6-2000

Multicast Internet Protocol

X. K. WANG
Kent Ridge Digital Labs

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Feng BAO
Singapore Management University, fbao@smu.edu.sg

DOI: https://doi.org/10.1016/s0140-3664(00)00167-5

Citation

WANG, X. K.; DENG, Robert H.; and BAO, Feng. Multicast Internet Protocol. (2000). *Computer Communications*. 23, (11), 1047-1054. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/53

Multicast Internet protocol

X.-K. Wang*, R.H. Deng, F. Bao

Ubiquity ab, Kent Ridge Digital Labs., 21 Heng Mui Keng Terrace, Singapore 119613

Abstract

In this paper, we first review the existing IPv4 based multicast protocols and identify their shortcomings. We then proposed a new multicast protocol, called Multicast Internet Protocol (MIP), which is both scalable and flexible. The design principle of MIP is fundamentally different from the existing IPv4 based multicast protocols. The issues related to MIP routing and implementations are also studied in this paper.

Keywords: Multicast; Routing protocol; MIP; Flexibility

1. Introduction

Multicast technology addresses the need of communication among a group of users or simultaneous dissemination of information to various destinations. The continued growth of Internet applications demanding multicast communications has led to the proposal of many multicast protocols. In principle, unicast technology is sufficient for all the communication requirements on the Internet, including multicast. The important problem is how to do the work with minimum cost and maximum efficiency. The heterogeneous nature of the data link layer protocols used in the Internet as well as the complex and dynamic topology of the Internet demands multicast protocols to: (1) overcome the difference of lower-level details and provide an unified interface to higher-level applications; (2) provide the mechanisms for seamless packet forwarding between subnets; and (3) provide the ability of managing both large and small multicast groups efficiently.

In this paper, we first review the existing IPv4 based multicast technologies and protocols over the Internet. We then identified their problems and propose a new multicast protocol, called Multicast Internet Protocol (MIP), which is both scalable and flexible. The design principle of MIP is fundamentally different from the existing IPv4 based multicast protocols. We also study issues related to MIP routing and implementations.

* Corresponding author.
E-mail addresses: xkwang@krdl.org.sg (X.K. Wang), deng@krdl.org.sg (R.H. Deng); baofeng@krdl.org.sg (F. Bao).

2. Review of multicast technology

Internet Protocol (IP) has been proven to be a successful network layer protocol for unicast communications over the Internet. Most of the existing multicast protocols were designed to be built on top of IP. We call these protocols IPv4 based multicast protocols. In this section, we introduce the important terminologies, make a brief review on IPv4 based multicast protocols and along the way, identify their limitations.

2.1. Address space

IP addresses are divided into several classes. Class D IP addresses (with the first 4 bits being 1110) are defined as multicast IP addresses. Among them, the base address 224.0.0.0 is reserved and cannot be assigned to any group. Addresses 224.0.0.1 to 224.0.0.255 are reserved for the use of routing protocols and other low-level topology discovery or maintenance protocols. Addresses 239.0.0.0 to 239.255.255.255 are reserved as “administratively scoped IPv4 multicast space” [10] for local usage; no router will forward IP packets in this address space.

Other well-known multicast addresses are: “all systems on this subnet” (224.0.0.1), “all routers on the subnet” (224.0.0.2), “all DVMRP routers” (224.0.0.4), “all OSPF routers” (224.0.0.5), etc. MBone was registered to use the address space from 224.2.0.0 to 224.2.255.255.

2.2. Multicast groups

Almost all the IPv4 based multicast protocols use the so called “Host Group Model” [7] as their multicast service

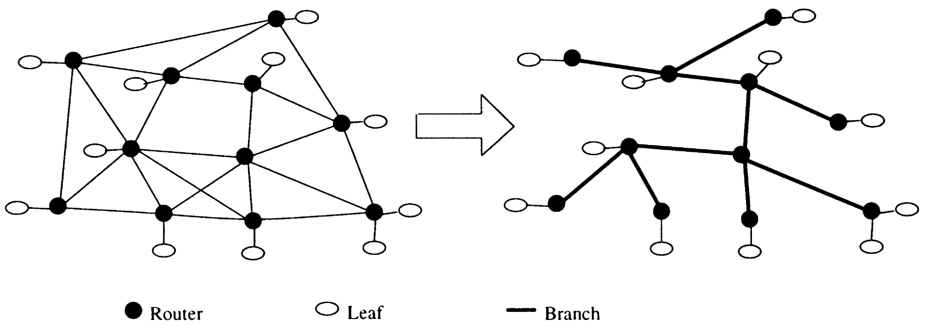

Fig. 1. Spanning tree.

