Journal of Information Processing Vol.21 No.3 441–449 (July 2013)

[DOI: 10.2197/ipsjjip.21.441]

Regular Paper

A Site-Exit Router Selection Method Using Routing Header in IPv6 Site Multihoming

Yong Jin[1,a)] Takuya Yamaguchi[2,†1,b)] Nariyoshi Yamai[3,c)]
Kiyohiko Okayama[3,d)] Motonori Nakamura[4,e)]

Received: October 22, 2012, Accepted: March 1, 2013

Abstract: With proliferation of the Internet and its services, how to provide stable and efficient Internet services via reliable high-speed network has become an important issue. Multihomed network is attracted much attention to provide stable and efficient Internet services. In this paper, we focus on the multihoming method in the IPv6 environment. In the IPv6 environment, each host can be assigned multiple IP addresses from different ISPs on one network interface, thus the multihoming is relatively easier than that in the IPv4 environment. However, since many ISPs adopt ingress filtering for security concerns, a multihomed site should select a proper site-exit router according to the source IP address of the packet to communicate with the outside the site successfully. In most site-exit router selection methods, a kind of source IP address dependent routing method is introduced which has some problems in terms of high deployment cost and lack of fault-tolerance and so on.In this paper, we propose a new site-exit router selection method using the routing header which can indicate the router to pass through in the IPv6 environment. This method introduces two middlewares, one into the inside server and the other into the site-exit router. The one in the inside server attaches a routing header which indicates a specific site-exit router to pass through according to the source IP address of the packet, and the other in the site-exit router removes the attached routing header from the packet, thus the inside server can communicate with the outside the site successfully as usual. We also implemented a prototype system including the proposed inside server and the site-exit router and performed feature evaluation as well as performance evaluation. From the evaluation results, we confirmed the proposed method worked well and the overhead of the middlewares are acceptable for practical use in the real network environments.

Keywords: multihoming, IPv6, site-exit router selection, routing header,

1. Introduction

Nowadays, the Internet has been widely deployed as the social information infrastructure and the amount of users keeps increasing significantly. In the meantime, for the Internet services such as WWW (World Wide Web) and E-mail, the requirements have changed from providing basic services to high-reliability accessing and high performance. As a solution to such requirements, multihoming technology (we focus on site multihoming technology in this paper), which allows the user network (user site) to be connected to the Internet by two or more Internet Service Providers (ISPs), attracts much attention. By using the multihoming technology, the ISPs can improve the fault-tolerance and performance of the Internet systems by correspondingly using the multiple backbones according to the destinations and the network conditions.

The site multihoming mechanism in the IPv6 environment is different from that in the IPv4 environment. In general, multiple IP addresses (one from each prefix offered by the corresponding ISP) are assigned to each host in the site and each host selects one of them as the source IP address when sending out packets. In addition, most ISPs run ingress filtering [1], [2] for the security concerns, accordingly the sent out packet needs to be routed via the proper ISP based on its source IP address to avoid being filtered. However, basically the user site performs route control based on the destination IP address thus the packets with the same destination IP address are being transferred via the same ISP even if their source IP addresses are different. Consequently, some packets can be dropped by the ingress filtering. To solve this problem, a solution has been proposed in Ref. [3] which is using the Source Address Dependent (SAD) routing [4]. However, this approach requires SAD routing function to be deployment in all the routers in the SAD domain which may spans wide area in the user site and this creates a high introduction cost issue upon deploying in the real network environment.

In this paper, we propose a new site-exit router (the router connecting the user site to the corresponding ISP) selection method using the routing header in the IPv6 site multihoming. In this proposed method, the key point is that when the internal host sends

1 National Institute of Information and Communications Technology, Koganei, Tokyo 184–8795, Japan
2 Graduate School of Natural Science and Technology, Okayama University, Kita, Okayama 700–8530, Japan
3 Center for Information Technology and Management, Okayama University, Kita, Okayama 700–8530, Japan
4 National Institute of Informatics, Chiyoda, Tokyo 101–8430, Japan
†1 Presently with West Japan Railway Company.
a) yongj@nict.go.jp
b) yamaguti@dist.cne.okayama-u.ac.jp
c) yamai@cc.okayama-u.ac.jp
d) okayama@cc.okayama-u.ac.jp
e) motonori@nii.ac.jp

out a packet, the appropriate site-exit router's IP address which is corresponding to the source IP address will be indicated in the packet using the routing header. By using this method, the user site not only can perform routing based on the destination IP address as usual but also can avoid the ingress filtering. Note that the purpose of this paper is to construct a multihomed network that can bypass the ingress filtering. Thus, how to take advantages of the multihomed network is beyond the scope of this paper.

In the rest portion of this paper, we describe conventional IPv6 site multihoming method as well as its problems in Section 2 and in Section 3, we introduce the issues of the multihoming method which is using the routing header then we also propose a new approach to solve the problems. We evaluate the feature as well as the performance of the proposed method and discuss its applicable range in Section 4 and finally, we conclude this paper and introduce some future works in Section 5.

