# IPv6 Anycast for Simple and Effective Service-Oriented Communications

*Satoshi Doi, Osaka University, Shingo Ata, Osaka City University, Hiroshi Kitamura, NEC Corporation, and Masayuki Murata, Osaka University*

## ABSTRACT

Although anycast communication supports service-oriented addresses, many of its current definitions in IPv6 are unclear. Furthermore, since there are no protocol standards or even a consensus on controlling routing, intersegment anycast communications are not yet available. In this article we first review IPv6-based anycast communication. At present, there are several possible applications that are suited to this. We then raise several problems and provide possible solutions to these. Based on this background, we present the Anycast Address Resolving Protocol (AARP) to establish TCP connections with a specific anycast address, and then propose a routing protocol for intersegment anycasts. Our proposed architecture makes anycast addresses more useful without (or with at most minimal) need for modifications/extensions to existing applications and/or upper-layer protocols.

## INTRODUCTION

Anycasting is a new networking paradigm supporting service-oriented addresses where an identical address can be assigned to multiple nodes providing a specific service. An anycast packet (i.e., one with an anycast destination address) is delivered to one of these nodes with the same anycast address. Anycast was first defined in RFC 1546 [1], which stated that the motivation for anycasting was to drastically simplify the task of finding an appropriate server on the Internet. The basic idea behind anycast communication is to separate the logical service identifier from the physical host equipment, that is, the anycast address is assigned on a type-of-service basis, which enables the network service to act as a logical host.

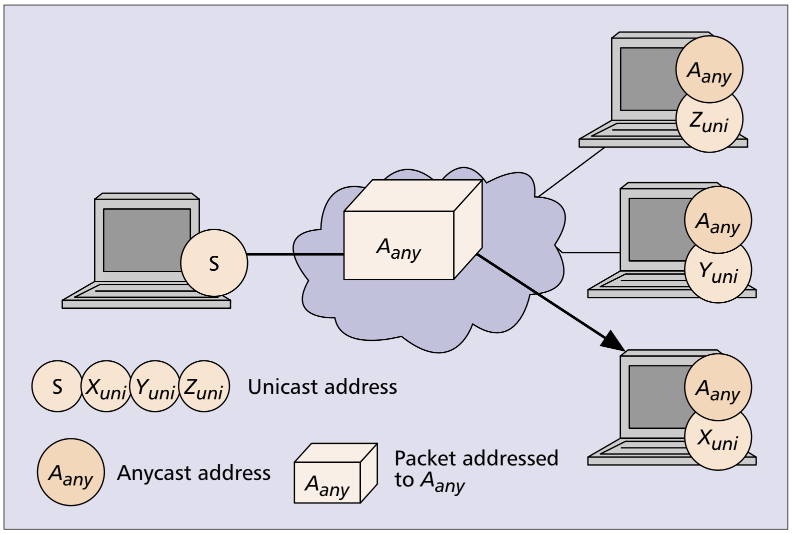
The Internet Protocol version 6 (IPv6) has three types of IP addresses, that is, unicast and multicast addresses as in IPv4, and an anycast address that is the subject of the current article. Table 1 summarizes the forms of communication for these addresses. A unicast address is a unique identifier for each network interface, and multiple interfaces must not be assigned the same unicast address. Packets with the same destination address are sent to the same node. A multicast address, on the other hand, is assigned to a group of nodes, that is, all group members have the same multicast address and packets for this address are sent to all members simultaneously. Like a multicast address, a single anycast address is assigned to multiple nodes (called *anycast membership*), but unlike multicasting, only one member of the assigned anycast address communicates with the originator at a time. Figure 1 has an example of anycast communication. There are three nodes associated with the anycast address $A_{any}$. When the source node sends a packet where the destination address is $A_{any}$, the packet is sent to one of three nodes ($X_{uni}$ in this figure), not to all hosts. The advantage of anycasting is that the source node can receive a specific service without knowledge of current conditions in service nodes and/or networks. When host $X_{uni}$ goes down, the packet for $A_{any}$ can be sent to another host ($Y_{uni}$ or $Z_{uni}$) (Fig. 1). How appropriately the destination node is chosen from anycast membership depends on the anycast routing protocol, which will be discussed later.

However, IPv6 anycasting still has several problems that need to be clarified within the context of the current specifications. First, we should have clear answers to "what kinds of applications are suited to using anycasting?" and "what are the advantages/disadvantages of using anycasting for applications?" Another problem with IPv6-based anycasting is that a routing protocol has not been included in its specifications, which is indispensable in making anycasting more widespread. The router should play an active role in deciding the destination network so that anycast packets can be appropriately forwarded. We need to design and implement an anycast routing protocol that is suited to anycast applications. We also need a migration scenario where the Internet will gradually be able to support anycasting. For example, anycast routing should work adequately (although not necessarily optimally) even when only a few nodes and/or routers support anycast routing within the Internet. We will discuss problems with anycast rout-

0163-6804/04/$20.00 © 2004 IEEE

| | Unicast | Multicast | Anycast |
|---|---|---|---|
| Communication form | Point to point | Point to multipoint | Point to point |
| Target of address | Node | Group | Service type |
| Membership | Single | Multiple | Multiple |
| Roles in C/S model | Both | Client (listener) | Server |

**Table 1.** *IPv6 address types.*



$A_{any}$
$Z_{uni}$
$A_{any}$
$Y_{uni}$
S
$A_{any}$
$A_{any}$
$X_{uni}$
S $X_{uni}$ $Y_{uni}$ $Z_{uni}$ Unicast address
$A_{any}$ Anycast address
$A_{any}$ Packet addressed to $A_{any}$

**Figure 1.** *Anycast communication.*

ing in the third section, and present our proposal in the fifth.