model. Under this model, the set of destinations of multicast packets is called a *host group*, and it is identified by a single *group address* or *multicast address*. To accomplish a multicast, a sender simply places a group address, rather than an individual unicast address, in the destination address field of a packet. When a host sends datagrams to a multicast address, all the group members can receive the datagrams. The datagrams will be forwarded by multicast routers if the sender and the group members are not located in the same subnet.

One of the problems associated with this approach is address space conflict since it is difficult to allocate a unique multicast IP address to all the hosts who want to transmit something to others. There is always the possibility that different host groups choose the same IP address as their multicast address. When this happens, the multicast address alone is insufficient to distinguish different multicast sessions. This not only results in address conflict, but also results in bandwidth wastage; if a multicast router forwards all the datagrams addressed to the multicast address to a subnet, some of the datagrams may be useless for the hosts in the subnet because of address conflict. So in the new proposal (IGMPv3 [4]) group members can tell the routers they want to receive datagrams from which source host. This implies that every multicast session is specified by a (source, group) pair where group refers to a specific multicast IP address.

2.3. Multicast routing protocols

2.3.1. Flooding

The simplest technique for delivering multicast datagrams to other subnets on the Internet is flooding. That is, when a multicast router sees a packet with a multicast address, the router checks whether this is the first time that it sees the packet. If it is, the router forwards the packet to all network interfaces except the one on which the packet arrived; otherwise, the packet is simply discarded. Flooding generates a large number of duplicate packets and uses all available paths across the Internet instead of a limited number of paths. It also eats a lot of resources since each router has to maintain a table of recently arrived packets.

2.3.2. Spanning tree

A more effective solution than flooding is to select a subset of the Internet topology to form a spanning tree (see Fig. 1). All the routers only forward packets along the branches of the tree. The loop-less topology of the tree guarantees that a subnet will not receive a packet twice as in flooding technique.

The spanning tree forwarding solution is powerful and is relatively easy to implement. However, it centralizes traffic on a small numbers of links and it may not provide the most efficient path between the source subnet and group members. Another disadvantage is that it broadcasts datagrams to all the subnets even if there are only a few nearby subnets who really need them.

2.3.3. Distance vector multicast routing protocol (DVMRP)

DVMRP [16,18] is an Internet routing protocol for datagram delivery to a group of hosts across an internetwork. It employs the Reverse Path Multicasting (RPM) [5] technique to generate IP Multicast delivery trees dynamically.

In DVMRP/RPM, the first packet is forwarded to all routers along a source-specific spanning tree. Then based on the IGMP information that leaf routers get, some useless leaves or branches are “pruned” from the packet forwarding tree. The “prune” messages are sent along the reverse path of the delivery tree (see Fig. 2). The pruned state of the multicast forwarding tree is be refreshed regularly to reflect the dynamic changes of both the group membership and the network topology. Periodically the prune information is removed from the memory of all routers and the next packet

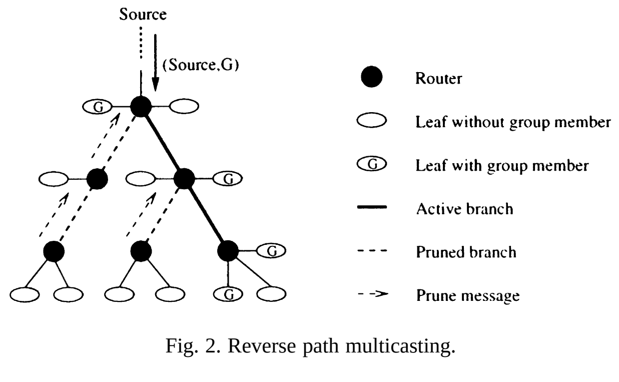

Fig. 2. Reverse path multicasting.