2. Conventional IPv6 Site Multihoming and Problems

2.1 Network Configuration

Figure 1 shows a typical multihomed network configuration we consider in this paper. The server site is connected to multiple ISPs (ISP A and B in the figure) and is allocated a prefix from each ISP which means each node in the server site is assigned multiple addresses within the prefixes. The site-exit routers RA and RB are set at the junctions of the ISPs and the server site. In general, these routers are set at geographically different locations, in other words, they belong to different network segments.

In the rest of this paper, we consider the scenario that the client outside the server site initializes connections to the server inside the server site. In this case, the server is allocated multiple IP addresses IP:A and IP:B by the ISPs A and B, respectively thus the client can select one of them as the destination IP address to communicate with the server. The IP address of each router inside the server site can belong to the ISP A or the ISP B but the outside IP addresses of the site-exit routers (RA (out) and RB (out)) should be the IP address allocated from the corresponding ISP.

We discuss the case that the inside server initializes the connection to the outside in Section 4.4.

2.2 Issues in the IPv6 Site Multihoming

In general, the communication route is controlled based on the destination IP address in the current Internet. However, in some cases the communication can be failed in the IPv6 site multihoming if the same routing policy is used. We describe the case in detail using **Fig. 2**.

We consider a scenario that the outside client initializes a connection to the inside server using the IP:A as the destination IP address based on the same network configuration as shown in Fig. 1. We present the inbound and outbound packet flows in this scenario using Fig. 2.

(1) The packet destined to the IP:A sent from the outside client is delivered to the inside server via the ISP A based on the routing protocol of the Internet.
(2) Then the inside server sends the response packet to the outside client. Here, the source IP address of the response packet is the IP:A. In the server site, the default route is set to the ISP B thus the response packet is relayed to the site-exit router RB.
(3) The response packet is dropped by the ingress filtering of the ISP B since the source IP address of the response packet does not belong to the prefix allocated by the ISP B.

As we can see from above operations, since the source IP address of the response packet needs to coincide with the destination IP address of the original initializing packet in the inbound communication, thus if we do not perform a route control to make the inbound route and the outbound route coincide then the communication might fail.

2.3 SAD Routing

As one solution to solve above ingress filtering problem, a technique called SAD (Source Address Dependent) routing has been approached. In this routing protocol, the routers in the server site determine the next-hop to relay the packet to based on the source IP address of the packet and for most routers this can be realized by introducing the PBR (Policy Based Routing) [5] feature.

We describe the packet flow when the server site uses the SAD routing using **Fig. 3**. In this figure, we assume all routers have the SAD routing feature and the packet with source IP address belonging to the ISP A's prefix Prefix (A) will be relayed to the next router which is destined to the ISP A. Similarly, the packet with the source IP address belonging to the ISP B's prefix Prefix (B) will be sent via the default route ISP B based on the conventional destination IP address base routing policy.

(1) The initial packet destined to the IP:A sent from the outside client is delivered to the inside server via the ISP A based on

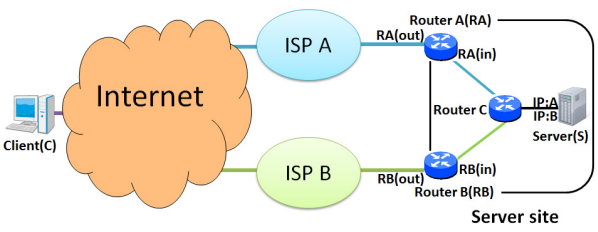

Fig. 1 IPv6 site multihoming configuration.

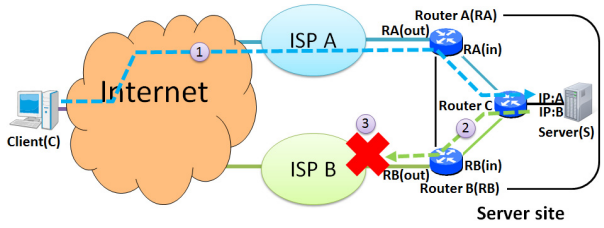

Fig. 2 Packet flow in conventional routing.

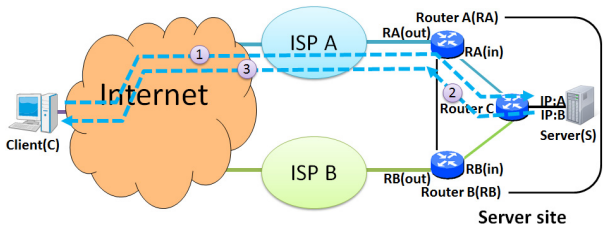

Fig. 3 Packet flow in SAD routing.

the routing protocol of the Internet.

(2) Then the inside server sends the response packet back to the outside client. Here, the source IP address of the response packet is set to the IP:A. The response packet will be sent to the Router C first and then the Router C relays it to the router A which is destined to the ISP A based on the SAD routing (since the source IP address IP:A belongs to the Prefix (A)).

(3) Similarly, the Router A relays the response packet to the ISP A based on the SAD routing.

With above operations, by using the SAD routing the server site can coincide the inbound and outbound routes of the communication, thus the response packet can avoid being dropped by the ingress filtering.

2.4 The Problem of the SAD Routing

Although the SAD routing can solve the ingress filtering problem, it is difficult to introduce the SAD routing into a large scale network environment and it also has problems in terms of high administrative cost and vulnerability of fault-tolerance. In this section, we describe these problems in detail.

