J. Bion
Cisco Systems
D. Farinacci
Procket Networks
M. Shand
Cisco Systems
A. Tweedly
Cisco Systems

Internet Draft
Document: draft-shand-erm-00.txt
Category: Informational

June 2000

Explicit Route Multicast (ERM)

Status of this Memo

1. Abstract

Conventional multicast builds a tree structure from the source of a multicast stream to the destinations of that stream. Multicast packets are forwarded from the sources down the tree by each intervening router. Each router contains state information to determine the next hop forwarding destination(s) for a packet. Where there are a large number of groups, this gives rise to a scaling problem. The amount of state to be stored in the routers becomes impossibly large.

It is attractive to consider the use of multicast to provide applications such as n-way voice and video conferencing, which would require a very large number (perhaps millions) of relatively small (between 3 and 10 members) multicast groups.

This proposal replaces the need for state in the intervening routers with a description of the delivery tree contained in the packet itself. This allows a multicast service to be deployed for very large numbers of small groups. The trade off is that the packet size is increased proportionate to the size of the delivery tree, which imposes an effective upper limit on the size of the group.

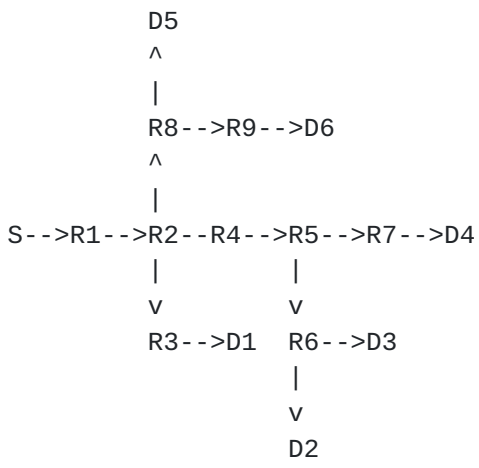
The encapsulation with the delivery tree, and the subsequent decapsulation are performed by the ingress and egress routers. The process is transparent to the hosts, which continue to operate conventional IP multicast data protocols.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [2].

3. Table of Contents

4. Overview



In the diagram above, R1 is the ingress (or encapsulation) router, and R3, R6, R7, R8 and R9 are the egress (or destination) routers. The delivery tree to reach all the destinations (D1 to D6) is indicated by the arrows. When R1 receives a packet addressed to the multicast group from S, it encapsulates the multicast packet in a unicast packet addressed to the first hop router R2 and includes in the packet a header which describes the required delivery tree. Note that the delivery tree need only include the routers that are either branch points in the tree, or delivery points. An intervening router, such as R4 above, need not be included. On receiving such a unicast packet, a router inspects the header to determine the next hop routers and duplicates the packet, adjusting the unicast destination address of each packet to be the next hop IP address.

R1 forwards to R2

R2 forwards to R3, R5, and R8

R5 forwards to R6 and R7

R8 forwards to R9

On reaching a destination router, the packet is decapsulated and the original multicast packet is forwarded to the final multicast destination(s) using normal multicast methods.

The details of the packet encoding and forwarding are described in Section 6

Only the encapsulating router (R1) is required to maintain state concerning the delivery tree. The remaining intervening routers merely forward based on the information in the header. The

information to build the delivery tree is acquired by the encapsulating router, by each destination router sending a 'trace' packet (as discussed in section 7.1) towards the source. At it traverses the network it records the address of each ERM capable router traversed. So in the above example, R1 would receive the following packets.

(D1) R3, R2, R1

(D2, D3) R6, R5, R4, R2, R1

(D4) R7, R5, R4, R2, R1

(D5) R8, R2, R1

(D6) R9, R8, R2, R1

By combining this information (as described in section 6.3), it is a simple matter to build the delivery tree for inclusion in the packet, eliminating any redundant, non-branching nodes (such as R4 above).

In order to send the trace packets, the destination routers must know the source(s) of the group. Techniques to acquire this information are discussed in section 11.

Destination routers handle recovery from failures by re-sending trace packets when no traffic for the group has arrived after some period. In the absence of genuine traffic the encapsulating router sends periodic heartbeat traffic to inhibit the trace packets from still connected nodes. Details of these mechanisms are described in section 6.4.

5. Design Constraints

The following constraints were applied to the design of this protocol: -

1. It must require zero state in the intermediate routers.

   a) State in the 'ingress' and 'egress' routers is permitted.

   b) Solutions that require short duration dynamic per-group state in intermediate routers during tree building may be considered, but solutions that require no state at all are far preferable.

2. It must support multicast group of up to 10 members.

a) While ten group members is a minimum requirement, the actual constraints on a design concern the number of 'egress'

delivery points, each of which may deliver to any number of group members.

3. It must support at least 100,000 simultaneously active groups (source group pairs).

4. It must require NO changes to hosts.

   a) No changes in the operation of the network layer (and below) multicast functions are permitted.

   b) Solutions relying on novel application functions (for example to discover sources) are permitted.

5. Forwarding performance shall not be dramatically worse than conventional multicast.

6. It shall not impact the forwarding performance for other unicast or multicast traffic.

7. Bandwidth utilisation shall not be dramatically worse than conventional multicast.

   a) This means that any encapsulation overhead shall be kept to a minimum.

   b) The delivery tree shall be genuinely multicast. For example, simply unicasting a separate packet from the ingress router directly to each egress router would not be an acceptable general solution (except where the topology so dictates).

   c) The additional 'control' traffic shall be minimal. In particular a good solution should scale to support a large number of 'simultaneous' group instantiations.

8. Delivery to the group should survive topology changes in the network.

   a) Unicast routing changes that leave the optimum delivery tree unchanged should have no effect.

   b) Changes that make the delivery tree sub-optimal may be ignored, but solutions that allow re-optimisation of the delivery tree with minimal control overhead would be preferred.

   c) Changes which would prevent delivery to some or all groups should be recovered from in a timely manner.

9. Leaving group members (egress points) should be removed from the delivery tree in a timely manner.

a) Where groups leave gracefully, the tree should be quickly pruned back to prevent unnecessary delivery.

b) Failed or unreachable egress points should be detected in a timely manner, and pruned from the tree (if recovery is not feasible).

10. Join latency should be comparable to that of conventional multicast.

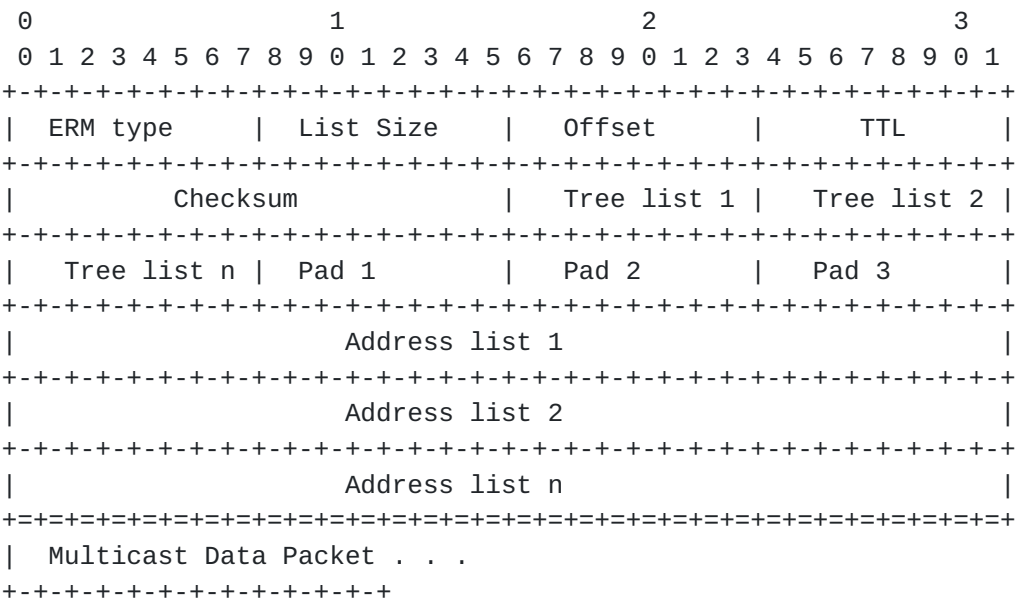
6. Packet encapsulation and forwarding

Multicast packets are encapsulated in a standard unicast packet as follows: -

- Destination IP address = IP address of first hop router
- Source IP address = IP address of encapsulating router
- Protocol = ERM (a new protocol type value TBA)
- TTL = TTL from multicast packet (minus 1).
- ToS = copied from multicast packet
- The data portion of the unicast packet contains an ERM header as described below, followed by the original multicast packet.