We also need to identify how stateful applications utilize anycasting in designing their routing protocols. Internet applications using all TCP-based or some UDP-based protocols are *stateful*; that is, end hosts establish the conditions of communication with each other and assume that their partners are identical during the exchange. This is very important because the current definition of anycasting is essentially *stateless*; that is, the destination host should be determined on a packet-by-packet basis by the routers. We will discuss our solution to this problem in the fourth section.

Of course, security considerations are also very important in anycast communications. A malicious node may *hijack* communication by announcing a spoofed advertisement through which it receives packets for the anycast address. Thus, some authentication mechanism is necessary to actually validate anycasting. Public-key-based authentication is one promising approach to counteract this problem, but further considerations of security are beyond the scope of the current article.

Finally, we need to note that the notion of anycasting is not only limited to the network (i.e., IP) layer, but can also be achieved in other (e.g., application) layers. As discussed in the next section, network- and application-layer anycasting have both strengths and weakness, and we focus on network-layer anycasting in this article.

The rest of this article is organized as follows. The next section discusses applications suited (and unsuited) to network-layer anycast communications. The third section is devoted to an introduction to the current implementations of anycasting and associated problems. Following that, we present our approach, the Anycast Address Resolving Protocol (AARP) that enables TCP communications utilizing anycast addresses. The fifth section discusses our routing scheme for intersegment anycasting, where scalability and deployment are taken into account. The final section concludes the article.

## ANYCAST APPLICATIONS

This section reviews what kinds of applications are suited to anycast communication. An excellent survey on IPv6 anycast addresses can be found in [2], where they introduce several applications that can be achieved through anycasting. One important example is *server location*, through which the sender host can choose one of many functionally identical hosts. As a result, load distribution among anycast hosts can be achieved if we utilize some appropriate anycast routing method, where anycast requests are evenly distributed to hosts. However, a simple method such as randomization among anycast hosts may not be sufficient since it is difficult to take the status of the resources of each server, such as the CPU load, into account in network-layer anycasting. Instead, application-layer anycasting should be employed in this case.

Another example is *service location* [2], where the sender host can communicate with an optimal (e.g., minimum delay or largest throughput) host chosen from multiple anycast hosts by specifying the anycast address. This is especially useful in dynamically changing environments such as mobile ad hoc networks. While this kind of service can be obtained through application-layer anycasting, the node can communicate automatically with an appropriate (e.g., nearest) server through network-layer anycasting.

In summary, the advantage of network-layer anycasting lies essentially in providing a simple mechanism where the source node can receive a specific service without the knowledge of service nodes and/or networks. However, this immediately implies that it is difficult to obtain rich functionalities, and the additional anycast examples listed below clearly demonstrate this.

### HOST AUTO-CONFIGURATION (PLUG & PLAY)

By defining and assigning a well-known anycast address to widely used applications (e.g., Domain Name Services [DNS] and proxy services), the user can reach these without knowing the location (i.e., their unicast address) of the server [1]. Moreover, the user can utilize these applications everywhere by specifying a well-known anycast address. DNS resolvers would no longer have to be configured with the IP addresses of their DNS servers, and it would be sufficient to send a query to a well-known DNS anycast address. This functionality can also be used for plug & play. Auto-configuration through anycasting is quite effective during the primitive setup phase (e.g., a DNS server cannot be used). When a host is plugged in, its IPv6 address is configured, automatically. However, to achieve true plug & play, various settings are necessary (e.g., configuring unicast addresses of DNS and proxy servers). If a well-known anycast address is installed in the hardware beforehand, end users can utilize these services without configuration.

Since auto-configuration of hosts is often done to discover locally provided servers (e.g., DNS, Simple Mail Transfer Protocol [SMTP], or proxy servers), anycast routing protocol crossing segments are not mandatory, which means that the scalability problem previously mentioned is not an issue here.

### THE GATE TO OVERLAY NETWORK

The *gate to overlay network* is another example of an anycast application. A distributed application like a peer-to-peer (P2P) service constructs a logical network topology among nodes participating in the service. However, the peer needs to know the address to connect the logical network prior to participating in the service. Each peer only specifies the anycast address in order to participate in the logical network, and one of the participating peers becomes the *gate of the logical network* for the new node, which should be determined by the anycast routing protocol. The advantage of this model is that all processes are completed within its own protocol. Furthermore, even when the connected peer leaves the logical network, it is possible to continue participating via another peer, which is automatically changed by the anycast routing protocol. This cannot be attained with any of the existing technologies.

### IMPROVING SYSTEM RELIABILITY

Anycasting permits multiple hosts with the same address; by increasing the number of hosts, system reliability can be improved because it still works even if some of these fail.

Through a mechanism similar to this, we would have services with better properties to tolerate faults. However, if anycast packets are destined for other segments, we again need anycast routing.

## PROBLEMS AND POSSIBLE SOLUTIONS IN IPV6 ANYCASTING

This section discusses problems that remain with the current specifications for IPv6 anycasting because they contain too few definitions. It also reviews several solutions.

### HOW A HOST ANNOUNCES ITS PARTICIPATION IN ANYCAST MEMBERSHIP

There are no standards for nodes to announce that they can receive anycast packets except for publishing routing information for anycast addresses (i.e., the node must be a *router* in the IPv6 specifications). This implies that a host (i.e., not a router) that intends to participate in (or leave) anycast membership must have a different capability of notifying the nearest anycast router of its status (joining/leaving).