An easy way to realize SAD routing is to introduce the PBR feature into the routers as mentioned in the previous section. However, in this method, the network administrator has to configure the PBR feature in most inside routers correctly and this causes high administrative cost. If the PBR features in some routers do not work correctly then it will cause communication failure which will increase burden of the network administrators. For example, in Fig. 3, if the PBR feature in the Router C does not work correctly then the router C will relay the response packet to the default router Router B and this may cause a dead loop between the Router C and the Router B. Furthermore, in most of current popular routers, the PBR feature is generally based on static SAD routing which may cause vulnerability of fault-tolerance. For instance, when the link between the Router C and the Router A fails in Fig. 3, even if there exist a bypass route between the Router C and the Router A the SAD routing protocol can not dynamically change the policy and use the alive route.

In order to improve fault-tolerance, an extension method of the SAD routing that performs dynamic SAD routing [3] has been proposed, too. This extension method can dynamically bypass the broken point when some failures happen and use the backup route for the communications thus it can approve the fault-tolerance of the Internet system. However, again this extension method has to be introduced to all routers perform SAD routing in the server site, as a result it is difficult to deploy the solution to the real network environments.

Moreover, another method that establishes a virtual connection between the site-exit routers to control the outbound route correctly has been proposed [6]. For example, in Fig. 3, when the IP packet with the source IP address belonging to the Prefix (A) arrives at the wrong site-exit router Router B then the Router B can relay it to the Router A via the virtual connection and then the packet can be delivered via the right route. However, this method causes unnecessary communication and also if the the Router B is broken then the communication via the Router A will fail also.

3. Site-exit Router Selection Using the Routing Header

As described in the previous section, since the conventional route control methods used the SAD routing for site-exit router selection thus each of them caused problems in introduction and operation or in improvement of fault-tolerance. In this section, we propose a new site-exit selection method which is not based on the SAD routing to alleviate those problems.

3.1 Overview of the Proposed Method

The SAD routing works well to coincide the inbound route of the initializing packet and the outbound route of the response packet. The key point is how to select an appropriate outbound route of the response packet according to the inbound route of the initializing packet. Thus we consider a way to indicate the outbound route of the response packet other than the SAD routing and also solve the problems it has. In this section, we propose a site-exit router selection method using the extension header for source routing of the IPv6, Routing Header [7], which can indicate the routers to pass through.

Our research group has proposed a multihoming method using LSRR (Loose Source and Record Route) option [8] in the IPv4 multihomed network environment [9] but it is only applicable to TCP communications. In this method, when an outside client sends an initial packet to an inside server to establish a TCP connection, the site-exit router attaches an LSRR option to the initial packet before relaying it to the inside server. In most UNIX/Linux derived operating systems, when they receive an LSRR option attached initialization packet they also automatically attaches an LSRR option to the latter packets for the same connection to make them pass through the same site-exit router. Consequently, the outbound route of the response packet can be coincided with the inbound route of the initialization packet.

However, in the IPv6 environment, we found that even in UNIX/Linux derived operating systems, when the site-exit router attached the routing header to the incoming packets the inside server do not attach the routing header automatically to the corresponding response packets. Thus, in this proposed method, the inside server rather than the site-exit router attaches the routing header to the corresponding response packet in order to send the packet via the appropriate site-exit router based on its source IP address. On the other hand, when the site-exit router receives a packet being attached the routing header, it erases the routing header from the packet before relaying it. This operation is necessary because the packet with the routing header may be dropped by the routers outside the server site or by the client. With above operations, the routers inside the server site except the site-exit routers only need to perform destination address based routing as usual, thus this proposed method is comparatively easy to deploy in real network environments. For the fault-tolerance, since the route between the inside server and the site-exit routers can be controlled by the conventional destination IP address base routing so that the bypass route can be used in case the default route is broken. As a result, the proposed method can alleviate the problems of the SAD routing as expected.

3.2 Routing Header Attachment

In the proposed method, we introduce a new middleware into the inside server to attach the routing header to the packet. The middleware catches the packets with source IP address belonging to the Prefix (A) and destination IP address belonging to the outside the server site then attaches the routing header indicating the internal IP address of the site-exit router RA (in)[*1] and sends it out. The type of the routing header can be any type based on the agreement between the site-exit routers and the inside server.

On the other hand, for packets with the source IP address belonging to the Prefix (B), if the routes of all packet destined to the outside the server site including the default route are set to the Router B then there is no need to attach the routing header to these packet. However, if above packets need to be sent via the Router A then the routing header is required for these packet.

Finally, for the packets destined to the inside nodes, no matter what is the source IP address the routing header is not necessary.

In the prototype system, considering the ease of experiments, we manually configured the middleware to have the prefix for the site-exit router and the IP address scope of the server site network, but we also plan to use dynamic method such as DHCPv6 [19] to automate those configurations in the future.

3.3 Routing Header Deletion

In the proposed method, the site-exit routers firstly pick out the real destination IP address of the packet from the routing header attached by the inside server and change it with the current destination IP address, secondly deletes the routing header from the packet and finally relays the packet to the corresponding ISP. Since normal routers do not have such features that we need to add these functions to the site-exit routers.