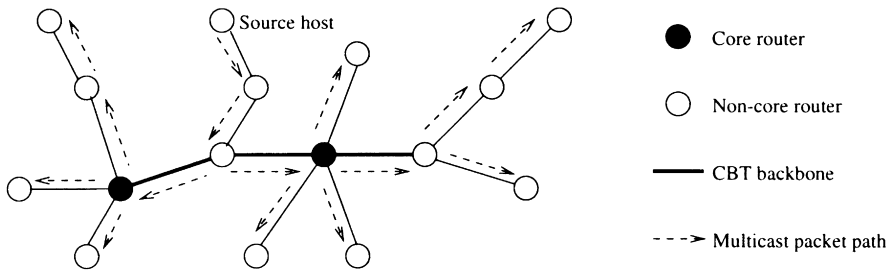

Fig. 3. Core-based trees.

containing the (source, group) pair is forwarded to all the leaf routers again. Sometimes a “graft” message is used to graft a pruned branch. Hosts can leave or join a group any time after the multicast session has been established.

The first drawback of RPM is that multicast packets must be periodically forwarded to every router in the Internetwork. The second drawback is that each router is required to maintain the state information of all (source, group) pairs. This drawback is amplified when the number of (source, group) pairs getting large.

2.3.4. Core-based trees (CBTs)

Core-based trees (CBT) [1–3] are another set of multicast forwarding algorithms/protocols. Unlike previous algorithms which build a source-rooted, shortest-path tree for each (source, group) pair, CBTs constructs a single delivery tree for all the sources in a group (see Fig. 3). It is quite similar to the spanning tree algorithm except it allows each group to use their own core-based tree. CBT has a single router or a set of routers, which act as the core of a multicast delivery tree. Every source has to transmit their datagrams to the core, then the core will deliver the packets to the places where they are needed. CBT has a number of advantageous features comparing with RPM. Firstly, CBT core router is only required to maintain state information for each group, not for each (source, group) pair. Secondly, CBT makes more efficient use of network bandwidth, for it does not require the multicast packets be periodically forwarded to all routers in the Internet.

Despite these benefits, CBTs still have their limitations. They may result in traffic concentration and bottlenecks near the core routers since traffic from all sources traverses the same set of links as it approaches the core.

3. Multicast Internet protocol

In this section, we present a new multicast protocol called Multicast Internet Protocol (MIP). We first state our protocol design objectives, followed by the detailed protocol description.

As described in Ref. [11], there are three group communication models: (a) the collaborative model (N to N); (b) the dissemination model (1 to N); and (c) the gather model (N to 1). Our protocol does not intend to solve the problem of the “gather model”, since we cannot see any relationship between gather model and multi*cast*. Natively, MIP is a multicast protocol for the dissemination model (1 to N). A set of MIP multicast sessions together can form a (N to M) conference.

3.1. Design goals

Maximum scalability: Scalability of a multicast protocol is concerned with the ability of the protocol to function properly as the multicast group sizes increases. A good multicast protocol over the Internet should operate efficiently for both small area as well as the Internet wide applications. For example, multicast sessions over a subnet or a few adjacent subnets should be confined within the appropriate areas; if, however, the underlying multicast protocol needs first to forward datagrams to a “core” router far away, it is obvious a waste of bandwidth.

Minimal router overhead: It will be an impossible task for all the routers to maintain information for all the multicast sessions in the Internet. In an ideal MIP, if the transmission is within the same subnetwork or among subnetworks nearby, routers and subnets far away need not know about the transmission. A router is involved in the transmission if and only if it is on the forwarding path and it has duty to forward transmissions. Of course, if a host wants to join a multicast session that the adjacent routers are not aware of, the routers must have some ways to look for the multicast session and forward the datagrams to the subnets where the datagrams are needed.

No redundant forwarding: The problem of redundant forwarding has been solved partially in some of the existing proposals albeit at a relatively expensive price. One example is RPM, in which it is required to forward the first datagram to all the routers on the Internet. This problem can be solved completely by using receiver-initiated membership service model.

Avoiding address conflict: To avoid the problem of address conflict in multicast, existing protocols use the (source, group) pair to distinguish a multicast session from each another. This solution is cumbersome, to say the least. We solve this problem by taking a completely approach in our proposal.

Compatibility with existing solutions: A new protocol must have some compatibility with the existing protocols in order to have user acceptance and ensure smooth transition.