6.1 ERM header format

The ERM header contains the following fields



- ERM type -- Type of ERM packet 128 = (encapsulated multicast data). Note that the high order bit is set to indicate that the packet contains an ERM route and should be processed using ERM forwarding.

- List Size -- Number of addresses in the address list (1 byte). The offset to the start of the address list (a) is therefore ceiling((6 + n)/4) 32 bit words and hence the total length of the ERM header (i.e. the offset to the start of the encapsulated multicast data packet) is (a + n) .

- Offset -- The numerical offset of the receiving node's entry in the tree list (1 byte). This is initialised to 0 for delivery to the first hop router, since the first hop router's address does not appear in the list.

- TTL -- Normally indeterminate, but used when forwarding over a layer 2 multicast capable subnetwork. (see Section 8.4.1)

- Checksum -- Ones compliment checksum of ERM header excluding the first four bytes (to avoid the necessity to recalculate when the Offset or TTL fields are modified).

- Tree list -- The list describing the delivery tree (n bytes, where n is the number of entries in the tree list).

- Padding -- To start Address list on 4-byte boundary.

- Address list -- The list of IP addresses for the delivery tree (4n bytes)

6.1.1 Tree list format

The tree list is encoded as a list of parents. Each entry in the list describes the entry number of that entry's parent. At any point in the tree, the set of nodes to which the packet should be replicated on the next hop can be identified as the set of nodes having the present node as their common parent. Thus the example tree above would be represented as

0, 1, 2, 2, 4, 4, 2, 7

where the address list is

R1, R2, R3, R5, R6, R7, R8, R9

An entry of zero indicates that this node's parent is the root of the tree.

This is interpreted as follows

Entry	Parent	Entry Address	Parent Address
1	0	R1	(root)
2	1	R2	R1
3	2	R3	R2
4	2	R5	R2
5	4	R6	R5
6	4	R7	R5
7	2	R8	R2
8	7	R9	R8

In reality, it is not necessary to include information about R1 in the tree, since R1 is the encapsulation node. Similarly it is not necessary to include information about R2 in the tree, since the packet is unicast addressed to that node. Hence the information actually included in the packet would be just.

0, 0, 2, 2, 0, 5

with an address list of

R3, R5, R6, R7, R8, R9

This is interpreted as

Entry	Parent	Entry Address	Parent Address
1	0	R3	(root)
2	0	R5	(root)
3	2	R6	R5
4	2	R7	R5
5	0	R8	(root)
6	5	R9	R8

6.2 Forwarding

On receipt of an ERM encapsulated packet, an ERM router performs the following actions.

1. Checks whether it is a delivery (egress) point for this multicast group, by examining the multicast destination address in the encapsulated multicast packet, and if so takes a decapsulated copy of the multicast packet, updates the TTL of that packet to be the TTL in the received ERM packet -1, and forwards the packet via normal multicast.

2. Determines the next hop forwarding destination if any (see section 6.2.1 below) and forwards a copy of the entire encapsulated packet to each of those destinations with the following changes

- Set the destination IP address in the unicast header to the new destination

- Decrement the TTL in the unicast header

- Update the offset field of the ERM header

- Adjust the unicast header checksum accordingly

  Note that the source IP address in the unicast header is unchanged by the forwarding process. I.e. it remains set as the IP address of the encapsulating source ERM router.

6.2.1 Interpreting the tree list

The children of node n are found by scanning the list looking for the value n. So in the example

0, 0, 2, 2, 0, 5

with an address list of

R3, R5, R6, R7, R8, R9

At the first hop node the offset has the value 0, so we look for that in the list and find entries 1 (corresponding to R3), 2 (corresponding to R5) and 5 (corresponding to R8).

At R3 the offset has the value 1, and since there are no entries with the value 1, we do not forward any further.

At R5 the offset has the value 2, and we find entries 3
(corresponding to node R6) and 4 (corresponding to node R7).

At R6 the offset has the value 3, and there are no entries with value 3.

At R7 the offset has the value 4, and there are no entries with value 4.

At R8 the offset has the value 5, and we find entry 6 (corresponding to node R9).

Note that while a generalized parent tree does not require any particular ordering, the forwarding algorithm can be improved slightly by requiring that it be in preorder form. This allows the search for the offset value to start at entry offset + 1 instead of having to scan the entire list.

6.3 Building the multicast tree

Given a set of 'trace' lists such as those in the example above

(D1) R3, R2, R1

(D2,D3) R6, R5, R4, R2, R1

(D4) R7, R5, R4, R2, R1

(D5) R8, R2, R1

(D6) R9, R8, R2, R1

A parent tree can be constructed by processing each trace list in turn (in the order in which they arrived - see Section 6.3.1 below) and assigning sequential Ids to each unique router address encountered. The parent of each node can then be entered by taking the ID of the next router in the list. We don't need the address of the encapsulating router, so we enter zero for it's ID.

So after processing the first trace we have

2, 0

With an address list of

R3, R2

After the second trace we have

2, 0, 4, 5, 2

With an address list of

R3, R2, R6, R5, R4,

And after all traces have been processed

2, 0, 4, 5, 2, 4, 2, 7

With an address list of

R3, R2, R6, R5, R4, R7, R8, R9

Note that this example is NOT in pre-order form. The actual tree-list in an ERM packet MUST be in pre-order form (see section 6.3.4).

Because trace packets may be processed sequentially, it is a simple matter to accommodate a new receiver, merely by 'adding' its trace packet to the existing tree.

To permit correct identification of non-branching and dead nodes (see below) it is necessary to record which nodes are terminators (i.e. R3, R6, R7, R8 & R9 in the example. In particular R8 is necessary to prevent it from being removed as a non-branching node, and hence failing to deliver packets to D5.)

6.3.1 Changing routes

The algorithm above will always build a tree incorporating the most recent route from the root to any particular node, overriding any previous routes. This seems to be reasonable behaviour given that the most recently received trace packet probably reflects the most recent routing information.

When routes change, it is likely that some portion of the tree will no longer reach any destination. Such 'dead' portions must be pruned off to avoid unnecessary bandwidth wastage. There are two obvious ways to deal with this.

1. Detect that the parent of a node was already set and is being changed to a new value, then follow up the chain of old parents until a node is reached with more than one child (found by scanning the list looking for nodes with parents pointing to this node).

2. Alternatively, the dead branches can be left in place, then pruned by performing a depth first exploration the entire tree from the root looking for nodes that do not lead to a delivery point.

6.3.2 Loops

Partially looping trace packets (as a result of dynamic routing changes) will be dealt with naturally by the above algorithm. When

the trace packet crosses its own path the loop will be removed from the tree just as if it had been a new route.

Clearly, persistently looping trace packets will not arrive at their destination and will be treated as dropped trace packets. It is possible that such a packet may overflow the trace list before the hop count is exhausted. When there is no room in a trace packet to add an ERM entry, the packet should be discarded.

Note that since a Trace Packet has the DF bit set in its outer IP header, a trace packet may also be discarded if it exceeds some link MTU.

6.3.3 Removing non-branching nodes

The tree built by the above algorithm may include non-branching nodes (such as R4 in the example). These can be removed by performing a depth first exploration of the tree from the root and removing nodes that have exactly one child (a node which is also a terminator is never removed). Note that this must be done after any dead branches have been pruned, since removal of dead branches may generate further single child nodes.

It is possible to perform the dead branch removal and non-branching node removal during the same exploration. However, this may not be desirable since a new trace packet can be added to the tree after dead branch removal, but NOT after non-branching node removal (since the new trace path may merge with the old at a node that was previously non-branching).

Performing dead branch removal after each (set of) trace packet(s) may be desirable since it allows the memory used to store the dead nodes to be recovered.