The most simple way is to use PC router. In this case, we can use the features of the operating system itself such as PF (Packet Filter) [10] and divert [11] of OpenBSD and pick up the routing header attached packet and process above operations. However, in some cases the PC router may be not appropriate because of the performance and feature problem. In such a case, we consider to use PBR feature in the site-exit routers and relay only the routing header attached packet to the PC router and perform above processes.

Moreover, we also consider a way to activate the type 2 routing header [12] which is used in the mobile IPv6 if the site-exit routers have this feature. In this case the type 2 routing header attached packet can be relayed to the ISP which is the same as usual and causes no problem. However, the use of type 2 routing header in site-exit router selection has been deprecated in Ref. [12], thus whether this feature works well or not depends on the implementation.

3.4 Overall Operations

We present the overall operation of the proposed method using in **Fig. 4**.

In this figure, we assume the outside client initializes a connection to the inside server using the IP:A as the destination IP address. The numbers in the figure correspond to the step numbers in the following.

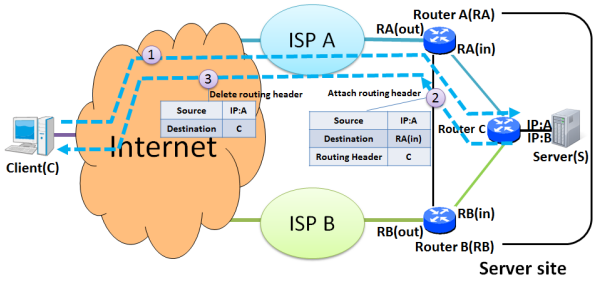

Fig. 4 Route control using the routing header.

(1) The packet destined to the IP:A from the outside client is sent to the inside server via the ISP A based on the routing protocol of the Internet.

(2) When the inside server sends the response packet to the outside client, it attaches a routing header indicating the inside IP address of the Router A RA (in) in order to make the response packet pass through the Router A. Here, the destination IP address of the packet is set to RA (in) thus the packet can be routed to the Router A based on the conventional destination IP address base routing protocol.

(3) When receiving the response packet, the Router A changes the destination IP address of the response packet to the real one which is set in the routing header and deletes the routing header from the packet then relays the packet to the ISP A.

3.5 Approach to Path MTU Reduction

As described in the previous section, a 24-byte routing header will be attached to the packet in the proposed method. This causes an increase of the packet size and consequently this increase may make the size of the packet exceed the MTU (Maximum Transfer Unit) of the inside server or the Path MTUs between the inside server and the site-exit routers. At the viewpoint of the inside server, it equals to 24-byte reduction of the Path MTU which means the inside server needs to adjust the payload of the packets according to the changed Path MTU.

To solve this problem, we add two functions to the middleware in the inside server. First, if the size of the packet being attached the routing header exceeds the MTU of the interface then the middleware sends the type 2 ICMPv6 message indicating "Packet Too Big" back to the operating system. In this message, the middleware informs the apparent MTU 24-byte smaller than the real one to the operating system [*2]. Second, if the middleware receives the ICMPv6 message indicating "Packet Too Big" from the Internet then it processes the following tasks.

(1) The middleware confirms if the original packet caused the MTU exceeding includes the routing header by checking the partial of the original packet included in the ICMPv6 message.

(2) If the original packet includes the routing header then the middleware creates a new ICMPv6 messages indicating a 24-byte smaller MTU than the real one and relays the mes-

[*1] In fact, here, the destination IP address of this packet is set to the RA (in) and the real destination IP address is set in the routing header.

[*2] For example, if the MTU of the interface is 1,500 bytes then the middleware sends back 1,476 bytes as new MTU.

sage to the operating system.

(3) If the original packet does not include the routing header then the middleware relays the message to the operating system without any change.

In the above operations, it is possible to adjust the size of the payload in the packet if the packet being attached the routing header causes MTU exceeding problem.

4. Implementation of the Prototype System and Evaluations

In order to confirm the proposed method satisfies its purpose we implemented a prototype system consists of the inside server with the routing header attachment feature and the site-exit router with the routing header deletion function based on the proposed method and evaluated its feature and performance. The feature evaluation is necessary to confirm if the proposed method works as we expected and the performance evaluation is also required to verify if the proposed method can be applied to real network environment so that we can conclude if the proposed method solves conventional problems. In this section, we describe the implementation method and the feature evaluation as well as the performance evaluation of the prototype system. We also discuss about the applicable range of the proposed method here.

4.1 Implementation of the Prototype System

In the implementation, since our main purpose is to verify if the proposed method works well as we expected that we used PC router as site-exit router for the simplicity, and also we used OpenBSD as the operation system in both the inside server and the site-exit router. This is because the divert function of the IPv6 is available in the OpenBSD. For the routing header, we used type 0 routing header of the IPv6. The type 0 routing header has been deprecated in Ref. [13] for security concerns but as we described in Section 3.2, any type of routing header can be used if there is an agreement between the inside server the site-exit router. Moreover, it is simply able to activate the process of the type 0 routing header in the OpenBSD so that we adopted the type 0 routing header in the implementation. Note that in the prototype system we only installed the proposed method to the site-exit router and the inside server as middleware. This means the deployment cost of the proposed method is much smaller than the conventional SAD method since the SAD protocol needs to be installed to the inside server and all the routers in the SAD domain.