3.2. Multicast session

A multicast session refers to a multicast data stream from a sender to a group of receivers. Such a session makes a life circle. When a multicast session is set up, some network resources are allocated to it; when a session is killed, all the resources allocated to it will be released. A multicast session has fixed sender/source but can have unlimited number of dynamically changing receivers.

One important problem is how to identify a multicast session in order to distinguish one from another. Almost all the existing multicast protocols are based on the IP multicast architecture which uses "Host Group Model" [12] as its service model. Initially, this model was intended to solve the "N to M" multicast problem, but it quickly led to the problem of address conflict. The solution used in existing protocols to solve the address conflict problem is to use a (source, group) pair to identify a multicast session. However, this approach makes the concept of "group" almost useless. In our proposal, we assume that all the sending hosts have a unique IPv4 address which can be used to identify a multicast session. But this is not enough, for one sender host may need to have multiple multicast sessions at the same time. In MIP, we can use a 16-bit (or 32-bit) integer "port" number together with the source's IP address to identify a multicast session. The "port" here is just a software concept used to distinguish multicast sessions set up by the same host. It is quite similar to the "port" in TCP and UDP protocols. Therefore, we use the (source, port) pair to identify a multicast session. Note that this pair is unique throughout the Internet.

3.3. Multicast internet protocol (MIP)

3.3.1. Header format

A possible header format for the proposed MIP is given below, where

0 7	8	15	16 31
8-bit type (version)	header length	reserved	16-bit total length
	source IPv4 address		
	32-bit port number		
16-bit header checksum		16-bit data checksum	
	options (if any)		
	data (if any)		

- *type:* type/version number of the protocol. Currently it is 0x01.
- *header length (4-bit):* the number of 32-bit words in header.
- *reserved:* reserved for future use, must be zero.
- *total length:* length of the MIP header and the MIP data in bytes. The minimum value for this field is 16 bytes.
- *source address:* 32-bit source IPv4 address.
- *port:* 32-bit port number, selected by source host.
- *header checksum:* checksum for the MIP header.
- *data checksum:* checksum for the MIP data. If the calculated checksum is 0, it is stored as all one bits (0xffff). If the transmitted checksum is 0, it indicates that the sender did not compute the checksum.

As shown in Fig. 4, MIP can be implemented directly on low-level protocols such as Ethernet and Token Ring, or on existing IP Multicast protocol, or even on TCP/IP unicast protocol when multicast support is not available from the subnet. Regardless of its implementations, MIP provides a unified interface to the upper-layer applications which use multicast services.

Although MIP may use IP as its lower-layer protocol, it is not designed to use IP routing mechanism for forwarding multicast datagram. MIP routers are different from IP routers. A MIP router may or may not run on the same machine with an IP router. MIP implementation issues will be discussed in Section 4.

3.3.2. MIP routing

MIP uses a receiver-initiated, link-state routing mechanism. As shown in Fig. 5, a sender a1 on subnet A sets up a multicast session (a1, port) with receivers of this session located in the same subnet as a1 (such as a2,a3,…), in the adjacent subnet B (such as b1,b2,…), and in subnets D and F (such as dn,fn,…).

When a receiver wishes to join a multicast session (source, port), it first needs to find out if the source host is in the local subnet. The receiver does this by using the "subnet mask" to do an "and" operation with the source IP address, and compare it with its IP address' "masked" volume. This kind of technique has already been widely used in IP (unicast) routing algorithms. If the sender is in the same subnet as the receiver, then no MIP routing is needed, the receiver can receive datagrams from the sender directly; otherwise, the receiver looks for the MIP routing table for more information about the source host. The "MIP routing table" tells which MIP router should be used to reach the source host. This technique is similar to what is used in IP (unicast) routing algorithms but with a fundamental difference. In MIP the receiver looks for a path towards the sender while in IP unicast routing a datagram from the sender finds its way towards the receiver. If IP routers can function as MIP routers at the same time, the "MIP routing table" and the IP unicast routing table can be merged into one table. After a receiver or a MIP router has decided which MIP router to use to reach the multicast session's sender, it sends a JOIN_REQUEST to the MIP router which will in turn forward datagrams of the multicast session to the receiver.