6.3.4 Building the packet headers

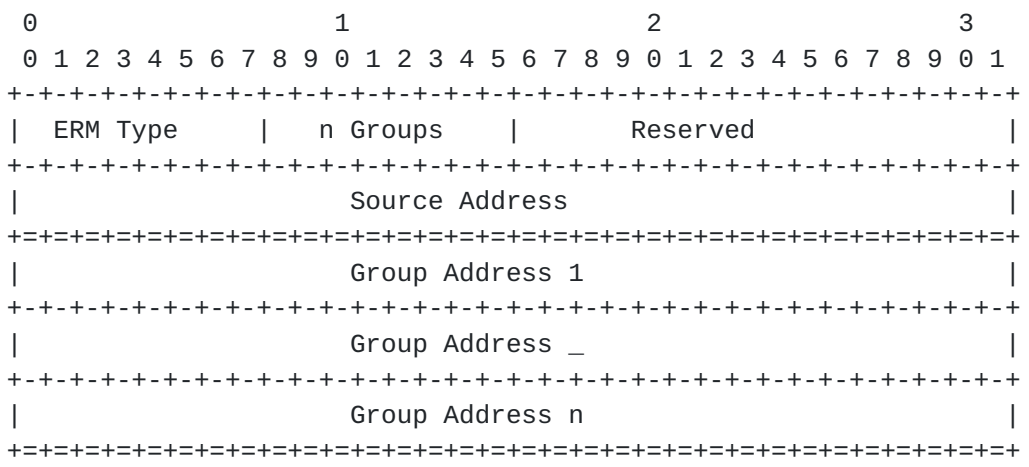
The tree lists in all ERM packet headers MUST be in pre-order form (in order to permit the previously described forwarding optimization). Packet headers in preorder form are easily built from the complete parent tree by performing a depth first exploration and reassigning node IDs. Note that for these purposes the trees are built with the first hop router(s) as the root. If there are multiple first hop routers (i.e. the encapsulating router is a branch point), there will be multiple distinct trees.

6.3.5 Removing a destination

When a destination router detects that it has no more members of the group, it unicasts a 'prune-leave' message directly to the current source ERM router (current_source_ERM_router see section 7.4) and sets current_source_ERM_router to NO_MEMBERS. (The value NO_MEMBERS is returned when there is no record for the group at the destination

router. I.e. there is no need to retain 'negative' state for the group after its members have gone away.) The prune message is

carried in an IP packet with protocol type ERM, and has the following data format.



- ERM Type (1 byte) - Type of ERM packet = 2 (prune-leave) or 3 (prune-change)

- n Groups (1 byte) - Number of Group Addresses

- Source Address - Multicast source IP Address

- Group Addresses (4 * n Groups bytes) - List of Group Addresses to be pruned.

The only information required in the prune message is the [S, G] and the address of the destination router. The latter is obtained from the Source IP address of the IP packet. The encapsulating router can then mark the destination router as no longer being a terminator, and remove the dead branch by either of the techniques outlined above. Note that it is NOT necessary to have access to the original trace packet in order to remove it.

Prunes are not acknowledged. If the prune message is lost, unwanted multicast data may still arrive. The value of NO_MEMBERS in current_source_ERM_router is deemed to match no source ERM router address, and hence results in the (rate limited) re-transmission of prune-leave messages as described in section 7.4.

6.3.5.1. Timer based destination removal

A destination router may die without being able to send a prune-leave message, or it may become partitioned from the rest of the network. In these circumstances, we want the destination to be eventually removed from the delivery tree. This is achieved by the source ERM router maintaining a timer (n*t1) (on the order of a few minutes) with each destination. This is reset by the arrival of a

trace packet from that destination. On expiry of the timer, the destination is removed as if a prune-leave message had been received.

Destination routers maintain a timer with the value t1, and send a trace packet when it expires (irrespective of the arrival of multicast data - unlike the t2 timer described in section 6.4.2.1).

6.3.6 Memory scaling issues in the encapsulating router

As a minimum the encapsulating router needs to store

- The set of all the unique ERM router addresses mentioned in trace packets it receives. Addresses of nodes pruned because they are on dead branches may be safely forgotten, but addresses of non-branching nodes must be retained in case they are subsequently needed.

- For each group (identified by [S, G]), a node list of length N, (where N is the number of unique addresses in the set of trace lists for that group), consisting of offsets into the address list.

- For each group, a parent list of length N.

- Note that there is no need to keep the trace lists themselves.

6.4 Detecting and recovering from failures

6.4.1 Unicast routing topology changes

ERM encapsulated packets are unicast between branch point ERM routers. Changes in unicast topology between ERM routers that do not affect reachability will simply be accommodated by normal unicast routing. ERM encapsulated packets will still be delivered to the next ERM router.

Where the topology changes such that the existing delivery tree is no longer optimum (but is still connected), the old sub-optimal delivery tree will continue to be used until such time as it is re-evaluated as the result of receiving new trace packets. This may occur as a result of new receivers joining the group on destination routers that were not previously receiving the group, or as a result of delivery failure (see section 6.4.2.1). It will also occur periodically at an interval of t1 as described in section 6.3.5.1. Hence, the maximum time for which a non-optimal delivery topology will persist is t1, and it will usually be much less, especially in the part of the tree near the root, where multiple traces contribute towards the topology discovery.

6.4.2 Intermediate ERM router failures

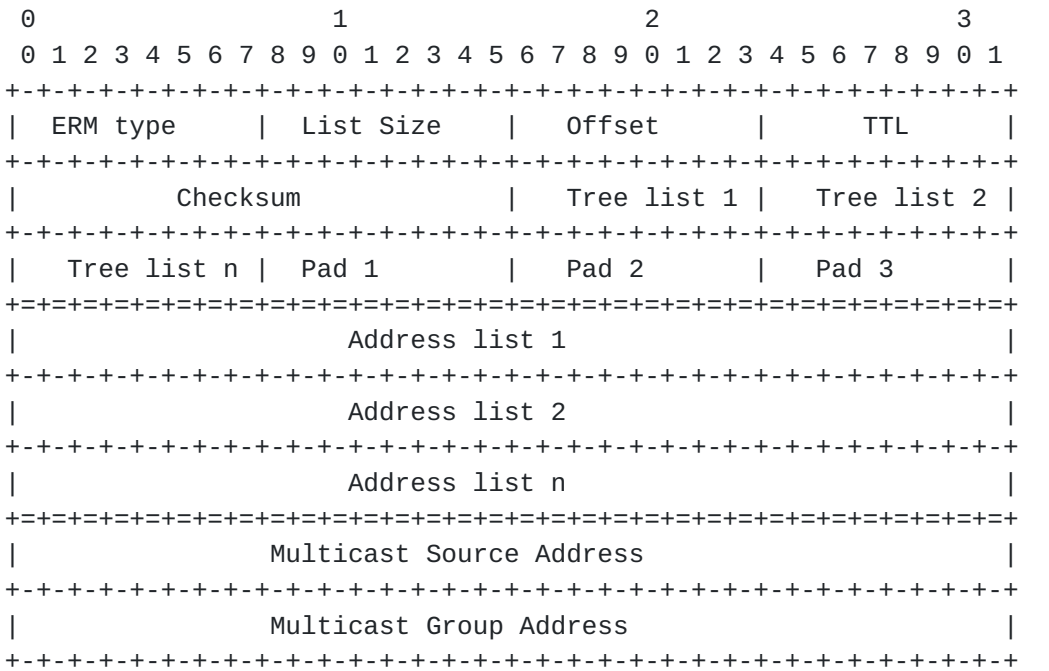
6.4.2.1. Router Failures

Failure of an intermediate ERM router on the delivery tree will cause all destinations below it to stop receiving data. Each

destination router runs a timer (n*t2, where t2 is the expected maximum interval between data packets, and n is the number of lost packets which can be tolerated before recovery is initiated.) which is reset by the receipt of data for the corresponding [S, G]. On expiry of the timer, a new trace packet is sent towards the source, which will discover a new delivery path (should one exist). Since trace packet delivery is unreliable it is necessary to allow multiple trace packet attempts to be made until a trace-ACK is received (see section 7.1.2). However it may be that the source really is unreachable, and no acknowledgement will ever be received. It would be wasteful to continue trace attempts under those circumstances.

A counter C is maintained per [S, G] and is incremented on each transmission of a trace packet containing G towards S. Receipt of a trace-ACK referring to [S, G] resets the counter. If the counter exceeds some limit L, no further trace attempts are made for [S, G] until the process is re-initiated by the application and somehow that fact is reported to the application.

It is envisaged that t2 would be of the order of a second (perhaps less) to allow recovery of a voice over IP connection within a few seconds. However, sending trace packets at this frequency would be expensive. Therefore, in the absence of any real data for [S, G] for a period t2, the encapsulating router sends a dummy 'heart-beat' ERM encapsulated packet carrying no data packet. These have ERM type 130, with a standard ERM tree header followed by [S, G].



Receipt of such a packet by the destination router causes the timer to be reset in the same way as a normal data packet and hence inhibits the recovery attempt, but no output multicast packet is generated.