We present the internal configuration of the inside server in **Fig. 5**. The Packet Filter picks up the packets need to attach the routing header (as described in Section 3.2) from the packets sent by the operating system kernel and delivers those packets to the attach routing header program via the divert socket. Then the attach routing header program attaches the routing header to those packets and sends out them via the divert socket. Also, the Packet Filter delivers the "Packet Too Big" ICMPv6 message to the adjust MTU program via the divert socket and then the adjust MTU program changes the MTU size if necessary and delivers the ICMPv6 message to the operating system kernel.

Similarly, **Fig. 6** shows the internal configuration of the site-exit router. In the site-exit router, the routing header attached

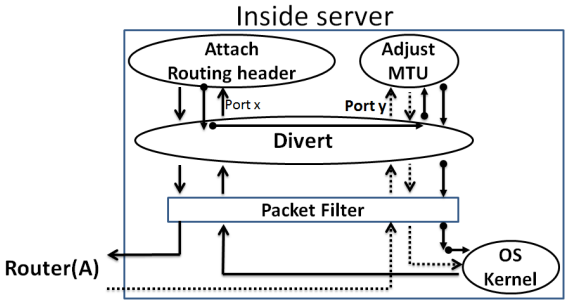

Fig. 5 Inside server configuration.

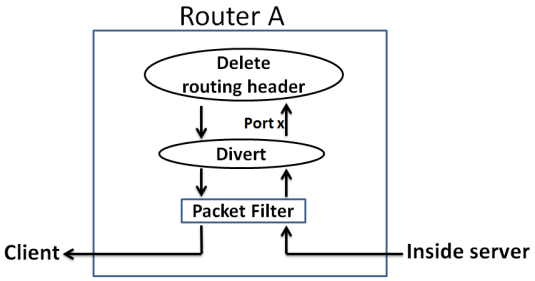

Fig. 6 Site-exit router configuration.

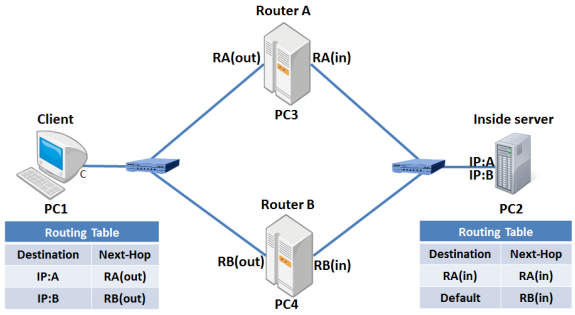

Fig. 7 Evaluation network environment.

Table 1 The specs of the PCs.

PC	CPU	Memory	
PC1	Core2 Duo 2.93 GHz	2 GB	FreeBSD 8.2
PC2	Core2 Duo 2.93 GHz	2 GB	OpenBSD 5.0
PC3	Core2 Duo 2.93 GHz	2 GB	OpenBSD 5.0
PC4	Core2 Duo 2.93 GHz	2 GB	FreeBSD 8.2

packet is delivered to the delete routing header program via the Packet Filter and the divert socket then the delete routing header program deletes the routing header from the packet and sends the packet out via the divert socket.

4.2 Feature Evaluation

First, we performed feature evaluation using the prototype inside server and the site-exit router to confirm if the communication between the outside client and the inside server can succeed. **Figure 7** shows the evaluation network environment and **Table 1** shows the specs of the PCs used in the evaluation. We used 100BaseTX for all links in the evaluation environment.

In the evaluation environment, the default gateway of the inside server (PC2) is set to the normal router (without any proposed feature) Router B (PC4) and for the outside client (PC1), the next-hop for the destination IP:A is set to the Router A (the prototype router of the proposed method) and the next-hop for the destination IP:B is set to the Router B.

In the feature evaluation, we set up an HTTP server on the inside server and make the outside client access it via the IP:A and the IP:B individually by indicating the IP addresses directly. In fact, when we operate the proposed method in the real network environment, we can simply register both the IP:A and IP:B as A resource records of the inside server's FQDN (Fully Qualified Domain Name) in the DNS (Domain Name System) and let the outside client select one of them to access the inside server. Furthermore, we can use some dynamic route selection methods based on practical use of the DNS such as the approaches of Refs. [14], [15] and make the outside client select a proper route according to the network conditions to access the inside server.

In the feature evaluation, we also monitored the routes that the packets pass through and the existence of the routing header in the packet as well as the packet size on the inside server using the tcpdump [16]. As a result, we confirmed that the corresponding site-exit router was selected properly in both cases of the IP:A and the IP:B were indicated as the destination IP address. Furthermore, we confirmed the routing header was attached in the inside server and deleted in the site-exit router correctly when the Router A was selected as the site-exit router. Finally, we also confirmed the packet size was changed to 1,476 bytes on the link between the inside server and the Router A which means the MTU was adjusted properly.

