longer time. To further study the performance of the different models, we conducted some experiments on a cluster of PCs. These PCs are connected to a LAN with 100Mbytes/s network cards running Window NT, JDK, IBM Aglets 1.0.3 [14]. For route generations, the experiments are based on a PC of Pentium IV 1.8GHz CPU and 512 Mbytes RAM. For serial migration and binary dispatch, the experiments are put on a cluster of PCs of Pentium 200MMX CPU and 64 Mbytes RAM. All programs run on the top of the Tahiti servers from the ASDK [1, 14] and JDK from Sun Microsystems [15].

To encrypt a route, we use the RSA algorithm [16] and the length of each key is 1024 bits. Before generating a signature, hash function MD5 [12] is used to generate a hash value with a fixed-length of 128 bytes. For the third experiment, since all PCs have the same configuration, the performance differences are totally from the difference of serial migration and parallel dispatch. All results are illustrated in Figures 6 to 8. Each result is the average of four independent executions.

7.1 Experiment 1-Route Generation: Westhoff's Model vs. Binary Dispatch Model (Structure I-V)

In this experiment, we compare the route generation time of Westhoff's model and our 5 secure structures. All results are shown in Figure 6. When the number of addresses is fewer than 64, all models deliver similar performances. When the number becomes 64 or more, the binary dispatch model begins to outperform the serial model.

The route generation performances of 5 secure structures are pretty close to each other. The time for structure (V) is longer other 4 structures since they are simpler. With the increase of the number of addresses, the time for Westhoff's model increases very fast. When generating the route with 1024 addresses, the program of the Westhoff's model ran out of memory after the 771st address is added where the heap size is set up to 1200 Mbytes and it has reached the maximum.

Theoretically, when there are *n* addresses, the binary dispatch model should do the encryption for *2n-2* times. For the serial model, it is *n* times only. The time complexities are both *O(n).* If the encryption time for a message is a constant, the route generation time of the binary dispatch model is obviously longer. Nevertheless, the encryption time varies with the length of the encrypted message. For the binary dispatch model, *n* times' encryptions are spent on all leaf nodes in the dispatch tree where the length of each route is only about 200 bytes. Unfortunately for Westhoff's model, each time after encryption, the route's length is increased at least with a length of a network address and a signature. So, the encryption time will gradually increase with the increase of the route length. When the number of addresses is large, the total encryption time will become very long.

For example, when there are 512 addresses, the Westhoff's model performs 512 encryptions. As we measured, it uses 190 seconds (about 9.6% of overall time) to complete the first 256 encryptions and 1793 seconds (about 90.4% of overall time) for the last 256 encryptions. The total time is 1983 seconds. For the binary dispatch model (structure (V)), it completes all encryptions in 114 seconds for 512 nodes taking 39 seconds (about 34% of overall time) for first 256 leaf nodes. The binary dispatch model (structure (V)) obtains 94% saving for 512 addresses.

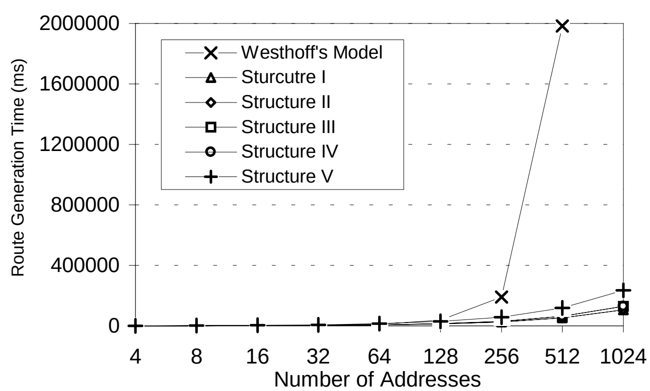

Figure 6 Route Generation Time for Westhoff's Model and Binary Dispatch Model

7.2 Experiment 2-Route Generation: Li's Model vs. Robust Binary Dispatch Model

In this experiment, we compare the route generation time for models with one substitute route. For Li's model, we implemented the case of skipping a failed host.

The results shown in Figure 7 illustrates that though the time complexities of two models analyzed in Section 5 are different (i.e. *O(n)* vs. *O(nlog₂n)*), their performance difference is not very significant.

Li's model can outperform a bit better in most cases. But when the there are 1024 addresses, Li's model becomes inferior. The binary dispatch model obtains 70% saving. When having 2048 addresses, the program of Li's model runs out of memory after running several hours. The reason is the same as we analyzed in experiment 1.

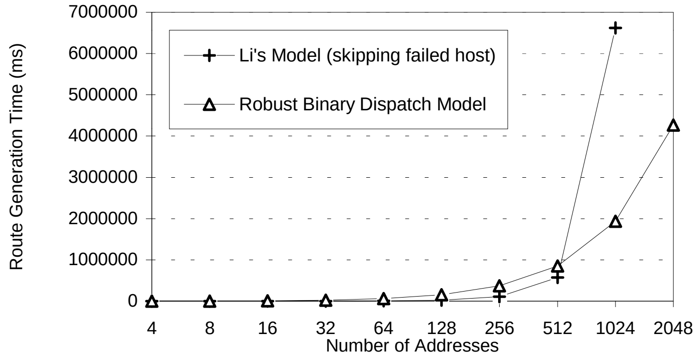

Figure 7 Comparison of the Time for Generating a Route with 1 Substitute Route

7.3 Experiment 3: Serial Migration vs. Binary Dispatch

In this experiment, we tested up to 64 hosts to compare the migration/dispatch time of different models neglecting any robustness mechanism. In the implementation, a mobile agent does not access any local data so that the measured time is used for migration or dispatch only. To obtain each independent result, we rebooted the Tahiti server to prevent the affect from the cache. The results are shown in Figure 8.