Provided ALL the destination routers that are no longer reachable succeed in re-establishing a working delivery path, any 'dead' branches will be completely pruned by the mechanisms described in section 6.3.1. However, if one or more destinations remain unreachable the branches leading to them will not be pruned until the expiry of the destination holding timer.

6.4.2.2. Silent sources

The heartbeat mechanism described above is satisfactory for dealing with short periods of silence, but becomes somewhat inefficient when sources exhibit long periods of silence. This can be alleviated by defining a holding time in the heartbeat, which would normally be n*t2. This is the period after which the receiver will begin to trace at t2. Setting this to a longer time allows the heartbeat rate to be reduced or even ceased altogether. However, this introduces a problem when transmission is resumed, since there is no mechanism to force a trace if there has been a failure during the period of silence.

A solution to this problem is as follows. On resumption of transmission the source ERM router sets a timer per destination router, and ERM multicasts a heartbeat packet with a zero holding time, thus triggering traces from all the destination routers. Receipt of the trace from a particular destination router clears the timer, but if no trace is received from a particular router, expiry of the timer causes a heartbeat packet to be unicast directly to the destination router concerned. If the destination is reachable at all, this packet (or possibly a subsequent one if there is random packet loss), will cause the destination to begin tracing to repair the ERM path. Failure to receive a response after (some number of) retries will cause the destination to timeout completely.

6.4.2.3. Router reachability failures

ERM router reachability failures are indistinguishable from router failures, and are dealt with by the same mechanism.

6.4.3 Destination ERM router failures

6.4.3.1. Router Failures

Failure of the destination router causes state for the [S, G] to be lost. If there is only one ERM router to which the multicast receiver can join, then recovery is impossible (until the router in question is re-booted). The branch of the tree leading to the unreachable destination will eventually be pruned by the expiry of the destination holding timer as described in section 6.3.5.1.

If there are multiple routers, then normal multicast operation will result in another router receiving the IGMP joins, and beginning the trace registration process in its own right. However the source ERM

router will treat this as a completely distinct delivery point, and will continue to attempt delivery to the old DR until its holding timer expires as above. This will result in a period of unnecessary packet transmission, but this will usually be restricted to the last hop.

6.4.3.2. Router reachability failures

These are dealt with as for intermediate ERM router reachability failures as described in section 6.4.2.3. If another route exists, recovery will be complete. If not, the destination will eventually be pruned by the expiration of the destination holding timer as described in section 6.3.5.1.

6.4.4 Source ERM router failures

If the source ERM router fails, all the tree state will be lost. Normal recovery mechanisms will result in destination nodes re-sending trace packets towards the source. If another route to the source exists, this may result in a new router becoming the encapsulating router and building a new tree as usual.

If the source ERM router doesn't fail, but is partitioned from the rest of the network, a new source ERM router may be initiated while the old source ERM router eventually (n*t1) prunes off its delivery tree as a result of the failure of periodic destination refresh.

7. Route discovery

The information for building the delivery tree is obtained from trace packets sent from the destination nodes towards the source of the group. The exact form of the trace packet mechanism is described in section 7.1. The way the delivery nodes discover the source address is described in section 11.

In the first instance we will assume that the source ERM router is the (single) router adjacent to the source. I.e. the ERM router knows that it is the encapsulating source ERM router by its proximity to the source. Later we will discuss how to extend this to permit the source to be separated from the encapsulating router(s) by a multicast cloud. Details of this are described in section 7.2

7.1 Trace Packets

A trace packet is sent from a destination when the first member of a group joins, and periodically as described above. The trace packet builds a list of ERM capable routers traversed in order to reach the source ERM router.

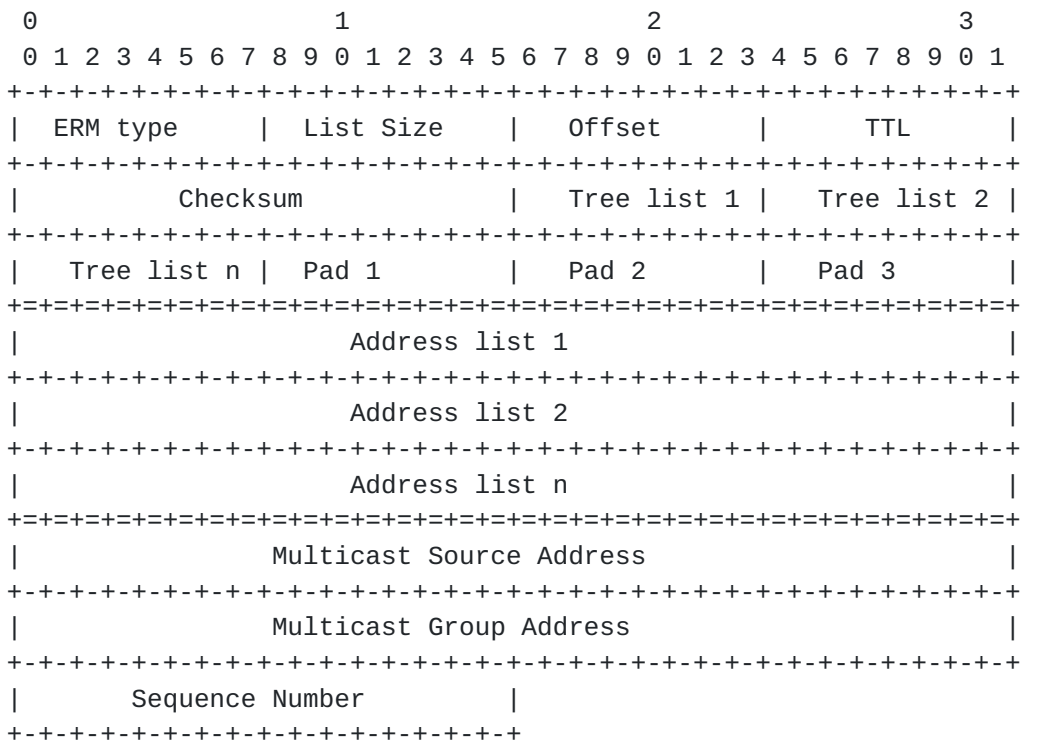
7.1.1 Choosing a unique IP address

There needs to be some guarantee about which IP address goes in the list. If the same node puts different IP addresses in the list for the same path, it will prevent correct identification of the delivery tree. If it uses the IP address of the outbound (from the source) interface, then a single node may appear as different nodes in two different traces, hence causing packet duplication over the upstream hop. If it uses the inbound (from the source) interface, (i.e. the interface address the trace packet is forwarded over), crossing paths won't be detected, since they will appear to be different ERM routers. Always choosing the lowest interface address avoids these problems, but it may cease to be available if the interface fails, causing perturbation of the ERM routes, even the infterface was not directly involved.

7.1.2 Acknowledgement of trace packets

The source ERM router acknowledges receipt of a trace packet, by sending an ERM encapsulated packet to a sub-tree of the optimised multicast delivery tree, which contains only the relevant destination router. No additional optimisation is performed on the tree, which may therefore contain multiple hops. Thus the acknowledgement packet is delivered over the same path which will be used for the delivery of multicast traffic and 'shares fate' with that traffic. Note that the acknowledgement of the first trace packet for the group will be delivered directly to the destination router, since the multicast tree will consist entirely of that one hop. As more destination routers are added, the tree will approach the final multicast delivery tree.

The ERM type is 129 (trace-ACK - the high order bit indicating that it contains an ERM route and should be forwarded using standard ERM forwarding) and the 'encapsulated data' consists only of [S, G] and the two byte sequence number of the trace packet being ACKed.



On triggering a trace packet for a group, the destination router sets the value of current_source_ERM_router for that group to zero. The trace packet is re-sent every TRACE_REPEAT_INTERVAL seconds, incrementing the sequence number on each transmission until a trace ACK with the current sequence number is received. (The value of TRACE_REPEAT_INTERVAL needs to be guaranteed to be greater than the normal round trip time for trace /trace-ACK packets between the destination and the source). The IP address of the source ERM router for that multicast source (from the IP source address of the trace ACK packet) is then recorded in current_source_ERM_router. This is used to detect changes in the Source ERM router (see section 7.4).

7.1.3 Becoming the source ERM router

When an ERM router determines that it is the source ERM router (as described elsewhere), it performs the actions associated with a member of that group sending an IGMP register. I.e. it does a PIM join or whatever, to pull down the multicast traffic for that [S, G].

7.1.4 Ceasing to be the source ERM router

When the last destination for [S, G] is removed from a source ERM router (either as a result of receiving an ERM prune, or as a result of the destination holding timer expiring), the router performs the

appropriate multicast leave operations and purges all state for [S, G].

