MALLOC Working Group
Internet Engineering Task Force
INTERNET-DRAFT
August, 1998
Expires February 1999

Deborah Estrin (USC/ISI)
Ramesh Govindan (ISI)
Mark Handley (ISI)
Satish Kumar (USC/ISI)
Pavlin Radoslavov (USC/ISI)
Dave Thaler (Microsoft)

The Multicast Address-Set Claim (MASC) Protocol

<draft-ietf-malloc-masc-01.txt>

Status of this Memo

Abstract

This document describes the Multicast Address-Set Claim (MASC) protocol which can be used for inter-domain multicast address set allocation. MASC is used by a node (typically a router) to claim and allocate one or more address prefixes to that node's domain. While a domain does not necessarily need to allocate an address set for hosts in that domain to be able to allocate group addresses, allocating an address set to the domain does ensure that inter-domain distribution trees will be locally-rooted, and that traffic will be sent outside the domain only when and where external receivers exist.

Copyright Notice

1. Introduction

This document describes MASC, a protocol for inter-domain multicast address set allocation. The MASC protocol is used by a node (typically a router) to claim and allocate one or more address prefixes to that node's domain. Each prefix has an associated lifetime, and is chosen out of a larger prefix with a lifetime at least as long, in a manner such that prefixes are aggregatable. At any time, each MASC node will typically advertise several prefixes with different lifetimes and scopes, allowing Multicast Address Allocation Servers (MAAS's) in that domain or child MASC domains to choose appropriate addresses for their clients.

The set of prefixes (''address set'') associated with a domain is injected into an inter-domain routing protocol (e.g., BGP4+ [MBGP]), where it can be used by an inter-domain multicast tree construction protocol (e.g., BGMP [BGMP]) to construct inter-domain group-shared trees.

Note that a domain does not need to allocate an address set for the hosts in that domain to be able to allocate group addresses, nor does allocating necessarily guarantee that hosts in other domains will not use an address in the set (since, for example, hosts are not forced to contact a MAAS before using a group address). Allocating an address set to a domain does, however, ensure that inter-domain multicast distribution trees for any group in the address set will be locally-rooted, and that traffic will be sent outside the given domain only when and where external receivers exist.

2. Requirements for Inter-Domain Address Allocation

The key design requirements for the inter-domain address allocation mechanism are:

- Efficient address space utilization, which naturally implies that address allocations be based on the actual address usage patterns, and therefore that it be dynamic.

- Address aggregation, that implies that the address allocation mechanism be hierarchical.

o Minimize flux in the allocated address sets (e.g. the address sets should be reused when possible.)

o Robustness, by using decentralized mechanisms.

The timeliness in obtaining an address set is not a major design constraint as this is taken care of at a lower level [MALLOC].

3. Overall Architecture

The Multicast Address Set Claim (MASC) protocol is used by MASC domains to claim and allocate address sets for use by Multicast Address Allocation Servers (MAASs) within each domain. Typically one or more border routers of each domain that requires multicast address space of its own would run MASC. Throughout this document, the term "MASC domain" refers to a domain that has at least one node running MASC; typically these domains will be Autonomous Systems (AS's). A MASC node (on behalf of its domain) chooses an address set to claim, sends a claim to other MASC domains in the network, and waits while listening for any colliding claims. If there is a collision, the losing claimer gives up the colliding claim and claims a different address set.

After a sufficiently long collision-free waiting period, the address set chosen by a MASC node is considered allocated to that node's domain. Three things may then happen:

a) The allocated prefix can then be injected as a "multicast route" into the inter-domain routing protocol (e.g., BGP4+ [MBGP]) as "G-RIB" Network Layer Reachability Information (NLRI), where it may be used by an inter-domain multicast routing protocol (e.g., BGMP [BGMP]) to construct group-shared trees. To reduce the size and slow the growth of the G-RIB, MASC nodes may perform CIDR-like aggregation of the multicast NLRI information. This motivates the need for an algorithm to select prefixes for domains in such a way as to ensure good aggregation in addition to achieving good address space utilization.

b) The node's domain may assign to itself a sub-prefix which can be

used by address allocation servers within the domain.

c) Sub-prefixes may be allocated to child domains, if any.