In addition, we also confirmed the proposed method worked well when we set a commercial L3 switch between the site-exit router and the inside server. We show the evaluation environment in **Fig. 8**. In this evaluation, we confirmed the L3 switch transfered the packet with routing header attached to the indicated site-exit router as we expected. This evaluation result confirmed the proposed method only needs to be installed in the site-exit router and the inside server.

4.3 Performance Evaluations

As explained in the previous section, the proposed site-exit router selection method has to deal with a 24-bytes routing header attachment and deletion processes as well as the path MTU reduction because of the additional routing header. The additional processes and path MTU reduction may cause higher overhead and lower performance in the proposed method than in the conventional route control. Therefore, we ran performance evaluations to measure the decrease of throughput and the increase of delay in the proposed method using the same network environment of Fig. 7 by comparing the overheads as well as the delays between the prototype inside server/site-exit router and the normal inside server/site-exit router.

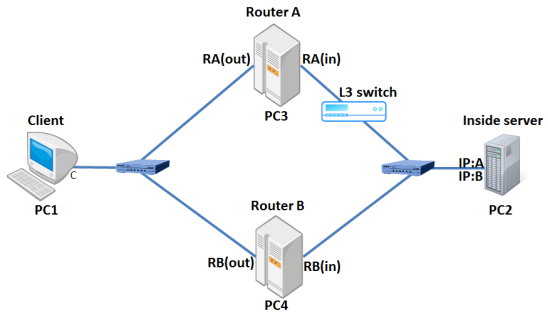

Fig. 8 Evaluation network configuration with L3 switch.

In the performance evaluations, we configured the PC3 and the PC2 to normal site-exit router and the normal inside server for the existing system, while to the prototype site-exit router and the prototype inside server for the proposed system. When we perform the experiment in the proposed system, in the prototype inside server (PC2) we configured the Packet Filter to pick up corresponding packets and deliver them to the Attach Routing header program via the Divert socket and then the Attach Routing header program sends out the packets after attaching routing header. For the "Packet Too Big" ICMPv6 packets the Packet Filter delivers them to the Adjust MTU program via the Divert socket and then the Adjust MTU program adjust the size of the MTU if necessary and sends to the operating system kernel as we described in Section 4.1. On the other hand, in the prototype site-exit router, we configured the Packet Filter to pick up corresponding packets and deliver them to the Delete routing header program via the Divert socket and then the Delete route header program sends the packets out after deleting the routing header.

Using these network configurations we measured the TCP throughput between the PC1 (client) and the PC2 (server) using the iperf [17]. For the delay measurement, we used TCP and measured the delay from sending the SYN packet to receiving the SYN+ACK packet between the PC1 (client) and the PC2 (server). Then we compared the measured TCP throughput and the delays in order to compare the performance of the normal system and the proposed system. **Tables 2** (inbound) and **3** (outbound) shows the measured throughputs in the inbound direction and outbound direction individually. **Table 4** shows the measured delays from the PC1 (client) to the PC2 (server). From the results, first, we can conclude that the overhead of the prototype system and delay caused by the proposed method is small enough to be practically used in the real network environment. Then, we can see that the 1.6% of throughput reduction is relatively bigger in the outbound direction. We consider this overhead caused by the 24-byte payload of the routing header and the reduction is reasonable and acceptable.

4.4 Discussion of Applicable Range

Since some prerequisites are required to apply the proposed method that in some network environment the proposed method

Table 2 Result of throughput measurement — Inbound direction.

PC2, PC3 category	Throughput	Gap with normal
Normal router	88.56 Mbps	-
Prototype router	88.48 Mbps	0.1%

Table 3 Result of throughput measurement — Outbound direction.

PC2, PC3 category	Throughput	Gap with normal
Normal router	88.48 Mbps	-
Prototype router	87.04 Mbps	1.6%

Table 4 Result of delay measurement.

PC2, PC3 category	Delay	Gap with normal
Normal router	0.18 ms	-
Prototype router	0.23 ms	0.05 ms

may be not applicable. Thus, in this section we discuss the applicable range of the proposed method.

4.4.1 Disable the Routing Header

The Type 0 Routing Header has been deprecated for security concerns in Ref. [13]. However, the Type 0 Routing Header of the IPv6 is different from thee LSRR option of the IPv4, it basically is not checked on the routers which are not indicated to pass through. Thus, the proposed method will work well unless there exist some routers that are configured to check and drop the type 0 routing header attached packets in the server side network. Furthermore, the packets with the type 0 routing header attached will not be relayed outside the server side network in the proposed method which means the packet will not cause the same problem outside the server site. Finally, the packets with the type 0 routing header attached can be filtered by the site-exit router at the edge of the server site network thus some kinds of attacks using the type 0 routing header will not be a security problem to the server site network.

4.4.2 Apply for the Outbound Connection

Since in the proposed method, the routing header indicates the site-exit router corresponding to the source IP address of the packet, we consider the proposed method is applicable to not only inbound connections but also outbound connections. For the outbound connections, the selection of the source IP address remains as a research topic and the reference [18] provides some standard procedures for default address selection. However, the proposed method can select the proper site-exit router based on the source IP address despite of what kind of source IP address selection method used.