7.1.5 Incongruent unicast and multicast topologies

The ERM trace packet follows the unicast topology towards the source, and hence the distribution tree branch points will be determined by the unicast routing topology. However, where the unicast and multicast topologies are incongruent (as provided for by MBGP, for example), an ERM capable router can choose to forward the ERM trace packet according to its multicast forwarding information. Provided ERM capable routers are located at the critical divergence points between the two topologies, this will allow some degree of separation of the two topologies. The separation will not be complete, however for two reasons.

1. When forwarding ERM trace packets a non-ERM capable router will choose the unicast forwarding path even if it has a multicast path (since it has no knowledge that the ERM trace packet has anything to do with multicast).

2. When forwarding ERM data between ERM branch points, any router (whether or not it is ERM capable) will forward according to the unicast forwarding path to the next branch point.

There is a risk of looping trace packets, where the unicast and multicast topologies are sufficiently diverse and the trace packet is forwarded according to the multicast topology by some routers and according to the unicast topology by others. The capability of ERM routers to forward trace packets according to the multicast topology should therefore be used only with some caution.

7.2 Trace packets

A unicast packet containing the router alert option is addressed to the source address. It is forwarded normally by non-ERM capable routers (but it will incur the penalty of RA processing to determine it is not interesting). ERM capable routers append their IP address to the list, update the offset and re-forward it towards the source address.

The packets have the following format (see also the modifications proposed in sections 7.3.1 and 8.1):-

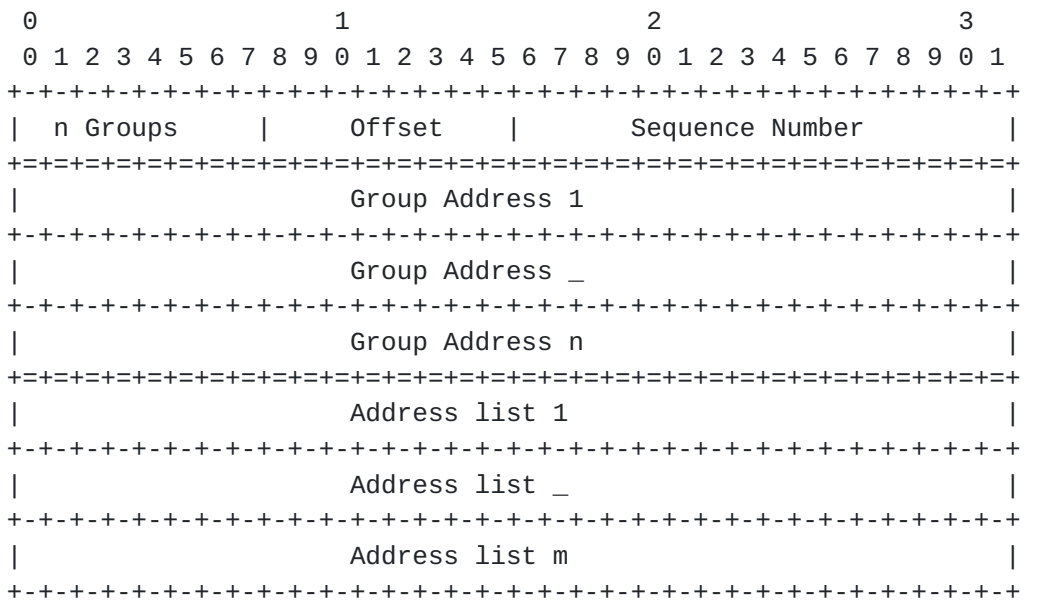
- Normal unicast IP header with RA option and "Don't Fragment" set.

- Destination address = source address of multicast group

- Source Address = Destination router ID

- Total Length = IP Header length + (max trace length +1)*4

- Protocol = ERM
- Data, comprising



- n Groups (1 byte) = Number of Group Addresses
- Offset (1 byte) = Offset (in 4 byte units) of next free position (initialised to zero)
- Sequence number (2 bytes)
- Group Address (n Groups * 4 bytes) = List of Group Addresses to which this trace refers
- Address List (max trace length * 4 bytes) = List of ERM router addresses (initialised to zero)

7.3 Locating the source ERM router(s)

Up to now, it has been assumed that the source ERM router is adjacent to the source host and can identify itself as such. The source ERM router

1. Records the state information from the trace packets, and builds the delivery tree.
2. Sends a trace-ACK to the originator of the trace packet.
3. Encapsulates subsequent multicast data packets for [S, G].
4. Performs whatever actions are necessary to pull down multicast traffic for the group (see section 7.1.3)

An ERM router that is NOT the source ERM router retains no state from the trace packets it forwards (in accordance with design constraint 1).

However, this assumption places the considerable constraint on the design that all sources must be adjacent to an ERM capable router. In reality the source may be separated from the 'first' ERM capable router by a conventional multicast domain because

1. It is not feasible to deploy ERM capable routers adjacent to every host.

2. It is required for administrative reasons to use conventional multicast for that portion of the delivery tree.

7.3.1 Remote source detection.

A small modification to the format of the trace packet allows the above constraint to be relaxed. The data portion of the 'trace' packet is carried as the data portion of an ICMP echo request packet with destination address the group source IP address, source address the destination router IP address (initially) and an RA option. The ICMP echo request identifier is set to the protocol type assigned to ERM to provide some protection against aliasing with genuine 'ping' traffic.

An ERM router intercepting the packet updates the trace information with its own unique IP address (including adjusting the offset pointer and the IP header total length field) and also sets the source IP address of the ICMP packet to be its own unique IP address. Both the ICMP and IP header checksums must be modified as appropriate.

Any source host receiving the packet acts on it as a normal echo request and returns it to the last ERM router (i.e. the most recent value inserted as the ICMP source IP address) as an echo reply with the trace data intact. On receiving such a packet, the ERM router establishes itself as source ERM router, builds the initial part of the tree from the enclosed trace list, and sends a trace-ACK to the initiator of the trace (i.e. the first address in the trace list).

There's an obscure case that needs discussion. Since the returned echo response packet will, presumably, still have the RA option set, it will be examined by all routers between the source and the source ERM router. It is possible, perhaps as a result of dynamic topology changes, or asymmetric routing, that one or more of these routers may be ERM capable. This gives rise to the strange situation that an ERM router has been found that appears to be 'closer' to the source than the router which had previously been identified as the source ERM router. However, this is 'closer' in the source to destination sense. Since multicast routing uses RPF, the original, which is closer in the destination to source sense, is preferred. If it turns

out that the dynamic routing changes converge such that the second router really is 'closer' in the required direction, then the source ERM router change procedures described in section 7.4 will ultimately resolve the situation. Hence any echo responses seen by

an ERM capable router, which are not directly addressed to it, can safely be ignored.

7.4 Changing the source ERM router

When the source ERM router is not an immediate neighbour of the multicast source, routing changes may result in a different source ERM router being identified by subsequent trace packets. The new source ERM router will begin to encapsulate data packets down the delivery tree, but the original source ERM router will also continue to encapsulate packets down its delivery tree, until the destination router holding timer (see section 6.3.5.1) expires. Thus multicast data will be duplicated for the period of the destination router holding timer.

In order to minimise the period of duplication, a destination router checks the source address (i.e. the address of the encapsulating source ERM router) of each ERM encapsulated packet received, including heartbeat packets. If it does not match the value of current_source_ERM_router corresponding to the IP source address of the encapsulated multicast packet (or that of the heartbeat packet), it indicates that duplicate data may be being received. The data (if any) is delivered in any case (a short duration of duplication is preferable to the risk of dropping data erroneously), but an ERM prune-change is triggered, to be unicast directly to the unrecognised source ERM router. These prunes are rate limited.

A value of zero in current_source_ERM_router (indicating that the current source ERM router is unknown because a trace is in progress, as described in section 7.1.2), is deemed to match any source ERM router. No prunes are sent until the correct source ERM router has been identified, by receiving a trace-ACK.

A value of NO_MEMBERS in current_source_ERM_router (indicating that the destination router no longer has members for the group as described in section 6.3.5) is deemed to match no source ERM router. Hence, rate limited prune-leaves are sent to the source address of the encapsulated packets in response to ERM encapsulated data for the group from any source ERM router.