### HOW AN UPPER-LAYER STATEFUL PROTOCOL IS SUPPORTED

Anycasting has a stateless nature, where it cannot ensure that all packets belonging to the same anycast address will go to the same destination node.

However, this leads to serious problems in that stateful protocols like TCP cannot be supported. When a host initiates a TCP connection to an anycast address, the receiving host cannot set its own anycast address as the source address for the acknowledgment packet. The IPv6 specifications prohibit the anycast address from being set into the source address field of the packet header. This is basically because an IPv6 anycast address does not identify a single source node. If the protocol allowed the anycast address to be set into the source address of the packet, the receiving host could not be sure that all packets sent during the communication had come from the same host. This means that the host cannot receive an acknowledgment for a packet sent to an anycast address. Weber and Cheng [2] recently discussed the *anycast address mapper* proposed by Oe and Yamaguchi [3]. It translates anycast addresses to corresponding unicast addresses at the host receiving anycast packets; this is done prior to anycast communication. However, each application must be modified to use the anycast address mapper to map an anycast address before communication begins. Our solution is similar to the anycast address mapper, but there is no need to modify upper-layer protocols or applications. This is discussed later.

### HOW ANYCAST ROUTING IS ACHIEVED

The current anycast standard does not define the routing protocol, and there are several challenging issues that need to be resolved in designing anycast routing protocols.

***Scalability*** — The routing entries for anycast addresses cannot be aggregated because anycast membership locations are widespread regardless of their actual prefix. Hence, routing entries for anycast addresses should be stored individually on the router. It is easy to imagine explosions in routing tables as anycast addresses get to be more widely used.

***Criteria for Selecting Anycast Membership*** — The meaning of *appropriate* needs to differ among applications. For example, if an application requires a faster response, the propagation delay between the source node and anycast node is extremely important; the nearest node for anycast membership should be chosen. The criteria for anycast routing strongly affect anycast communication capabilities.

***Security Issues*** — Maintaining anycast membership is particularly important. The easiest way for a host to gain membership is for it to simply advertise the routing entry for the associated anycast address to the router. However, such an approach can sometimes lead to serious security problems in that the anycast host can freely add or delete anycast entries in the routing table.

*The routing entries for anycast addresses cannot be aggregated because anycast membership locations are widespread regardless of their actual prefix. Hence, routing entries for anycast addresses should be stored individually on the router.*

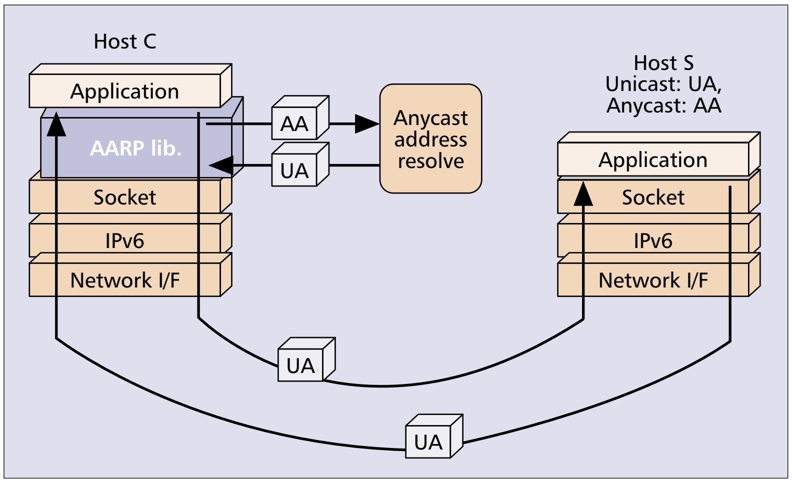


**Figure 2.** *The protocol stack of AARP.*

One important feature of anycast addresses is that they should be assigned from the same address space as a unicast address and are thus syntactically indistinguishable from unicast addresses. RFC 1546 originally recommended assigning anycasting its own address space because it expected this to greatly reduce the risk of applications mistakenly failing to recognize anycast addresses. However, when we consider the deployment of anycast routers, it is very likely that some routers on the Internet will not be able to process anycast addresses. If these addresses are allocated in a unicast address space, it is not necessary for legacy routers to deploy special operations for communication; these simply pass on anycast packets through unicast forwarding, expecting that packets can be reached. As it is difficult for an anycast router to decide whether the receiving packet's destination address is anycast or unicast, designing an anycast routing protocol is problematic.

There have been several proposals for an anycast routing protocol [2], but to the best of our knowledge none of these have conformed to IPv6 anycast specifications, and anycast addresses are allocated in their own address space, which is different from the unicast address space. However, the routing protocol we propose below allows the same space to be used for both unicast and anycast addresses.

## Anycast Address Resolving Protocol

Our proposal fills a gap between anycast and upper-layer protocols like TCP and UDP without the need to modify applications or protocols [4]. More specifically, the task of the AARP is to resolve the anycast address specified by the application into the corresponding unicast address. Figure 2 outlines the protocol stack for anycast communication with the AARP. The AARP is implemented as a kind of dynamic link library (DLL) that overwrites the original (i.e., the provided operating system) application programming interfaces (APIs). We call this library the AARP Library (AARP Lib in Fig. 2), which provides the same set of APIs as the original IPv6 socket APIs, and hooks them to resolve anycast addresses. It converts an anycast address into its corresponding unicast address prior to calling the original APIs. The anycast address is only used in the application and AARP Library layers. Layers below the AARP Library are not aware of the anycast address, and only handle the translated unicast address.