4.4.3 Multiplication of Site-exit Router

Considering redundancy, in some organizations it is possible to set multiple site-exit routers for one single ISP. For example, in an university with several remote campuses, each campus can be connected with multiple different ISPs. In such a network configuration, since the proposed method can indicate only one IP address for one prefix that it can not select proper one from multiple site-exit routers against to one ISP. However, if we configure a virtual IP address for the multiple site-exit routers and each site-exit router advertise the route information for the virtual IP address with an appropriate cost then the inside server can select the one with the smallest cost.

On the other hand, we also discuss the changing of site-exit router when one of them is down. Basically, in this paper we consider the route selection is performed per connection and we have proposed some route selection mechanism by practically using DNS responses. When one of the site-exit routers is down in the middle of communication, the session can not be recovered. However, if the end node tries reconnection it can communicate using another available site-exit router since we can configure the TTL of the DNS record with a small value in order to let the end node perform name resolution again. Furthermore, in case of multiple site-exit routers exist for one ISP, we also can use anycast technology to bypass the down site-exit router and use the available one for communication. In fact, in the SAD method, when one of the site-exit routers is down during the communication, the communication can not be recovered. From this viewpoint, the effect is the same as in the SAD method if the proposed method only provides redundant routers for communication per connection (or per flow if includes UDP communication) without using anycast technology.

4.4.4 Scalability and Practicability

The scalability and practicability are important factors to be considered when we expect to apply the proposed method to the real network environment. Thus, we discuss the factors and feasibility by comparing the proposed method with the SAD method here.

Firstly, the scalability described here means that even if the scale of the network is very large the routers between the site-exit router and the inside server do not need to install the proposed method. We have described in previous sections that the proposed method needs to be installed in the inside server and site-exit routers while the SAD method needs to be installed in all the routers in SAD domain. Basically, we consider that middleware installation is easier than new feature addition. The proposed method can be applied to the inside server and site-exit routers as middleware while the SAD method needs to be installed as new feature to the routers. So in cases where there are many inside servers, from the number of machines needs to be configured with the method (proposed method or SAD) it seems the SAD method is easy to deploy. However, although the SAD method only needs to be installed in the routers within the SAD domain, but the SAD method is not previously available in the normal routers, thus it needs to be newly added to the routers or static policy routing needs to be configured by the network administrators. Accordingly, although Ref. [3] proposed a dynamic SAD method but the method also needs to be newly added to many existing routers in the SAD domain. On the other hand, in the proposed method, although the new feature needs to be installed in site-exit routers as well as in many inside servers, it is a kind of software update. Thus the deployment is much easier. Therefore, we can say the scalability of the proposed method is better than the SAD method.

Secondly, the proposed method can be introduced to the network step by step which means the new feature does not have to be installed to all the inside servers in the proposed method at one time. In other words, a non-compliant node can exist in the end node site in which the proposed method applied. In this case, we consider the proposed method can collaborate with some kind of tunneling technology such as the one mentioned in Ref. [3]. In such collaboration, when the packet from the non-compliant node reaches the wrong site-exit router, the wrong site-exit router can transfer the packet to the proper site-exit router through the tunnel. As a result, although the packet may be delivered via a bypass route which may cause longer delay, the communication can be successful. Instead, in the SAD method, the SAD routing feature has to be installed and configured appropriately in all the routers within the SAD domain at once. Therefore, we can conclude that the practicability of the proposed method is better than the SAD method.

Above all, we consider the proposed method has advantages in scalability and practicability.

5. Conclusion

In this paper, we proposed a proper site-exit router selection method based on the source IP address by using the routing header in the IPv6 site multihoming environment. We also implemented the prototype system of the proposed method and confirmed the proper site-exit router was selected appropriately as well as the overhead of the proposed method was small enough for practical use in the real network environment. According to the evaluation results, we can conclude that the existing problems of the site-exit router selection in the conventional SAD method can be alleviated by using this approach.

For future works, we consider how to evaluate the proposed method in the real network environment in collaboration with the dynamic traffic balancing feature using the DNS [14], [15]. Moreover, an automatic configuration mechanism by using DHCPv6 which mentioned in Section 3.2 is also included in the future work.

References

[1] Ferguson, P. and Senie, D.: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, RFC2827, IETF (May 2000).

[2] Baker, F. and Savola, P.: Ingress Filtering for Multihomed Networks, RFC3704, IETF (Mar. 2004).

[3] Ohira, K. and Okabe, Y.: Host-Centric Site-Exit Router Selection in IPv6 Site Multihoming Environment, *Proc. 1st International Workshop on Protocols and Applications with Multi-Homing Support* (*PAMS 2011*), pp.696–703 (2011).

[4] Bagnulo, M., Garcia-Martinez, A., Rodriguez, J. and Azcorra, A.: End-site routing support for IPv6 multihoming, *Computer Communications Journal*, Vol.29, No.7, pp.893–899 (0000).

[5] Cisco Systems Inc.: Policy-Based Routing (online), available from ⟨http://www.cisco.com/warp/public/cc/pd/iosw/tech/plicy_wp.pdf⟩ (accessed 2012-04-18).