3.1. Claim-collide vs. query-response rationale

We choose a claim-collide mechanism instead of a query-response mechanism for the following reasons. In a query-response mechanism,

replicas of the MASC node would be needed in parent MASC domains in order to make their responses be robust to failures. This brings about the associated problem of synchronization of the replicas and possibly additional fragmentation of the address space. In addition, even in this mechanism, address collisions would still need to be handled. We believe the proposed claim-collide mechanism is simpler and more robust than a query-response mechanism.

4. MASC Topology

The domain hierarchy used by MASC is congruent to the somewhat hierarchical structure of the inter-domain topology, e.g., backbones connected to regionals, regionals connected to metropolitan providers, etc. As in BGP, MASC connections are locally configured. A MASC domain that is a customer of other MASC domains will have one or more of those provider domains as its parent. For example, a MASC domain that is a regional provider will choose one (or more) of its backbone provider domains as its parent(s). Children MAY be configured with their parent MASC domain, but in general may use heuristics (e.g. the domain to which they point default unicast routes) to automatically select one or more parent domains. Similarly, parents may be configured with children domains. At the top, a number of Top-Level Domains are connected in a (sparse) mesh and share the global multicast address space.

Figure 1 illustrates a sample topology. Double-line links denote intra-domain TCP peering sessions, and single-line links denote inter-domain TCP connections. T1 and T2 are Top-Level Domains (e.g., backbone providers), containing MASC speakers T1a and T2a, respectively. P3 and P4 are regional domains, containing (P3a, P3b), and (P4a, P4b)

respectively. P3 has a single customer (or "child"), C5, containing (C5a, C5b, C5c). P4 has three children, C5, C6, C7, containing (C5a, C5b, C5c), (C6a, C6b), and (C7a) respectively.

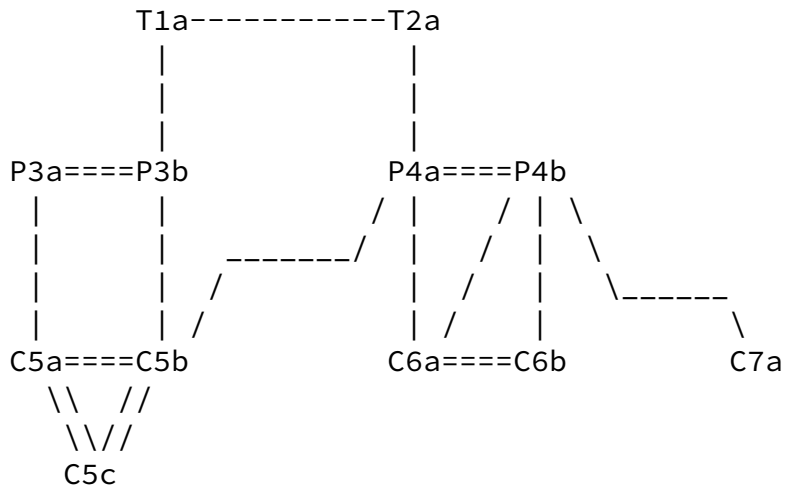


Figure 1: Example MASC Topology

All MASC communications use TCP. Each MASC node is connected to and communicates directly with other MASC nodes. The local node acts in exactly one of the following four roles with respect to each remote note:

INTERNAL_PEER

The local and remote nodes are both in the same MASC domain. For example, P4b is an INTERNAL_PEER of P4a.

CHILD
A customer relationship exists whereby the local node may obtain address space from the remote node. For example, C6a is a CHILD in its session with P4a.

PARENT
A provider relationship exists whereby the remote node may obtain address space from the local node. For example, T2a is a PARENT in its session with P4a. Whether space is actually requested is up to the implementation and local policy configuration.

SIBLING
No customer-provider relationship exists. For example, T2a is a SIBLING in its session with T1a.

A node's message will be propagated to its parent, all siblings with the same parent, and its children. Since a domain need not have a direct peering session with every sibling, a MASC domain must propagate

messages from a child domain to other children, can propagate messages from a parent domain to other siblings, and, if a Top-Level Domain, it must propagate messages from a sibling to other siblings, otherwise may propagate messages from a sibling domain to its parent and other siblings.