7.5 Failures in the multicast delivery to the source ERM router etc.

Since the source ERM router is sending heartbeats towards the destinations to suppress traces even in the absence of multicast data, only periodic traces will be generated while the delivery path between the source ERM router and the destinations remains intact. This is true even if there is a multicast delivery failure between the source and the source ERM router(s). If the traces from the

destinations had not been suppressed, they might have been able to discover a new source ERM router, which had connectivity to the source.

It is not possible to use a heartbeat from the source to the source ERM router(s) to detect this type of failure, since this would require co-operation from the source host. Once a source ERM router has joined the conventional multicast delivery tree, it is the conventional multicast protocols which will (attempt to) maintain the delivery path from the source to the source ERM router(s). Failures of intervening routers and links should (if connectivity still exists at all) not affect the reliable delivery of multicast data to the source ERM router(s). If multicast routing fails to deliver multicast data to a particular source ERM router, then it is possible that the ERM router has become partitioned from the conventional multicast network. If this is the only feasible source ERM router, then recovery is impossible. But it may be that some other potential source ERM router still has multicast connectivity.

Each destination router is sending periodic traces at the rate of once per t1 seconds. In the steady state these will all converge on the source ERM router in question. Thus, there are m opportunities per t1 seconds for a periodic trace packet to discover an alternative source ERM router, where m is the number of destination routers associated with both the multicast source and the source ERM router in question.

When such a trace packet discovers an alternative source ERM router, the mechanisms described in section 7.4 will cause a prune-change message to be sent to the original source ERM router. On receipt of such a prune-change message, the source ERM router performs the normal prune-leave action of removing the associated destination router from the delivery tree. In addition, it ceases transmitting downstream heartbeat packets to all destination routers associated with the source. Sending of heartbeat packets is not resumed until a period of (N+1)*t2 seconds has elapsed and a new trace packet for the source has been received. In the absence of genuine multicast traffic, this will cause the remaining destination routers served by this source ERM router to begin non-periodic tracing, and hence rapidly discover the new source ERM router if appropriate. If, on the other hand, multicast data is still arriving at the source ERM router, then this confirms that the conventional multicast delivery tree is still intact, and there is no harm in the non-periodic trace messages continuing to be suppressed.

The effect of these mechanisms is that such a failure will be repaired for the first destination in an average time of about t1/m seconds, and the remaining destinations should catch up in a further period of n*t2 seconds.

7.6 Summary of timers

The previous sections have identified a number of timers. Their use is summarised here for clarity.

7.6.1 Periodic timer t1

This timer is used for periodic functions to discover more optimal topology. Destination routers send periodic trace packets every t1 seconds, and failure to receive such a packet from a destination router for a period of n*1 results in the state for the destination being pruned.

A plausible value for t1 is 60 seconds.

7.6.2 Error recovery timer t2

This timer is used for protocol functions associated with the recovery from errors such as failed routers and links. The source ERM router guarantees to send ERM encapsulated data (genuine multicast traffic, heartbeats, or track-ACKs) at least once per t2 interval. Failure to receive such data for a period of n*t2 results in the destination router initiating a trace. In the absence of a trace-ACK, such traces are repeated up to TRACE_FAILURE_COUNT times at an interval of TRACE_REPEAT_INVERVAL. Once the count has been exceeded, the destination router abandons further attempt to join that [S, G] (until when?).

Service interruption as a result of router or link failure will be at least n*t2 seconds, rising in increments of TRACE_REPEAT_INTERVAL seconds if trace packets are lost.

A plausible value for t2 is 1 second.

7.6.3 Trace Repeat Interval

The interval between non-periodic trace attempts. This is probably n*t2.

8. Multicast capable subnetworks

The algorithms described so far will not take advantage of a subnetwork that has layer 2 multicast capability. A separate copy of the data packet will be unicast to each child router on the subnet. This compares poorly with conventional IP multicast which will (usually) multicast a single copy of each data packet to all the downstream routers on the subnet. The following sections describe enhancements to permit a similar optimisation for ERM.

8.1 Modified trace packets

When transmitting a trace packet, the source IP address of the enclosing ICMP packet is set, not to the ERM 'unique' IP address, but to the actual transmitting interface IP address. The ERM

'unique' address is still inserted in the trace list as before.

When receiving a trace packet over a layer 2 multicast capable LAN (only), a check is made to determine whether the source IP address of the enclosing ICMP packet is a direct neighbour over that interface. If so an additional pseudonode identifier is inserted into the trace list before also inserting the ERM 'unique' address as normal. The pseudonode identifier is assigned uniquely to the interface within the scope of that router, and has the top 3 bits (i.e. class D) set to allow it to be distinguished from a genuine node identifier (since a class D address will never be used as the ERM 'unique' address).

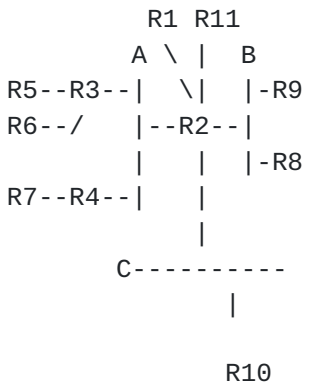
The additional cost of this is 4 bytes per multicast capable subnetwork traversed by the trace packet. In the worst case this could double the number of entries.

8.2 Modified tree building algorithm

When processing the trace packet the pseudonode identifier is removed (i.e. it never appears in the tree address list placed in the ERM data packet header.). However its presence is noted. If the set of logical children of a particular pseudonode (i.e. the children of the parent of the pseudonode whose traces include the pseudonode) has two or more members, those children are retained even if they themselves have only one child. This ensures that the routers that will receive the multicast ERM data packet will always appear in the address list even if they are not a branch point. This is necessary to enable them to identify their position in the delivery tree, since the multicast packet is of necessity the same for all recipients. In addition each pseudonode of a particular router is given a unique identifier in the range 1-15. This may or may not correspond to the original interface number, which may or may not be encoded in the pseudonode identifier carried in the trace packet. This identifier is encoded in the top 4 bits of the tree list entry for each child (the bottom 4 bits being the offset of the parent).

In order to maximise the size of distribution tree which can be accommodated within the 4 bit parent offset restriction, it may be observed that no leaf node (i.e. one with no children, whether or not it is a delivery point), by definition, is ever referenced as a parent. By arranging that all leaf nodes appear at the end of the trace list (this can be done without breaking the pre-ordering requirement), it can be guaranteed that all of the 15 available parent offset Ids are assigned to nodes which are referenced as parents.

8.3 An example



We would expect the following trace packets towards router 1.

5,3,A,2,1
6,3,A,2,1
7,4,A,2,1
10,C,2,1
8,B,2,1
9,B,2,1
11,2,1

We keep 4, even though it is not a branch point, because A has multiple children (3 and 4).

The resulting trace list would be (using hex to make the top and bottom 4 bits clearer)

00,01,12,12,03,03,04,22,22,02,02

with an address list of

r1,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11

8.4 Modified forwarding algorithm

When searching for children, only the bottom 4 bits of the trace list entry are compared to the parents index. Thus in the above example, R2 will treat R3, R4, R8, R9, R10 and R11 all as its children. However, it performs the following additional tests on the set of children.

1. If the top 4 bits are zero, a copy is unicast to the child as normal. Thus R10 and R11 receive unicast copies.

2. If the top 4 bits are non-zero, it performs a next hop lookup on

the corresponding IP address, to determine the output interface, and multicasts, to all-ERM routers, a copy over that interface. I.e. the destination address of the outer IP header is set to the

all-ERM routers IP multicast address. It then marks all subsequent entries in the children set as having been processed. Thus R3 has an entry of 12, hence it looks up R3, determines that interface A is the output interface, multicasts a copy over that interface and sets the remaining entries for 12 (R4) as processed. The next child of R2 is then R8, and the process is repeated over interface C.

On receipt of a multicast copy, an ERM router needs to find its own position in the address list. This could be achieved by always scanning the address list looking for one's own address, but that involves n 4 byte compares. The number of compares can be restricted by the following algorithm. Note that when sending the multicast copy, the pointer is always adjusted for the first of the children.

If the entry corresponding to the pointer has the top 4 bits zero no address check is required and the algorithm works as before. If the top 4 bits are non zero, the address corresponding to the pointer is checked against the receiving router's own address. If it matches, the current location is correct and forwarding can proceed as normal. If not, it searches along the tree list (starting from the current location) looking for the same value (including the top 4 bits) as the current entry. If it find it, it again checks the corresponding address, and if it matches, the search terminates and the packet is forwarded as normal, otherwise the search continues. If the search fails to find an entry the packet is discarded, since this must have been a multicast packet received by virtue of a router which is not a member of the tree being on the same LAN as members of the tree.