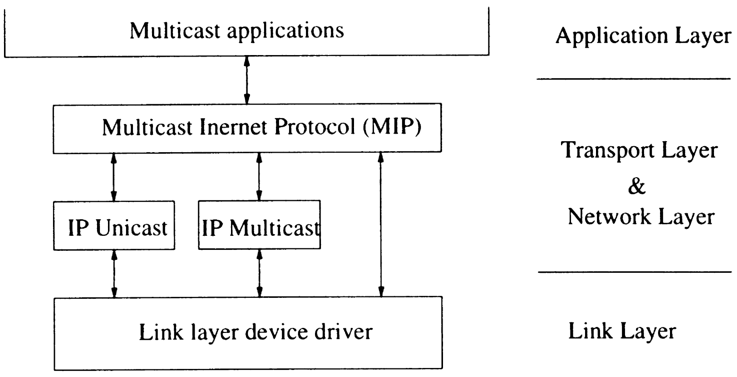

Fig. 4. Multicast Internet protocol in TCP/IP layer model.

3.3.3. JOIN_REQUEST and JOIN_ACK messages

A JOIN_REQUEST can be generated by a host or a leaf router. This join message is sent hop-by-hop towards the source host of the session (the IP address of the source host is known as a part of the session ID). The origination host caches ⟨session,NULL,upstream interface⟩ state for each join process. If there is no response received within [RTX_INTERVAL], the origination host is responsible for retransmissions of this message.

If the interface over which a JOIN_REQUEST is to be sent supports multicast, the JOIN_REQUEST will multicast to all the MIP routers on that link. All the routers on the link may process the join, but only the “next-hop router (NR)” may forward it. Here the NR is selected by the host/router who send/forward the join. The NR’s IP address is included in the JOIN_REQUEST massage. If the next-hop router is not on-tree for the session, and it has not already forwarded a join for the same session, a next NR is selected and the join is forwarded to the next hop on the path towards the source.

If the receiving router/host is the source host or it is a on-tree router, the JOIN_REQUEST is “terminated” and acknowledged by means of a JOIN_ACK. The JOIN_ACK is sent over the same interface as the corresponding JOIN_-REQUEST was received. At the same time, the router will add the interface to its child interface list for the session, if it has not already. A JOIN_ACK is multicast or unicast, according to whether the outgoing interface supports multi-cast transmission or not.

The routers who received the JOIN_ACK message will try to match the message from the cached transient state. If a match is found, a forwarding cache entry for the session is created, and the JOIN_ACK is forwarded to the interface towards the host where the JOIN_REQUEST came from.

3.3.4. QUIT_NOTIFICATION and FLUSH_TREE messages

A MIP forwarding tree is pruned whenever a router’s child interface list for a session becomes “NULL”. A QUIT_NOTIFICATION is sent to the router’s up-stream direction when the branch is going to be pruned. The QUIT_NOTIFICATION is not acknowledged, and the