### Address Resolving Process in AARP

When host C wants to establish anycast communication with a host whose anycast address is AA, the process for anycast address resolution is as follows:

1) Host C calls the socket API (e.g., `connect()` in TCP) with the anycast address (AA) within its parameters. The AARP Library's API is called instead of the socket layer's API.

2) The AARP Library converts the anycast address into the unicast address (UA) in the callee function.

3) After conversion, the AARP Library calls the original socket API through the UA.

4) After communication has been established, all packets from host C are given the UA in their destination addresses and transferred to host S.

### The Address Conversion Method

Because of IPv6 anycast's protocol specifications, it cannot identify the address as anycast by itself and for conversion the host connecting to the anycast address should receive at least one packet from the destination host. There are two approaches to convert this address.

***The Probe Packet Method (Client-Initiated)*** — The host sends a probe packet to the anycast address prior to the start of communication, and it can obtain the destination's unicast address from the source address of the reply packet.

***The Piggyback Method (Server-Initiated)*** — The anycast host appends its anycast address to the packet when sending it back to the connecting peer. It can recognize that the packet has been sent from the host associated with the anycast address by checking the information that has been added to the packet.

The probe packet method requires additional network bandwidth to probe packets, which wastes network resources. However, the piggyback method requires applications to be modified so that the anycast address can be piggybacked on the packet. Since we needed to avoid any modifications to applications, we used the packet probing approach to include the unicast address in the AARP.

### Implementation of the AARP

We implemented and tested the AARP [4]. To resolve the anycast address into its corresponding unicast address, we used ICMPv6 ECHO REQUEST/REPLY packets. Since the anycast address should not be set in the source address of the packet header, the anycast membership host sets the corresponding unicast address in the source address field of the ICMP packet instead of the anycast address. Therefore, the host that receives the ICMP ECHO REQUEST

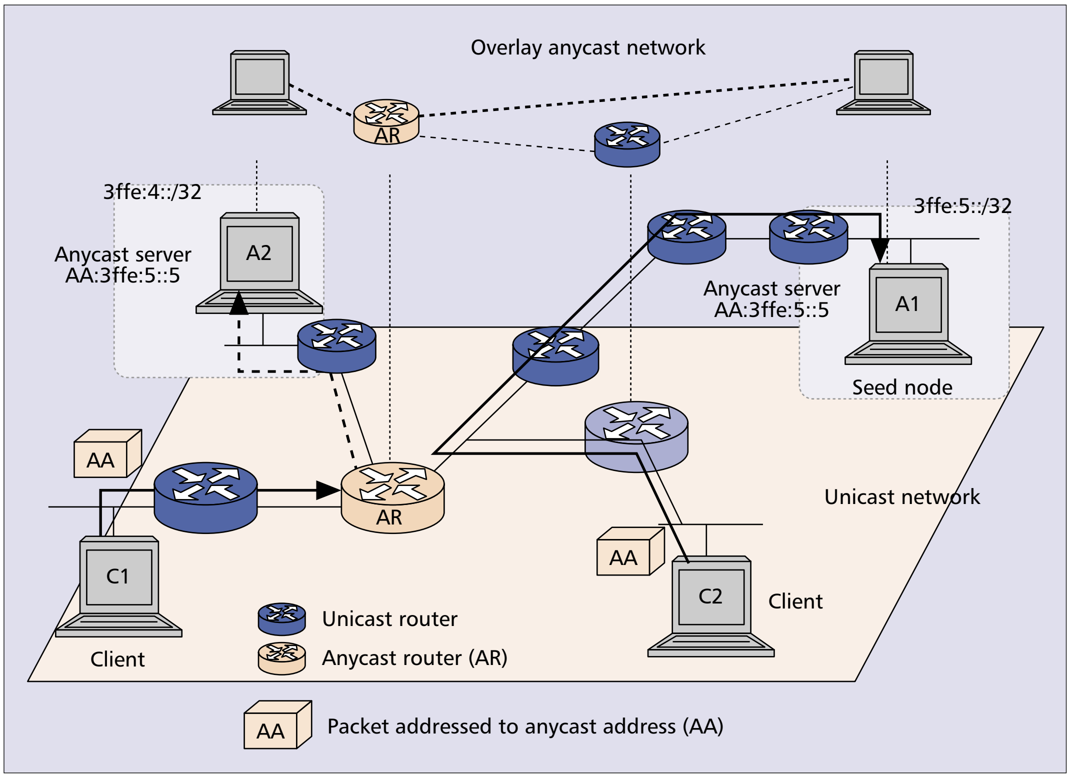


**Figure 3.** *The proposed architecture.*

*We allowed unicast and anycast addresses within the same space and to do this we chose a seed node from anycast membership before assigning an anycast address. We then established the anycast address of membership to be the unicast address of the seed node.*

packet sent to an anycast address will send this packet with its unicast address. If the AARP cannot use the ICMPv6 mechanism, special software is required to respond to the probe packet from the caller host.

Our AARP library also provides a cache table for resolved anycast addresses. When the anycast address is not cached in the table, the AARP sends a probe packet to resolve the anycast address. The resolved unicast address is stored in the cache table with a timer, and will be deleted when the timer expires. That is, for the client, packets to the anycast address are delivered to the same anycast server until the cache expires. Otherwise, the AARP returns the resolved unicast address from the cache table, and a probe packet is not sent until the entry for the anycast address has expired in the cache table. Note here that this simple approach using ICMP packets cannot solve security problems. Even if a malicious user hooks the ICMP ECHO REQUEST packet and sends it, the client uses the source address of this packet.