To proceed with the example, when R3 receives the multicast copy it finds the pointer at 3, and the top 4 bits of the entry have the value 1. It therefore checks its own address against 3, finds a match and forwards normally. When R4 receives the packet, it performs the same checks, but this time the address doesn't match, so it searches for the next entry (4) with the value 12. The address check of 4 now succeeds and forwarding proceeds normally. If there were another router (R12 say) on LAN A, which was not part of the delivery tree, it too would receive the multicast packet, but both checks would fail, and so it would be discarded. Unfortunately this means that on a LAN with n downstream members of the tree, not only do each of those routers have to perform up to n checks (overall a total of n(n+1)/2 checks per packet), but every router on the LAN which is not a member of the tree must also perform n (failing) checks.

8.4.1 Setting the TTL

Transmitting to the all-ERM routers multicast with a TTL greater than one could potentially allow these packets to be multicast forwarded off the LAN. Therefore the TTL in the outer IP header of such packets MUST be set 1. In order to preserve the 'real' TTL,

this is first copied from the outer IP header to the TTL field of the ERM header. On reception of such a multicast packet, the original TTL (suitably decremented) is restored in the outer IP header.

9. Further Optimisations?

Further optimisations may be possible, taking advantage of the fact that there may be multiple groups that share the same source or source ERM router(s). On balance, they may not be worth the extra complexity they introduce, but they are discussed below for completeness.

9.1 Multiple groups with the same source

A destination router may have multiple active groups that share the same multicast source (and implicitly, the same source ERM router). In this case a single trace MAY be sent for the entire set of groups, containing a list of the group addresses to which it refers. Not only does this reduce the bandwidth requirements for trace packets, but it may also reduce the memory requirements in the source and destination ERM router(s). In this case the destination ERM router need maintain current_source_ERM_router only per source, and not per [S, G].

However, separate trace-ACKs will still be required, since the ERM delivery trees for the groups may be different, owing to their different membership. For similar reasons, it is not possible to use traffic or ERM heartbeats arriving at a destination router for one group to imply correct operation of any other group even though they share the same source and source ERM router.

9.2 Minimising the source impact of trace packets

The algorithms described above require the multicast source to process at least one trace packet per periodic time t1 for every distinct destination router served by all groups to which it is transmitting. Under normal conditions the traces will all converge at one (or more) source ERM router(s). The portion of the trace from the source ERM router to the multicast source and back is only necessary to detect the arrival of a new ERM capable router closer to the source (once the initial detection of the source ERM router has been accomplished). Limiting the number of such packets can therefore reduce the load on the multicast source.

When a router sees a trace packet travelling to the source S, and it already has encapsulation state for [S, *] (I.e. it is a source ERM router for [S, *]) it can 'short circuit' the delivery of such trace packets provided at least one such packet per t1 interval is allowed

to pass unimpeded. Such short-circuited packets must be processed as if they had been received as responses from the source. This reduces the load on the multicast source to one trace packet per t1

interval, but still maintains the possibility of discovery of a closer source ERM router within that interval.

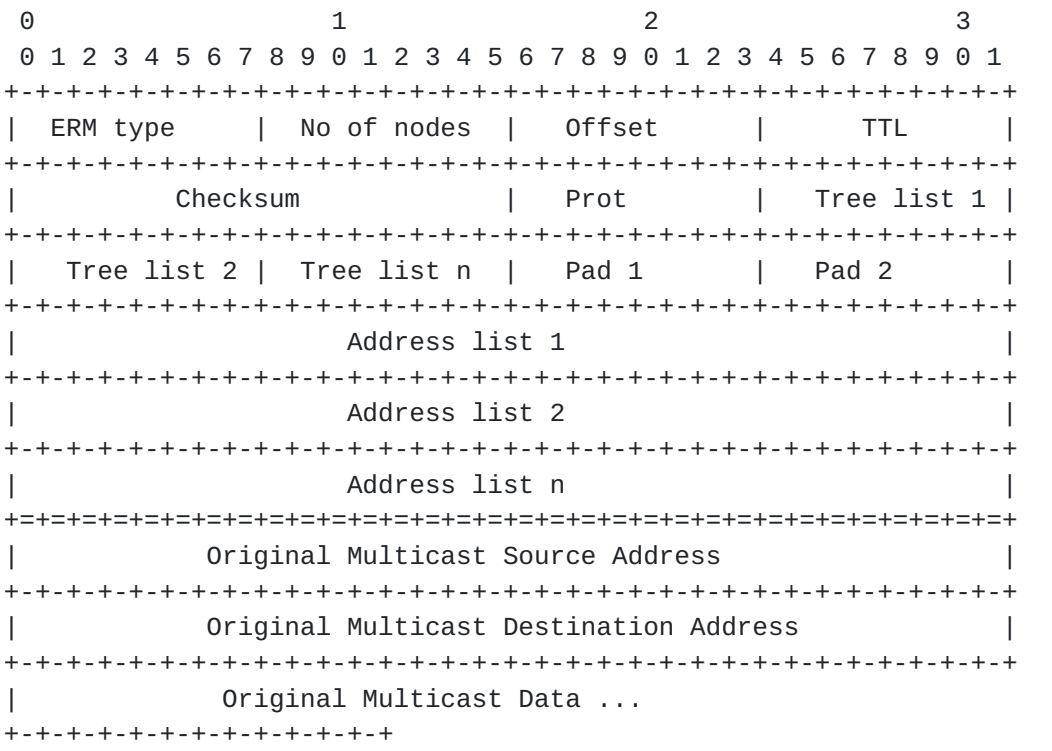
Note that this optimisation does not necessarily conflict with the mechanisms described in section 7.5 for recovering from failures which require the detection of a new source ERM router. Only those trace packets that actually pass through the current source ERM router are affected. Where the path from the destination router to the potential source ERM router does not pass through the existing source ERM router there will still be m opportunities per t1 seconds.

9.3 Multiple groups with the same source ERM router, but different sources

Since the sources are different, it is necessary to use separate traces, as they may subsequently identify different source ERM routers for the groups. The same arguments as above preclude the use of common trace-ACKs or heartbeats unless the delivery trees of the groups are identical.

9.4 Minimal Encapsulation

In the existing encoding for an ERM data packet, several fields are duplicated between the original multicast header and the IP header of the ERM data packet. By using similar techniques to RFC 2004 [3 "Minimal Encapsulation within IP", this duplication can be avoided. The modified ERM data packet is as follows:-



The original multicast source and destination addresses from the multicast packet (together with the data of course) are kept intact, but the preceding parts of the original header are stripped out.

The original Prot is stored in the ERM header (the outer IP header prot, being set to ERM).

ToS, ident, frags and TTL are copied into the outer IP header (and copied back on decapsulation).

The ERM header TTL field is still required for multicast capable LANs as described above in section 8.

The ERM checksum covers the original multicast source and destination addresses as well as the other ERM header fields. However if all the ERM header fields were covered it would be necessary to incrementally adjust the checksum when the offset (and TTL) are changed. Therefore, the ERM checksum is defined to start at the checksum field. Omitting the offset exposes the venerability to corruption of the offset but this can only cause the following errors: -

1. If it is corrupted to an earlier value, the packet may be returned to a previous point in the delivery tree, causing duplication along the branches not leading to the current router.

The branch below the current router will receive a single copy of the packet. No data loss will occur, and a single instance of corruption will only cause a single instance of duplication, since the contents of the offset field will be reset.

2. If it is corrupted to a later (but still valid) value, the intervening branches will suffer packet loss (since the packet will appear to 'jump' to the later point in the tree).

3. If it is corrupted to an invalid value, the branch below the current router will suffer packet loss.

9.4.1 Benefits

This saves 11 bytes (or thereabouts, because of alignment issues) compared to the full encapsulation. For example, in what might be a common case of three delivery points from a common fan out (i.e. 3 addresses in the address list), the minimal encapsulation would cost a total of 32 bytes encapsulation overhead, compared to 44 bytes with full encapsulation headers. For a typical uncompressed VoIP packet, which is about 50 bytes, that's a 64% overhead compared to 88%. This should be compared to the 'overhead' of using separate multicast packets, which in the above case would be 200%, since 3 unicast packets would be required.

9.4.2 Costs.

This form of encapsulation is less efficient for encapsulation and decapsulation. It also precludes the possibility of using a separate ToS value for the ERM encapsulation, since this must be copied from the original multicast packet.