5. Address Space Structure

5.1. Managed vs Locally-Allocated Space

Each domain has a "Managed" Address Set, and a "Locally-Allocated" Address Set. The "managed" space includes all address space which a domain has successfully claimed via MASC. The "locally-allocated" space, on the other hand, includes all address space which address allocation servers inside the domain may use. Thus, the locally-allocated space is a subset of the managed space, and refers to the portion which a domain allocates for its own use.

For leaf domains (ones with no children), these two sets are identical, since all claimed space is allocated for local use. A parent domain, on the other hand, "manages" all address space which it has claimed via MASC, while sub-prefixes can be allocated to itself and to its children.

5.2. Prefix lifetimes

Each prefix has an associated lifetime. If a domain wants to use a prefix longer than its lifetime, that domain must "renew" the prefix BEFORE its lifetime expires (see Section 6.2). If the lifetime cannot be extended, then the domain should either retry later to extend, or should choose and claim another prefix.

After a prefix's lifetime expires, MASC nodes in the domain that own that prefix must stop using that prefix. The corresponding entry from the G-RIB database must be removed, and all information associated with the expired prefix may be deleted from the MASC node's local memory.

5.3. Active vs. deprecated prefixes

Each prefix advertised by a parent to its children can be either "active" or "deprecated". A "deprecated" prefix is a prefix that the parent wishes to discontinue to use after its lifetime expires. The "active" prefixes only are candidates for size expansion or lifetime

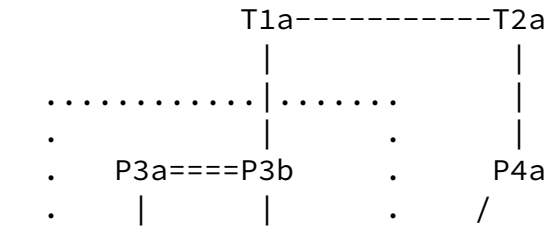
extension. Usually, this information will be used by a child as a hint to know which of the parent's prefixes might have their lifetime extended.

5.4. Administratively-Scoped Address Allocation

MASC can also be used for sub-allocating prefixes of addresses within an administrative scope zone [SCOPE]. A MASC node can learn what scopes it resides within by listening to MZAP [MZAP] messages.

A "Zone TLD" is a domain which has no parent domain within the scope

zone. Zone TLDs act as TLDs for the prefix associated with the scope. Figure 2 gives an example, where a scope boundary around domains P3 and C5 has been added to Figure 1. Domain P3 is a Zone TLD, since its only parent (T1) is outside the boundary. Hence, P3 can claim space directly out of the prefix associated with the scope itself. Domain C5, on the other hand, has a parent within the scope (namely, P3), and hence is not a Zone TLD.



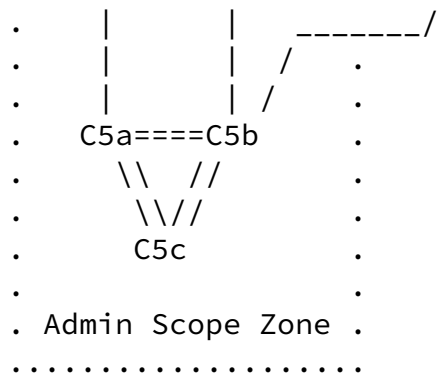


Figure 2: Scope Zone Example

It is assumed that the role of a node (as discussed in Section 4) with respect to a given peering session is the same for every scope in which both ends are contained. A peering session that crosses a scope boundary (such as the session between C5b and P4a in Figure 2) is ignored when propagating messages that pertain to the given scope. That is, such messages are not sent across such sessions.

6. Protocol Details

6.1. Claiming Space

When a MASC node, on behalf of a MASC domain, needs more address space, it decides locally the size and the value of the address prefix(es) it will claim from one of its parents. For example, the decision might be based on the knowledge this node has about its parent's address set, its siblings' claims and allocations, its own address set, the claim messages from its siblings, and/or the demand pattern of its children and the local domain. A sample algorithm is given in Appendix A.

A MASC node which is not in a top-level domain can initiate a claim toward a parent MASC domain if and only if it currently has an established connection with at least one node in that parent domain.