In tests, we used TCP (`telnet`, `ftp`) and UDP applications (DNS). We monitored packet exchanges by these applications with `tcpdump`. The results revealed that AARP makes anycast communications possible by only specifying the anycast address in all existing applications (e.g., `ftp` *anycast_addr*).

## Design of Inter-Segment Anycast Routing Protocol

### Design Choices

The design choices we made in our anycast routing protocol are as follows:

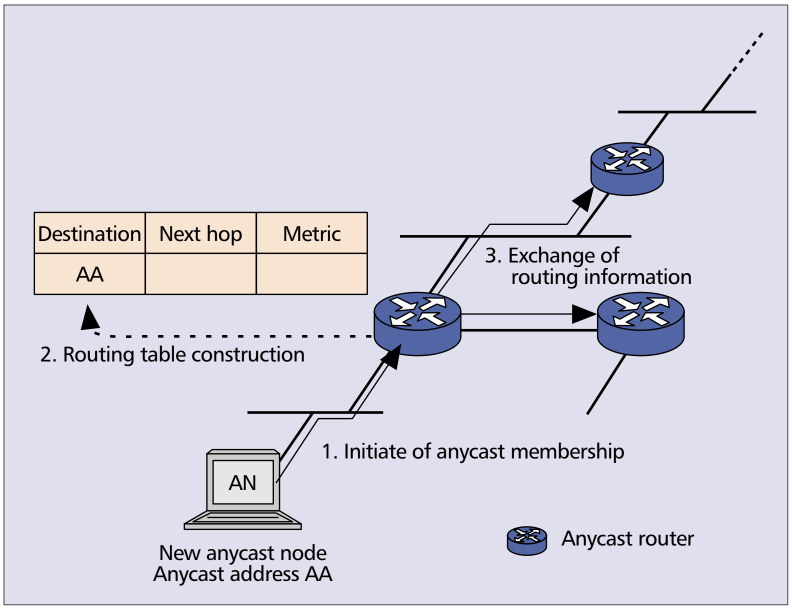
1) We allowed unicast and anycast addresses within the same space; to do this we chose a *seed node* from anycast membership before assigning an anycast address. We then established the anycast address of membership to be the unicast address of the seed node. The anycast router forwards an anycast packet to an appropriate node within the anycast membership. However, the unicast router only tries to forward the anycast packet to the seed node. An anycast packet leaving an arbitrary node is at the very least sent to the seed node. Any packet destined for the anycast address is guaranteed to be sent to at least one destination node.

2) We envision the gradual deployment of anycasting, and the protocol works correctly in our architecture and offers advantages even if there is only one anycast router between the sender and seed node. Its impact will increase as more anycast routers are deployed.

3) We adopted an approach that modifies multicast routing to the anycast routing protocol to reduce the complexity of implementation, since they have many similarities.

### Proposed Architecture

Figure 3 is an overview of the routing architecture we propose and there are two types of routing topologies. The *unicast network* is the existing network topology where both unicast and anycast packets are forwarded on the basis of a unicast address. In the *anycast network*, anycast-aware routers (called *anycast routers*) are connected to one another, and only anycast packets are forwarded by treating their addresses as anycast addresses. The anycast network can thus be considered a logical overlay network over the unicast network.



Destination
Next hop
Metric
AA
3. Exchange of routing information
2. Routing table construction
1. Initiate of anycast membership
AN
New anycast node
Anycast address AA
Anycast router

■ **Figure 4.** *An overview of the anycast routing protocol.*

In an anycast network, nodes are not physically (i.e., directly) connected, but connected via various kinds of logical peer-to-peer connections (e.g., virtual path, tunneling, or encapsulation). An anycast router is upper-compatible, does anycast routing functions, and has the capabilities of unicast routers. An anycast router has an extra routing table (called an *anycast routing table*) to handle anycast addresses. An anycast routing table consists of at least (*anycast address* and next *anycast router's address*) pairs. When a packet arrives at the anycast router, it first checks the anycast routing table to find an entry regarding the destination address of the packet. If it finds this, the packet is treated as an *anycast packet* and forwarded to the next anycast router according to the anycast routing table. Otherwise, it is forwarded through the unicast routing mechanism.

Figure 3 has an example of anycast routing where we have assumed that the node selection criterion is the number of hops. A smaller count is more appropriate here. In Fig. 3 short cylinders represent routers, and the one labeled AR is an anycast router. The other (i.e., unlabeled) short cylinders are unicast routers. There are two anycast members for the anycast address `3ffe:5::5`. Note here that `3ffe:5::5` is also the unicast address of anycast server A1. Here, node A1 is the seed node of anycast membership for `3ffe:5::5`. The other node, A2, is in a different network (`3ffe:4::/32`). Let us now consider where two nodes (C1 and C2) send packets destined for anycast addresses `3ffe:5::5`. The difference is whether there is an anycast router on the route to seed node A1. C1 first forwards the packet to router AR through unicast routing (solid arrow). Intermediate router AR is an anycast router and can detect that the packet is also an anycast packet.

According to anycast routing (dashed arrow), AR then forwards it to node A2, which is the node nearest C1. However, since there is no AR between C2 and A1, the packet is simply forwarded to A1 through unicast routing only. Note that there is a more appropriate node (A2) in this network. For example, if we replace the router next to C2 (light blue cylinder) with an AR, the packet could be transmitted to the more appropriate A2 node through anycast routing.

The above description reveals that our anycast routing protocol works appropriately even when there are a limited number of ARs. If these are increased, better routing is achieved. When all routers in the network are anycast, flexible routing adopting a control policy using various metrics will be possible.