[6] Huitema, C., Draves, R. and Bagnulo, M.: Ingress filtering compatibility for IPv6 multihomed sites (online), available from ⟨http://ops.ietf.org/multi6/ietf61/IETF61_ingress_filter.pdf⟩ (accessed 2012-04-18).

[7] Deering, S. and Hinden, R.: Internet Protocol, Version 6 (IPv6) Specification, RFC2460, IETF (1998).

[8] Postel, J. (Ed.): Internet Protocol, RFC 791, IETF (1981).

[9] Jin, Y., Yamaguchi, T., Yamai, N., Okayama, K. and Nakamura, M.: NAT-based Multihoming Method Applicable to Inbound Connection (Recommended Paper) (In Japanese), *J. Inf. Process.*, Vol.52, No.12 (2011).

[10] PF: The OpenBSD Packet Filter (online), available from ⟨http://www.openbsd.org/faq/pf/⟩ (accessed 2012-04-18).

[11] divert - kernel packet diversion mechanism, OpenBSD Programmer's Manual (online), available from ⟨http://www.openbsd.org/cgi-bin/man.cgi?query=divert⟩ (accessed 2012-04-18).

[12] Johnson, D., Perkins, C. and Arkko, J.: Mobility Support in IPv6, RFC3775, IETF (June 2004).

[13] Abley, J., Savola, P. and Neville-Neil, G.: Deprecation of Type 0 Routing Headers in IPv6, RFC5095, IETF (Dec. 2007).

[14] Jin, Y., Yamai, N., Okayama, K., Seike, T. and Nakamura, M.: A Dynamic Route Selection Method Using Multiple DNS Replies for Inbound E-mail Delivery on MUltihomed Environment (In Japanese), *J. Inf. Process.*, Vol.51, No.3 (2010).

[15] Jin, Y., Yamai, N., Okayama, K. and Nakamura, M.: An Adaptive Route Selection Mechanism Per Connection Based on Multipath DNS Round Trip Time on Multihomed Networks, *J. Inf. Process.*, Vol.20, No.2, to appear (2012).

[16] TCPDUMP/LIBPCAP public repository (online), available from ⟨http://www.tcpdump.org/⟩ (accessed 2012-04-18).

[17] Iperf Project (online), available from ⟨http://iperf.sourceforge.net/⟩ (accessed 2012-04-18).

[18] Draves, R.: Default Address Selection for Internet Protocol version 6 (IPv6), RFC3484, IETF (Feb. 2003).

[19] Droms, R. (Ed.), Bound, J., Volz, B., Lemon, T., Perkins, C. and Carney, M.: Dynamic Host Configuration Protocol for IPv6 (DHCPv6), RFC3315, IETF (July 2003).

Yong Jin received his M.E. degree in electronic and information systems engineering and Ph.D. degree in Industrial Innovation Sciences from Okayama University, Japan in 2009 and 2012, respectively. In April 2012, He joined the Network Architecture Laboratory of the Photonic Network Research Institute in the National Institute of Information and Communications Technology, Japan, as a researcher. His research interests include network architecture, traffic engineering and Internet. He is a member of IPSJ and IEICE.

Takuya Yamaguchi received his B.E. and M.E degrees in engineering from Okayama University, Japan, in 2011 and 2013, respectively. He is currently with the West Japan Railway Company. His research interests include distributed system, network architecture and Internet.

Nariyoshi Yamai received his B.E. and M.E. degrees in electronic engineering and his Ph.D. degree in information and computer science from Osaka University, Osaka, Japan, in 1984, 1986 and 1993, respectively. In April 1988, he joined the Department of Information Engineering, Nara National College of Technology, as a research associate. From April 1990 to March 1994, he was an Assistant Professor in the same department. In April 1994, he joined the Education Center for Information Processing, Osaka University, as a research associate. In April 1995, he joined the Computation Center, Osaka University, as an assistant professor. From November 1997 to March 2006, he joined the Computer Center, Okayama University, as an associate professor. Since April 2006, he has been a professor in Information Technology Center (at present, Center for Information Technology and Management), Okayama University. His research interests include distributed system, network architecture and Internet. He is a member of IEICE and IEEE.

Kiyohiko Okayama received his B.S., M.S. and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1990, 1992 and 2001, respectively. After he has worked in the Department of Information System at Osaka University and in the Graduate School of Information Science at Nara Institute of Science and Technology as a research associate, he joined the Department of Communication Network Engineering at Okayama University in 2000. From 2005 to 2011, he joined the Information Technology Center at Okayama University. Since 2011, he has been an associate professor in Center for Information Technology and Management at Okayama University. His research interests include network design and network security. He is a member of IEICE.

Motonori Nakamura graduated from Kyoto University, Japan, where he received his B.E., M.E. and Ph.D. degrees in engineering in 1989, 1991 and 1996, respectively. From 1994, he was an assistant professor at Ritsumeikan University. From 1995, he was an associate professor at Kyoto University. Currently he is a professor at National Institute of Informatics, Japan (NII) and the Graduate University for Advanced Studies (SOKENDAI). His research interests are message transport network systems, network communications, next generation Internet and Identity & Access Management. He is a member of IEEE, ISOC, IEICE and JSSST.