After the prefix address and size are decided, the claim proceeds as

follows:

a) The claim is scheduled to be sent after a random delay in the interval (0, [INITIATE-CLAIM-DELAY]). If a claim originated by a node from the same MASC domain is received, and that claim eliminates the need for the local claim, the local claim is canceled and no further action is taken.

b) The claim is sent to one of the parents (if the domain is not a top-level domain), all known siblings with the same parent, and all internal peers. A Claim-Timer is then started at [WAITING-PERIOD], and the MASC node starts listening for colliding claims.

c) If a colliding claim is received while the Claim-Timer is running, that claim is compared with the locally initiated claim using the function described in Section 6.1.1. If the local claim is the loser, it MAY be withdrawn and a new prefix must be chosen to claim. If the winning claim was originated by a node from the same MASC domain, no new claim will be initiated. If the local claim is the winner, no actions need to be taken.

d) If the Claim-Timer expires, the claimed prefix becomes associated with the claimer's domain, i.e. it is considered allocated to that domain and the following actions must be performed:

   o Advertise the prefix to its parent, and to all siblings with the same parent, by sending a PREFIX_IN_USE claim to them.

   o Inject the prefix into the G-RIB of the inter-domain routing protocol.

   o Send a PREFIX_MANAGED message to all children and internal peers, informing them that they may issue claims within the managed space. A sub-prefix may then be claimed for local usage.

Each MASC node receives all claims from its siblings and children. A received claim must be evaluated against all claims saved in the local cache using the function described in Section 6.1.1. The output of the function will define the further processing of that claim (see Section 12).

6.1.1. Claim Comparison Function

Each claim message includes:

- a "type", being one of: PREFIX_IN_USE, CLAIM_DENIED, CLAIM_TO_EXPAND, or NEW_CLAIM (PREFIX_MANAGED and WITHDRAW are not considered as claims that have to be compared)
- timestamp when the claim was initiated
- the claimed prefix and lifetime
- MASC Identifier of the node that originated the claim

When two claims are compared, first the type is compared in the following order:

PREFIX_IN_USE > CLAIM_DENIED > CLAIM_TO_EXPAND > NEW_CLAIM

If the type is the same, then the timestamps are used to compare the claims. In practice, two claims will have the same type if the type is either NEW_CLAIM (ordinary collision) or PREFIX_IN_USE (signal for clash). When the timestamps are compared, the claim with the smallest, i.e. earliest timestamp wins. If the timestamps are the same, then the claim with the smallest Origin Node Identifier wins.

6.2. Renewing an Existing Claim

The procedure for extending the lifetime of prefixes already in use is the same as claiming new space (see Section 6.1), except that the claim type must be CLAIM_TO_EXPAND, while the Address and the Mask of the claim (see Section 8.3) must be the same as the already allocated prefix. If the Claim-Timer expires and there is no collision, the desired lifetime is assumed.

6.3. Expanding an Existing Prefix

The procedure for extending the lifetime of prefixes already in use is the same as claiming new space (see Section 6.1), except that the claim type must be CLAIM_TO_EXPAND, while the Address and the Mask of the claim (see Section 8.3) must be set to the desired values. If the Claim-Timer expires and there is no collision, the desired larger prefix is associated with the local domain.

6.4. Releasing Allocated Space

If the lifetime of a prefix allocated to the local domain expires and the domain does not need to reuse it, all associated with this prefix resources are deleted and no further actions are taken. If the lifetime of the prefix has not expired, and if no subranges of that prefix have being allocated for local usage or by some of the children domains, the space may be released by sending a withdraw message to the parent domain and all known siblings of the same domain.

7. Constants

MASC uses the following constants:

[PORT-NUMBER]
: TODO. The TCP port number used to listen for incoming MASC connections.

[WAITING-PERIOD]
: The amount of time that must pass between a new claim, and a PREFIX_IN_USE. This must be long enough to reasonably span any single inter-domain network partition. Default: 172800 seconds (i.e. 48 hours).

[INITIATE-CLAIM-DELAY]
: The amount of time a MASC node must wait before initiating a new claim or claim for space expansion must be a random value in the interval (0, [INITIATE-CLAIM-DELAY]). Default value for [INITIATE-CLAIM-DELAY]: 600 seconds (i.e. 10 minutes).