We divided the anycast routing protocol into the following two processes to define it:
- Initiate Anycast Membership: The anycast router collects information on nodes that intend to join anycast memberships.
- Construct and Update Routing Table: According to information collected, anycast routers construct their own routing tables and then exchange routing information with one another to reconfigure these.

Figure 4 has an overview of our anycast routing protocol.

Note again that our basic motivation in supporting anycasting was to minimize overheads or implementation for deployment as much as possible. We therefore focused on the difference between anycasting and multicasting to develop an anycast routing protocol through multicast routing protocols. Anycasting and multicasting have many similar characteristics as well as some differences. Our first step in designing the anycast routing protocol is to clarify the similarities between anycasting and multicasting, and then show how to modify the existing multicast routing protocols to support anycast routing. For this article we chose three multicast routing protocols that are currently available and widely used in IPv4 networks:
- The Distance Vector Multicast Routing Protocol (DVMRP) [5]
- The multicast extension of Open Shortest Path First (MOSPF) [6]
- Protocol Independent Multicast-Sparse Mode (PIM-SM) [7]

Since each multicast protocol has both advantages and disadvantages, we defined the anycast routing protocol based on all of these:
- The Distance Vector *Anycast* Routing Protocol (DVARP)
- The *anycast* extension of OSPF (AOSPF)
- Protocol Independent *Anycast* Sparse Mode (PIA-SM)

These will be presented in turn in the subsections that follow. When modifying these protocols, we considered the following differences between anycasting and multicasting (Table 1):
- Communication form: In anycasting, only one member communicates with the originator. In multicasting, however, all members are equally treated in the routing table. A packet destined for an anycast address will be delivered to only one of the hosts with that address.
- Roles in the client/server model: An anycast address is assigned to the server in anycasting. In multicasting, however, a multicast

address is assigned to the client (i.e., multicast listener). Therefore, multicast membership may change frequently. If there are design points based on this feature in multicast routing protocols, we should modify these points.

Note that comparisons of these anycast routing protocols will be presented later along with some guidelines used in choosing the protocol.

### INITIATE ANYCAST MEMBERSHIP

Like multicasting, the host participating in (or leaving from) anycast membership must have the capability to notify the nearest anycast router of its status (joining/leaving). The method of finding a host participating in anycast membership (called *anycast host* below) is different and is based on the location of the anycast host. If the anycast host and anycast router are on the same segment, an extended version of Multicast Listener Discovery (MLD) is used [8]. We call this Anycast Receiver Discovery (ARD). An anycast host generates an MLD report message to the anycast router before joining the anycast. However, the anycast host sends an MLD leave message prior to leaving membership. Because the destination address field of MLD packets is set to the link-local address of routers (`FF02::2`), this method can only be applied where all hosts and routers reside within the same segment.

All edge routers must become anycast routers with the capability of ARD to enable intersegment anycast routing. However, this is unrealistic in the early stages of anycasting. One possible solution is to locate the authorization node on the same segment as the anycast router. A node with the capability to forward an anycast packet establishes a tunneling path to the authorization node. After establishing the tunneling path, the authorization node advertises anycast address information to the anycast router through ARD. The tunneling path is only used to announce anycast routing information.

Again, note that the method by which anycast hosts are collected sometimes leads to serious security problems. The anycast router should have some mechanism that prevents illegal and/or spoofed anycast host notifications.

### CONSTRUCTING AND UPDATING ROUTING TABLE

***DVARP*** — Since multicast membership is expected to change dynamically in DVMRP, it is hard to specify the route multicast packets will traverse before beginning transmission. Therefore, a flooding (or broadcasting) approach is effective. However, anycast membership does not change as frequently as that in multicasting, and its routing information is more stable. Therefore, DVARP does not use flooding but exchanges routing information periodically.

Figure 5a has an example of updating a DVARP routing table. DVARP operation is done as follows:

- If the anycast router detects changes in anycast membership, the anycast router updates/creates the routing entry in its own routing table.
- Each DVARP router periodically sends its own routing information to its adjacent routers.
- If a router receives routing information from adjacent routers, it updates entries in the routing table.

***AOSPF*** — Unlike DVMRP, routing information is not flooded when multicast packets in MOSPF first arrive. Instead, the router exchanges routing information with other routers when multicast membership changes. This approach suits anycast routing because its membership is more stable than multicast's. Therefore, AOSPF also adopts this membership change-driven approach, and the AOSPF routing table is exactly the same as DVARP's. AOSPF operation is as follows:

1) If the anycast router detects a change in anycast membership, it updates/creates the entry in its own link-state database.

2) When detecting membership changes, each AOSPF router immediately sends the link state update to adjacent AOSPF routers.

3) After updating/creating its own link state database, the router uses Dijkstra's Shortest Path First (SPF) algorithm and calculates the shortest path tree from the router. Then the anycast router creates/updates its routing table from the shortest path tree.

The AOSPF operation is similar to that of DVARP's except that it uses Dijkstra's SPF algorithm, and there is a difference in the frequency of routing information exchanges. DVARP periodically exchanges information, while AOSPF does this at topology change events, which greatly affects the convergence time for the routing table. When there is a change in route in AOSPF, it is transmitted faster than it is with DVARP.

***PIA-SM*** — PIA-SM uses a core-based-tree algorithm like PIM-SM, [7] and membership management is done by the *core* router. We called this core node the *rendezvous point* (RP) following PIM-SM. The RP is selected from all PIA-SM routers and has is responsible for managing anycast memberships. A packet toward an anycast address is transmitted once to the RP. After transfer to the RP, the packet can be forwarded to the appropriate anycast receiver by the RP. The registration information on the RP is equivalent to DVARP and AOSPF routing tables.