When the number of visited hosts is no more than 8, the performance differences are not significant. With the increase of the number of hosts, the migration time of any serial migration model increases very fast. In comparison, the dispatch time for binary dispatch model increases fairly slowly. Meanwhile, the migration time for Li's model is always shorter than that of Westhoff's model since in Li's model, 2 mobile agents are dispatched and each only visits *n/2* hosts. Nevertheless, its performance is not comparable to the binary dispatch model. When having 64 hosts, the binary dispatch model can get 70% and 82% savings respectively in comparison to Li's model and Westhoff's model.

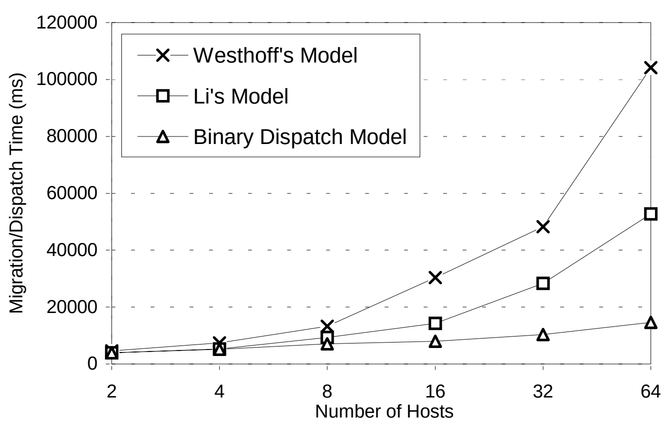

Figure 8 Comparison of The Migration/Dispatch Time

8. CONCLUSIONS

This paper presented several schemes to protect the route structures of mobile agents dispatched in parallel. A feedback based secure dispatch protocol further ensures that the dispatch sequence is strictly followed. We further explored a robustness mechanism to bypass temporarily unreachable hosts so as to deploy the predefined agents through substitute hosts and agents.

Our model can also be applied to the mobile code distribution in a Grid environment and secure message distribution in a distributed environment with minor modifications. For practical applications, mobile agents having the same type tasks and having physically close destinations can be put in the same group encapsulated with pre-encrypted routes. The clone-like strategy can be employed to create an instance of a new agent. For verifying the integrity of a coming agent, the pure code can be included in the signature of a route after being hashed when it is generated at the side of home host. We plan to study this next. Another direction for future work is to improve our model toward extensible route structures to facilitate a more flexible dispatch process.

9. ACKNOWLEDGMENTS

This work is supported by the NSTB/MOE funded project on Strategic Program on Computer Security (R-252-000-015-112/303). We would also like to thank the anonymous reviewers for their insightful and valuable comments.

REFERENCES

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

[2] Papastavrou S., Samaras G. and Pitoura E. (2000) 'Mobile agents for World Wide Web distributed database access', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 12, Issue 5, pp. 802 –820

[3] Wang Y., Tan K.-L. and Ren J. (2002) 'A study of building Internet marketplaces on the basis of mobile agents for parallel processing', World Wide Web Journal, Kluwer Academic Publishers, Vol. 5, Issue 1, pp 41-66

[4] Wilhelm U. G. (1997) 'Cryptographically protected objects', Technical Report, Ecole Polytechnique Federale de Lausanne, Switzerland

[5] Palmer E. (1994) 'An Introduction to Citadel-a Secure Crypto Coprocessor for Workstations', *Proceedings of IFIP SEC'94*, Curacao

[6] Sander T. and Tschdin C.F. (1998) 'Protecting Mobile Agents Against Malicious Hosts', *Mobile Agents and Security,* LNCS Vol. 1419, Springer-Verlag, pp. 44-60

[7] Kotzanikolaou P., Burmester M. and Chrissikopoulos V. (2000) 'Secure transactions with mobile agents in hostile environments', *Proc. The 5th Australian Conference on Information Security and Privacy (ACISP 2000),* LNCS 1841, pp.289-297

[8] Romao A. and Sliva M. M. (2001), 'Secure mobile agent digital signatures with proxy certificates', *E-Commerce Agents*, LNAI 2033, pp.206-220

[9] Varadharajan V. (2000) 'Security enhanced mobile agents', *Proceedings of the 7th ACM conference on Computer and Communications Security*, November 1 - 4, Athens, Greece, pp. 200 – 209

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

[11] Wang Y. and Tan K.-L. (2001) 'A secure model for the parallel dispatch of mobile agents', *Proc. of 4th International Conference of Information and Communications Security (ICICS2001)*, Springer-Verlag, LNCS Vol. 2229, pp 386-397

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

[13] CCITT Recommendation X. 509-1989 (1989) The Directory-Authentication Framework. Consultation Committee, International Telephone and Telegraph, International Telecommunication Union, Geneva

[13] Li T., Seng C. K. and Lam K. Y. (2000) 'A secure route structure for information gathering', *Proceedings of 2000 Pacific Rim International Conference on AI (PRICAI00),* pp 101-114

[14] http://www.trl.ibm.co.jp/aglets/

[15] http://java.sun.com/products/

[16] Rivest R.L., Shamir A., Adleman L. (1978), 'A method for obtaining digital signatures and public-key cryptosystems', *Communications of the ACM,* Vol.21, No.2, pp 120-126

[17] Wang Y., Tan K.-L. and Wang Y. (2003) 'A study of securing route structures for mobile agents dispatched in parallel', unpublished manuscript available upon request from authors.