[TLD-ID]
: The Paret Domain Identifier used by a Top-Level Domain (which has no parent). Must be 0.

[HOLDTIME]
: The amount of time that must pass without any messages received from a remote node before considering the connection is down. Default: 240 seconds (i.e. 4 minutes).

8. Message Formats

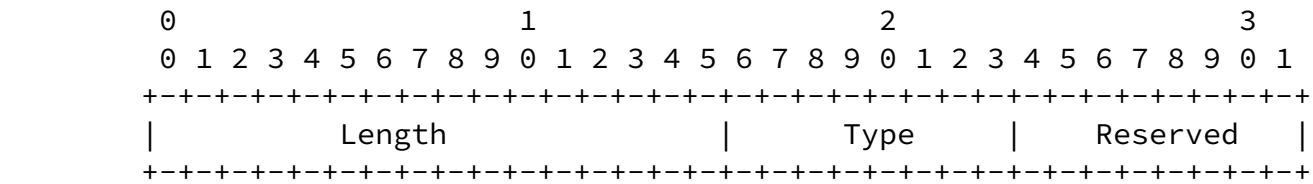
This section describes message formats used by MASC.

Messages are sent over a reliable transport protocol connection. A message is processed only after it is entirely received. The maximum message size is 4096 octets. All implementations are required to support this maximum message size.

All fields labeled "Reserved" below must be transmitted as 0, and ignored upon receipt.

8.1. Message Header Format

Each message has a fixed-size (4-byte) header. There may or may not be a data portion following the header, depending on the message type. The layout of these fields is shown below:



Length:
This 2-octet unsigned integer indicates the total length of the message, including the header, in octets. Thus, e.g., it allows one to locate in the transport-level stream the start of the next message. The value of the Length field must always be at least 4 and no greater than 4096, and may be further constrained, depending on the message type. No "padding" of extra data after the message is allowed, so the Length field must have the smallest value required given the rest of the message.

Type:

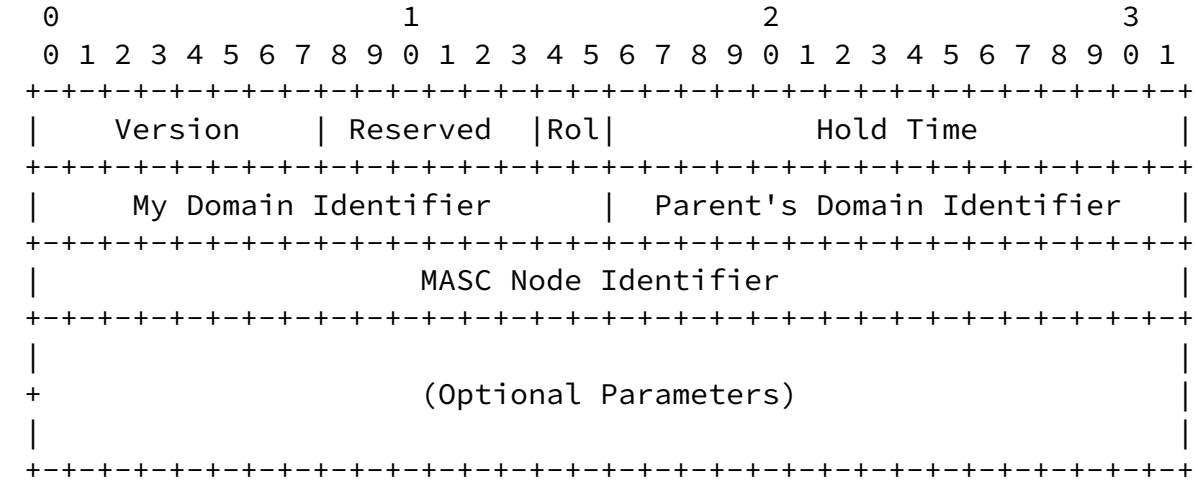
This 1-octet unsigned integer indicates the type code of the message. The following type codes are defined:

1 - OPEN
2 - UPDATE
3 - NOTIFICATION
4 - KEEPALIVE

8.2. OPEN Message Format

After a transport protocol connection is established, the first message sent by each side is an OPEN message. If the OPEN message is acceptable, a KEEPALIVE message confirming the OPEN is sent back. Once the OPEN is confirmed, UPDATE, KEEPALIVE, and NOTIFICATION messages may be exchanged.