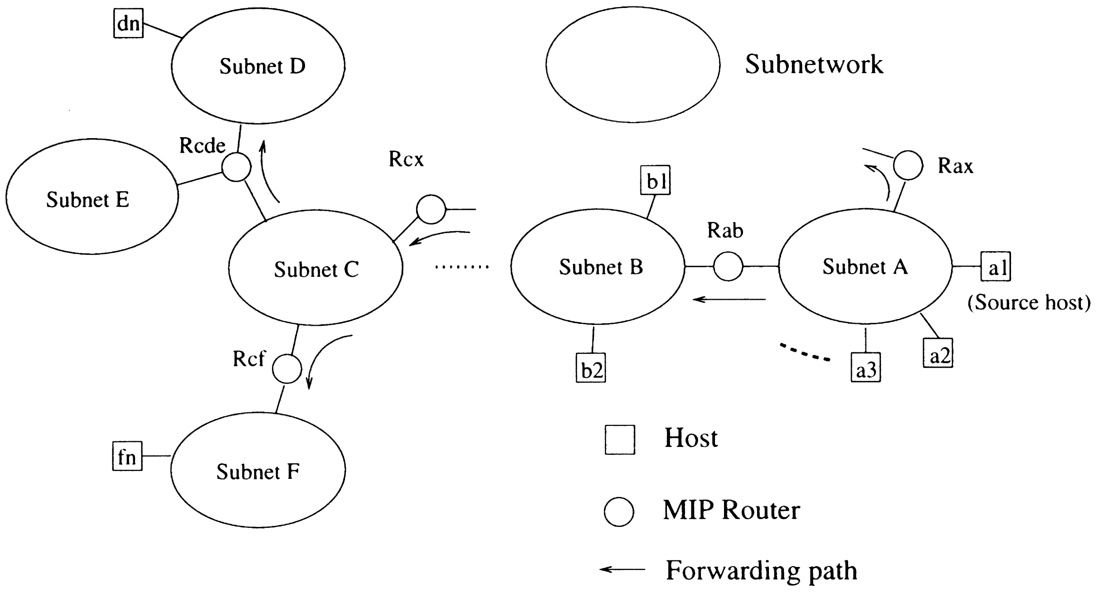

Fig. 5. MIP routing.

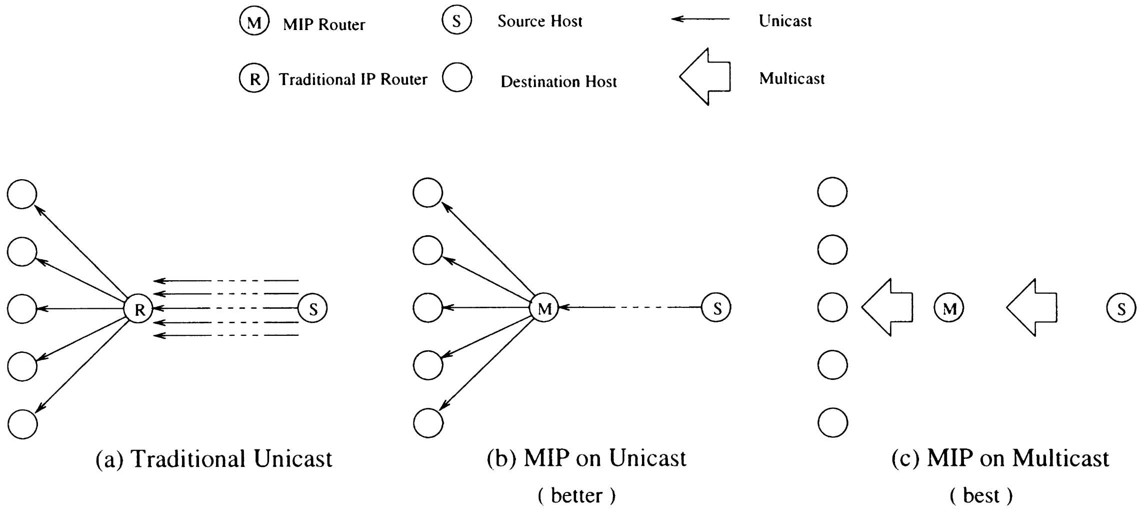

Fig. 6. Comparison between three distribute models.

sending of quit message also invokes the sending of a FLUSH_TREE message over each downstream interface for the session.

If the QUIT_NOTIFICATION is multicast and the arrival interface is a valid child interface, the router sets a cache-deletion-timer [CACHE_DEL_TIMER]. During this interval the hosts/routers who still want the multicast session can send the new JOIN_REQUEST to the router in order to cancel the cache-deletion-timer. Otherwise, the branch will be pruned when cache-deletion-timer expire.

FLUSH_TREE message is sent over each downstream interfaces to notify the prune of this “branch”. If a valid FLUSH_TREE message were received via a upstream interface, it will be forwarded to all the downstream interfaces of the session and all forwarding information for this session will be removed. A flush can specify a single session, or all MIP sessions.

3.3.5. ECHO_REQUEST and ECHO_REPLY messages

The ECHO_REQUEST and ECHO_REPLY massages can be used to let the child to monitor the reachability to its parent router. The ECHO_REQUEST messages are sent at [ECHO_INTERVAL] second intervals. If no response is received from the parent interface, the timer [SESSION_EXPIRE_TIME] will eventually expire. This results in the sending of a QUIT_NOTIFICATION to the upstream, and FLUSH_TREE message to all the downstream interfaces.