Because this core-based approach can be applied to PIA-SM, the PIM-SM mechanism can directly be used for it. Figure 5b has an example of a new registration to the PIA-SM's RP. We will now describe the operations of RP and PIA-SM routers.

1) If an anycast router detects changes in anycast membership, the PIA-SM router reports these to the RP, which detects two types of message packets, PIA-Join and PIA-Prune. The PIA-Join message indicates that a new anycast receiver has joined membership, and the PIA-Prune message indicates a node no longer belongs to it.

2) If the PIA-SM router (not RP) receives a

*All edge routers must become anycast routers with the capability of ARD to enable inter-segment anycast routing. However, this is unrealistic in the early stages of anycasting. One possible solution is to locate the authorization node on the same segment as the anycast router.*

*A packet toward an anycast address is transmitted once to the RP. After transfer to the RP, the packet can be forwarded to the appropriate anycast receiver by the RP. The registration information on the RP is equivalent to DVARP and AOSPF routing tables.*

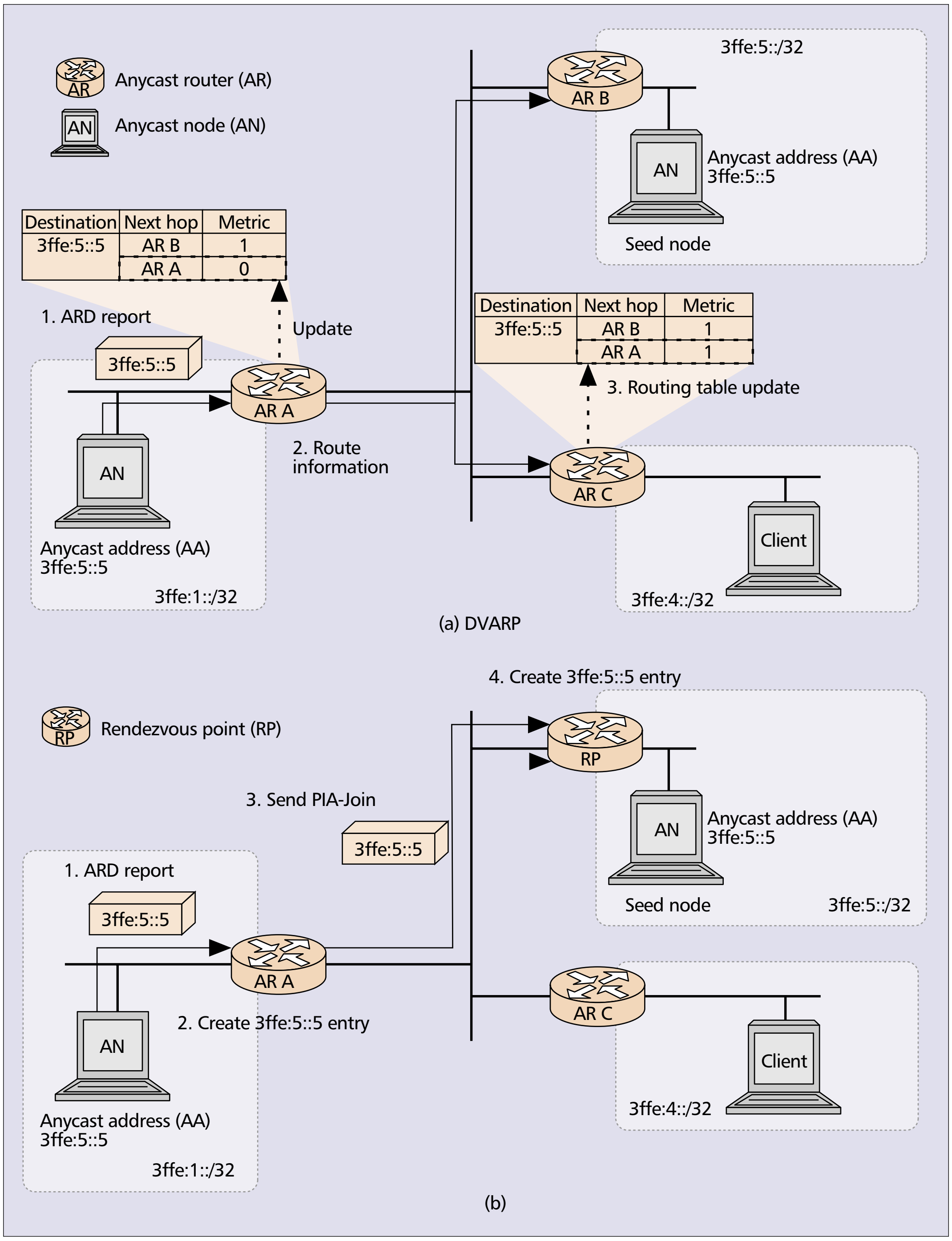


**Figure 5.** *DVARP and PIA-SM: a) updating a DVARP routing table; b) a new registration to a PIA-SM RP.*

PIA-Join (PIA-Prune) message, it creates (cuts) the corresponding anycast membership and sends the PIA-Join (PIA-Prune) to the upper PIA-SM routers toward the RP. If the PIA-SM already has a corresponding entry and the downstream PIA-SM indicates a different router, the next hop is added to an existing entry for multipath routing.

3) Similarly, if the RP receives a PIA-Join (PIA-Prune), it creates (cuts) the corresponding anycast membership. If the RP already has a corresponding entry indicating the downstream PIA-SM router is different, the next hop is added to the existing entry for multipath routing.