9.4.3 Problems with Fragmentation.

Fragmentation is an issue with ERM, whatever encapsulation is used, since an ERM packet may be fragmented by intermediate non-ERM routers (i.e. by performing normal IP fragmentation on the outer IP header). Since not all fragments will contain the ERM header, and hence cannot be ERM forwarded, it is necessary for ERM routers (i.e. the destination of the outer IP header) to perform re-assembly before ERM forwarding.

The situation is complicated if the 'minimal encapsulation' ERM header is used, since there is then only one set of fragmentation information. If the multicast packet were already fragmented before ERM encapsulation, it would invoke re-assembly at each ERM hop. Presumably, since it required fragmentation in the first place, it would then need to be re-fragmented for transmission.

Given that (potential) re-assembly at every ERM router is highly undesirable, the best solution may be to set the "don't fragment" bit in the outer IP header, and hence never do any re-assembly at an ERM router. (This would be desirable even with a full header encapsulation). In the case of 'minimal encapsulation' it would be

necessary to find a single bit somewhere in the ERM header to carry the original value of the DF bit. However, if the original multicast packet had previously been fragmented this could result in a packet

with DF set AND non-zero values for either or both of MF and fragment offsets! It is not clear whether this would be treated as an error by any IP implementation. If it would, then it would be necessary to store the whole 16 bits of the fragmentation fields in the ERM header, which makes the 'minimal' somewhat less attractive.

9.5 Final hop optimisation

The ERM tree information is never required for the final hops (that is, from the last fan out point to the delivery router - except of course for the cases where the last fan out router IS a delivery router). By stripping this out of the packet before the final forwarding, another 20 bytes could be saved (for the 3 way example) reducing the overhead to 24% for those hops. (The ERM type, prot, checksum and the SA and DA. are still required, giving a 12 byte overhead = 12 bytes). Such a packet shrinking operation is likely to be rather costly, but could perhaps be justified by the fact that the last hop is likely to be at the edge of the network and hence have lower bandwidth capable links.

Another way of looking at this is to say that for any particular hop, sending a single ERM packet is roughly comparable to sending 2 unicast packets (for 50 byte packets and small tree lists). So it is only on the final hops (where there would be no packet duplication even in the multiple unicast) that ERM is at a serious disadvantage. On hops which would require 3 or more unicast packets ERM almost always wins (except for large tree lists > 15 nodes, which is outside the design scope).

Of course ERM can never do better than true multicast.

10. Limiting the size of a group

Explicit Route Multicast is designed to operate with a 'small' group. However, the limiting factor is not the size of the group (i.e. number of group members) per se, but rather the size of the encoded delivery tree. This depends on the number of destination routers (which may each serve multiple group members) and also on the topology of the delivery tree.

N delivery nodes require an encoded tree of length 2N-1. The best case (ignoring the trivial case where the encapsulating router sends a packet directly to each delivery node, requiring a zero length encoded tree) is where each node on the tree is itself a delivery node, which requires an encoded tree of length N. Thus for any set of N delivery nodes, the encoded length may vary between N and 2N-1 depending on the topology. Since the topology may change during the lifetime of the group, the encoded tree length may also change

between these limits even if the number of delivery nodes remains constant. Equally, the number of delivery nodes may change as nodes join and leave the group.

A number of possible strategies for controlling the size of the encoded tree are discussed below.

10.1 Limiting the number of members

The absolute worst case is when each delivery node serves a single member, and the topology requires worst case encoding. Hence, if a maximum of 5 group members were permitted, the encoded tree would never exceed 9 addresses. This would be a simple strategy to explain to users, but is severely restrictive. It is also hard to police, since the actual number of group members is unknown to the ERM protocols.

10.2 Limiting the number of destination routers

Since each destination router is required to perform a trace and receive a trace-ACK when it 'joins' the group, it is possible for the source ERM router to check the current number of destination routers, and reject the join attempt (by sending a trace-NACK) for a new destination router. As above a hard worst case limit can be chosen which will guarantee an upper limit on the size of the delivery tree irrespective of any topology changes.

10.2.1 Multiple source ERM routers

Where there are multiple source ERM routers, each ERM router will independently acquire a set of destination routers, and limit the size of only that subset. Subsequent topology changes could then make one or more source ERM routers redundant which may in turn cause one or more of the remaining source ERM routers to exceed the limit. Possible approaches to solving this problem are:-

1. Dynamically remove one or more destination routers - not very friendly to existing users, but at least its simple! The normal algorithm of rejecting those above the threshold could be used. Doing anything else, such as LIFO, would be difficult. Since the state for the 'old' source ERM routers will be lost, if the new set of source ERM routers has no overlap with the old set, ALL the destinations will appear to be new. The first destinations to send traces will then be (arbitrarily) accepted and the remainder rejected.

2. Allow the encoded tree size (and hence the packet size) to exceed the desirable limit - but since the number of source ERM routers is unbounded, so too is the size of the encoded tree.

3. Adjust the delivery tree to remove one or more intermediate nodes at the expense of making the delivery tree less efficient, since multiple copies of a packet would be sent over some links. A

similar effect could be obtained by splitting the delivery tree into two (or more) parts, each of which is below the critical limit. Some intelligence into exactly which nodes to eliminate

could be introduced by having the ERM trace message include the current value the ICMP message hop count for each entry (from which the number of unicast hops corresponding to each ERM hop can be deduced.) This could be used as a weighting when evaluating the modified delivery tree.

4. Communicate the number of destination nodes associated with each source ERM router and enforce a limit for the entire set.

10.2.2 Topology changes with a single source ERM router

Even when there is only a single source ERM router, (or where there exist multiple source ERM routers, but their sub-trees do not become merged), a topology change can potentially result in a factor of 2 (actually (2N_1)/N) increase in the size of the encoded tree. This can be contained by limiting the maximum number of destination routers assuming the worst case topology. Alternatively, a more optimistic assumption about the topology could be made, and worse cases dealt with by using the techniques outlined in bullets 1-3 above.

10.2.3 Failure of a destination router

As has already been described in Section 6.4.3.1, it is possible that a destination router may fail and be replaced by the election of a new designated router. The failed router would still appear as a destination router until the expiry of the timeout period. There may be other circumstances where a router is being unnecessarily included in the delivery tree while awaiting timeout. This could cause the tree size to exceed the limit, and result in the 'new' router being rejected.

Under these circumstances the source ERM router MAY send one or more heartbeat packets with a zero holding time (as described in Section 6.4.2.2) and reduce the timer associated with each of the destination routers. Any destination routers that do not quickly respond with a non-periodic trace can then be eliminated in a timely manner, thus freeing up resources for the 'new' router.

10.3 Effect of tree encoding length on MTU

The space required in the ERM header for the encoded tree is unpredictable, and may vary during the lifetime of a group (as a result of topology changes, or joining and leaving of destinations). If the header size is kept to the minimum capable of containing the current delivery tree, then the header size, and hence the available MTU, will also vary. Conversely, if sufficient space in the header is always allocated to contain the worst case encoded tree, the MTU will remain constant, but there will be significant wasted

bandwidth. The variation is approximately 5(N-1). So for N = 5 it is about 20 bytes, and for N = 10 about 45 bytes - a significant fraction of the total packet size for small payloads.

The best compromise is to calculate MTU assuming worst case tree length, but adjust the header length to reflect the current encoded tree length requirements.

11. Source Discovery

ERM requires that the multicast source(s) for a particular group are known by all the egress routers (so that they can send traces towards the source(s)). The means by which the source addresses for a particular group are assigned are outside the scope of this draft.

12. Security Considerations

Security considerations will be addressed in a future version of this draft.

13. References

1 Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

2 Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997

3 Perkins, C.,"Minimal Encapsulation within IP", RFC 2004 October 1996.

14. Acknowledgments

The authors would like to acknowledge contributions made by Liming Wei, Richard Edmonstone and Brian Singerman.

15. Intellectual Property Rights Notice

proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

16. Author's Addresses

Joel Bion
Cisco Systems
375 East Tasman Drive
San Jose
California
USA
Phone: 408 527-9376
Email: jpbion@cisco.com

Dino Farinacci
Procket Networks
Phone: 408-954-7909
Email: dino@procket.com

Mike Shand
Cisco Systems
4, The Square,
Stockley Park,
UXBRIDGE,
Middlesex
UB11 1BN, UK
Phone: +44 20 8756 8690
Email: mshand@cisco.com

Alex Tweedly
Cisco Systems
4, The Square,
Stockley Park
UXBRIDGE
Middlesex
UB11 1BN, UK
Email: agt@cisco.com