3.3.6. Summary

The messages in MIP routing protocol may looks similar to that of in the CBT protocol [2] in some ways. Actually, the messages are similar because their jobs are also similar. But the difference between the CBT and MIP protocols is obvious. Firstly, no core router(s) is needed in MIP. The sender of a session is the “core” for the session. And furthermore, no “HELLO” protocol is needed to elect a designated router (DR). In MIP, the next-hop router (NR) is selected by the sender/forwarder of the JOIN_REQUEST message. Secondly, the service models are different. CBT and all the other multicast routing protocols designed for IP multicast use the “Multicast Group Model”; the MIP does not intend to follow this service model. In MIP protocols, we use “multicast session” instead of “multicast group”. The multicast session is a “light-weighted” version of multicast group, it is simpler and relatively easier to implement. And in MIP protocol, all the forwarding paths are single-directional, which is different from that of CBT and all the other multicast routing protocols based on the “multicast group model”.

4. Implementation of MIP

4.1. MIP in TCP/IP layer model

MIP is a network-layer protocol. Since MIP needs to know the “port number” of a multicast session, there is no need to insert a “transport layer” between it and the application layer. MIP provides multicast services directly to applications. It is expected that reliable multicast be provided in future versions of MIP.

MIP can be implemented on various kinds of lower-layer protocols, including multicast lower-layer protocols (such as Ethernet, Token-Ring, IP-Multicast) as well as unicast lower-layer protocols (such as IP-Unicast, TCP). The latter approach is often useful when the lower-layer protocol does not support multicast services. For example, when MIP datagrams need to go through an area that does not support multicast, a “tunnel” is needed between the two “islands” that support multicast.

4.2. Efficiency considerations

Fig. 6 depicts three types of distribution models. In the models, we assume that there is a set of destination hosts on a subnet and a source host (which may be located far away from the destination subnet) distributes data packets to the destination hosts. If a unicast protocol is used (as in Fig. 6a), an individual connection is required from the source to each destination host. Fig. 6b shows the situation when MIP is implemented on top of a unicast protocol. In this case, only one connection is needed to deliver datagrams from the source to the MIP router, the MIP router then distributes the datagrams to destination hosts individually. Since MIP router is located locally to the destination hosts, so this solution is more efficient than Fig. 6a. Fig. 6c shows the case when MIP is implemented over a multicast protocol where datagrams need to be transmitted only once from the source to a MIP router and then from the MIP router to the destination hosts. This is the most efficient solution among the three approaches.

4.3. Implementing MIP over different protocols

MIP can be implemented over different lower-level protocols. The details of these implementations may vary but the Application Programming Interfaces (APIs) that it provides to upper-layer applications will remain the same.

To implement MIP on Internet Protocol (unicast/multicast) will lead to many advantages. First, TCP/IP can provide an uniformed interface to its upper-layer protocols. This makes the implementation of MIP independent of specific underlying hardware. Once an implementation has been developed, it can be ported easily to other platforms which have TCP/IP support.

Another advantage to use TCP/IP as MIP's under-layer is that a small group of subnets can be combined into a single subnet using the "spanning tree" algorithm. Routers between these subnetworks just forward datagrams "transparently" to other subnetworks along a spanning tree. Then MIP can treat these subnets as a single subnet. No real MIP routers are needed among these subnets. Instead, a simple IP-Multicast router can be used.

Although implementing on IP-Multicast has many advantages, not all the IP routers and hosts can really support IP-Multicast. MIP does not intend to use the routing mechanism of IP protocol, so implement MIP directly over link-layer protocols will be a more efficient solution. If link-layer protocols are used to implement MIP, a new field "fragment" is needed in the MIP header. Because the maximal package length in different link-layer protocols may be different. MIP routers must handle the situations that a MIP package is larger than the physical limitations on a subnetwork.

4.4. Multicast service interface

MIP provides a set of APIs to upper-layer protocols or application programs. Two of the important APIs are shown below:

```
MIPSend (socket, interface, port,
service_type)
MIPListen (socket, interface,
source_address, port)
```

- *socket* is an implementation-specific parameter used to distinguish among different requesting entities (e.g. programs or processes) within the system;
- *interface* is a local identifier of the network interface on which sending/reception of the specified multicast address is to be enabled or disabled;
- *port* is a 32-bit/16-bit unsigned integer. It is selected by session sender;
- *service_type* indicates what kind of services (with/without reliable, with/without security, etc.) is needed by applications;
- *source_address* is a 32-bit source IP address.