## Comparisons of Anycast Routing Protocols

Let us now compare our proposed protocols: DVARP, AOSPF, and PIA-SM. We had the following three objectives in mind for our comparisons:

- Protocol overheads (e.g., CPU load and memory consumption)
- Convergence time due to membership changes
- Ease of implementing protocols

Table 2 summarizes the comparisons. Note that these are similar to what we obtained for multicast routing protocols. With respect to protocol overheads, both DVARP and AOSPF consume a vast amount of network resources, and their traffic consumption is almost linear for the number of anycast groups and number of nodes sharing the same anycast address. Therefore, these protocols can only be applied to small networks with high levels of available bandwidth.

PIA-SM has hardly any traffic consumption, however, because only the RP has routing information; the other PIA-SM routers do not. Therefore, PIA-SM is more scalable than the other two protocols. However, PIA-SM has one problem in that anycast packets are not transferred through the optimal path because they are always transferred through the RP. Another problem is that traffic concentrates around the RP. These problems cause extra delays in packet transmissions. Because of this, PIA-SM can be applied to large networks like the Internet.

DVARP takes a long time for routes to converge; AOSPF takes less. Since all routing information is only kept by the RP in PIA-SM, it is not necessary to exchange routing information.

The implementation of PIM-SM for IPv6 is already available; DVMRP and MOSPF are not as far as we know. Thus, PIA-SM is easier to implement than the other two.

| | | DVARP | AOSPF | PIA-SM |
|---|---|---|---|---|
| Overhead | Network | $O(gm)$ | $O(gm)$ | RP:$O(ng)$ |
| | Router | $O(gs)$ | $O(gs) + O(l*\log(gm))$ | RP:$O(gs)$ |
| Convergence | | Hop by hop | Hop by hop | None |
| Implementability | | Not available | Not available | Available |

$n$: Total number of nodes in the network, $g$: number of anycast groups, $m$: number of nodes that share the same anycast address, $s$: number of anycast routing entries, $l$: the total number of links.

**Table 2.** *A comparison of three anycast routing protocols.*

## Conclusion and Future Work

IPv6 anycasting has several problems in facilitating communications with existing applications. To solve these, we propose a new protocol, AARP, which changes an anycast address into a corresponding unicast address, which is used in actual communication after conversion. We demonstrate that communications with the anycast address can occur without changing the existing application program with this protocol.

We also propose and design three anycast routing protocols by focusing on and comparing similarities between anycasting and multicasting, and modifying the existing multicast routing protocol. In this article, however, we only discuss the design of one anycast routing protocol, although we are currently implementing others and evaluating their feasibility.

## References

[1] C. Partridge, T. Mendez, and W. Milliken, "Host Anycasting Service," RFC 1546, Nov. 1993

[2] S. Weber and L. Cheng, "A Survey of Anycast in IPv6 Networks," *IEEE Commun. Mag.*, vol. 42, no. 1, Jan. 2004.

[3] M. Oe and S. Yamaguchi, "Implementation and Evaluation of IPv6 Anycast," *Proc. 10th Annual Internet Soc. Conf.*, 2000.

[4] S. Ata, H. Kitamura, and M. Murata, "A Protocol for Anycast Address Resolving," Internet draft, draft-ata-ipv6-anycast-resolving-01.txt, Feb. 2004.

[5] D. Waitzman, C, Partridge and S. Deering, "Distance Vector Multicast Routing Protocol," RFC1075, Nov. 1988.

[6] J. Moy, "MOSPF: Analysis and Experience," RFC 1584, Mar. 1994.

[7] S. Deering *et al.*, "The PIM Architecture for Wide-Area Multicast Routing," *Proc. IEEE/ACM Trans. Net.*, vol. 4, Apr. 1996, pp. 153–62.

[8] B. Haberman and D. Thaler, "Host-based Anycast using MLD," Internet draft, draft-haberman-ipngwg-host-anycast-01.txt, May 2002 (expired Nov. 2002).

## Biographies

Satoshi Doi (s-doi@anarg.jp) is currently with Sony Corporation, Japan. He received B.E. and M.E. degrees in information science and technology from Osaka University in 2002 and 2004, respectively. His research work is in the area of designing protocols related to IPv6 anycasting.

Shingo Ata [M] (ata@info.eng.osaka-cu.ac.jp) is a lecturer in the graduate school of engineering at Osaka City University, Japan. He received M.E. and Ph.D. degrees in informatics and mathematical science from Osaka University in 1998 and 2000, respectively. His research includes design of communication protocols and performance modeling of communication networks. He is also a member of IEICE and ACM.

Hiroshi Kitamura [M] (kitamura@da.jp.nec.com) works at NEC Corporation, Japan, since 1990. He received B.S. and M.S. degree from Nagoya University in 1988 and 1990, respectively. He also received a Ph.D. degree in informatics and mathematical science form Osaka University in 2002. He has been engaged in research and development of Internet protocols. He currently focuses on research and development of IPv6, Mobile IPv6, and Plug and Play. He is also a member of IEICE.

Masayuki Murata [M] (murata@cmc.osaka-u.ac.jp) is a professor in the graduate school of information science and technology at Osaka University, Japan. He received M.E. and D.E. degrees in information and computer sciences from Osaka University, Japan, in 1984 and 1988, respectively. In April 1984 he joined Tokyo Research Laboratory, IBM Japan, as a researcher. In 1987 he moved to Osaka University. He has more than 300 papers in international and domestic journals and conferences. His research interests include computer communication network architecture, and performance modeling and evaluation. He is also a member of ACM, The Internet Society, IEICE, and IPSJ.