Since MIP uses sender's IP address and a "port" number to identify a multicast session, end users may identify a multicast session in the form of "host_name:port_number", such as: "nicpc26.krdl.org.sg:1234". If a default port is used, a MIP multicast session may be described in a "URL" format looks like "MAudio://audio.BBC.com". MIP first uses DNS to resolve the host name into IP address, then uses the default port number as the session's port number.

If a 32-bit port number is used in MIP implementation, all the existing IP-Multicast sessions can be mapped to MIP address space. The mapping algorithm from a (source, group) pair to a MIP 64-bit session ID is quite simple: just use the group address as the 32-bit port number while leaving source address field unchanged. This simple mapping requires that port numbers from 0xe0000000 to 0xeffffffff not be used by other MIP multicast sessions. If an application requires a MIP session with port number falling within the above space, routers should be able to recognize that this is a traditional IP-multicast session. Routers will then transform the IP-multicast packets to MIP packets and forward them to where they are required.

5. Conclusions

MIP is a new multicast protocol based on an entirely new service model. All the design goals described in Section 3.1 are met. We summarize the important features of MIP below:

- *No redundant forwarding:* MIP uses link-based distribution tree schedule, a link will exist if and only if it is needed. Therefore, there will be no redundant forwarding in MIP.
- *High efficiency:* A MIP router knows a multicast session

if and only if it is on the forwarding path of the multicast session. This feature is extremely important for MIP to have good scalability.

- *No centralized distribution point:* No centralized distribution point avoids performance bottlenecks as well as single point of failure.
- *Flexibility in routing algorithm:* MIP is a protocol for multicast and is largely independent of which routing algorithms to use. Selection of routing algorithms depends on the configuration of MIP routers' routing-tables.
- *Independent on lower-level protocols:* MIP can be implemented on various kinds of lower-level protocols, such as IP-multicast, link-layer protocols, or even unicast protocols. This feature is useful because MIP datagrams may need to be forwarded through subnets that does not support multicast.
- *Compatibility with existing architecture:* MIP routers can transform IP-multicast packets to MIP packets and then forward them to destinations. So MIP hosts can handle existing IP-multicast sessions.

MIP is a new multicast protocols which is different from the currently widely deployed IP-multicast architecture. However, as we discussed, MIP can co-exist with the existing protocols: MIP hosts can receive IP-multicast sessions.

Because MIP uses link-based architecture, it establishes independent distribution trees for each multicast session. As a result, many multicast sessions may put a heavy burden on MIP routers. However, MIP is still better than some IP multicast protocols in which every router needs to maintain a database to record all the multicast sessions on the Internet. In MIP, routers only have to know those sessions in which they are on the forwarding paths.

References

[1] A. Ballard, Core Based Trees (CBT) Multicast Routing Architecture, RFC 2201, September 1997.

[2] A. Ballard, Core Based Trees (CBT version 2) Multicast Routing: Protocol Specification, RFC 2189, September 1997.

[3] A. Ballard, Core Based Trees (CBT version 3) Multicast Routing, work in progress, August 1998.

[4] B. Cain, Internet Group Management Protocol, Version 3, IETF Internet Draft, 21 November 1997.

[5] S. Deering, D. Cheriton, Multicast routing in Internetworks and extended, ACM Transactions on Computer Systems 8 (2) (1990) 85–110.

[7] S. Deering, Multicast Routing in a Datagram Internetwork, PhD thesis, ftp://gregorio.stanford.edu/vmtp-ip/sdthesis.part?.ps.Z, December 1991.

[10] D. Meyer, Administratively Scoped IP Multicast, RFC 2365, July 1998.

[11] W. Mostafa, M. Singhal, A taxonomy of multicast protocols for Internet applications, Computer Communications 20 (1998) 1448–1457.

[12] J. Moy, OSPF Version 2, RFC 1247, July 1991.

[16] T. Pusateri, Distance Vector Multicast Routing Protocol, work in progress, August 1998.

[18] D. Waitzman, C. Partridge, S. Deering, Distance Vector Multicast Routing Protocol, RFC 1075, November 1988.

Further reading

S. Deering, Host Extensions for IP Multicasting, RFC 1112, Stanford University, August 1989.

W. Fenner, Internet Group Management Protocol, Version 2, RFC 2236, Xerox PARC, November 1997.

G. Malkin, RIP Version 2, RFC 1388, January 1993.