The minimum length of the OPEN message is 16 octets (including message header). In addition to the fixed-size MASC header, the OPEN message contains the following fields:



Version:

This 1-octet unsigned integer indicates the protocol version number of the message. The current MASC version number is 1.

Reserved:
Must be zero. Ignored by the receiver.

My Role (Rol):
The proposed relationship of the sending system to the receiving system:
0 = INTERNAL_PEER (sent from one internal peer to another)
1 = CHILD (sent from a child to its parent)
2 = SIBLING (sent from one sibling to another)
3 = PARENT (sent from a parent to its child)

Hold Time:
This 2-octet unsigned integer indicates the number of seconds that the sender proposes for the value of the Hold Timer. Upon receipt

of an OPEN message, a MASC speaker MUST calculate the value of the Hold Timer by using the smaller of its configured Hold Time and the Hold Time received in the OPEN message. The Hold Time MUST be either zero or at least three seconds. An implementation may reject connections on the basis of the Hold Time. The calculated value indicates the maximum number of seconds that may elapse between the receipt of successive KEEPALIVE and/or UPDATE messages by the sender.

My Domain Identifier:
This 2-octet unsigned integer indicates the Autonomous System number of the sender.

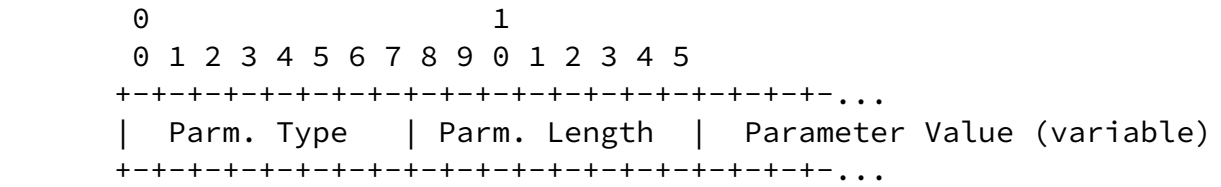
Parent's Domain Identifier:
This 2-octet unsigned integer indicates the Autonomous System number of the sender's parent. It is set to [TLD-ID] if the sender is a TLD. This field is used to determine the parent of a sibling, for use when propagating claims.

MASC Node Identifier:
This 4-octet unsigned integer indicates the MASC Node Identifier of the sender. A given MASC speaker sets the value of its MASC Node

Identifier to a globally-unique value assigned to that MASC speaker (e.g., an IPv4 address). The value of the MASC Node Identifier is determined on startup and is the same for every MASC session opened.

Optional Parameters:

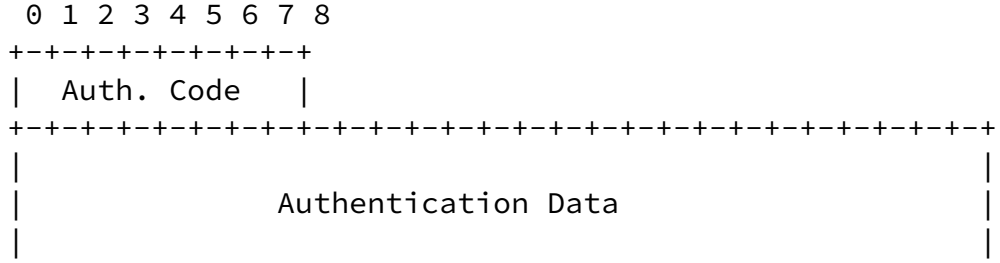
This field may contain a list of optional parameters, where each parameter is encoded as a <Parameter Length, Parameter Type, Parameter Value> triplet. The combined length of all optional parameters can be derived from the Length field in the message header.



Parameter Type is a one octet field that unambiguously identifies individual parameters. Parameter Length is a one octet field that contains the length of the Parameter Value field in octets. Parameter Value is a variable length field that is interpreted according to the value of the Parameter Type field.

This document defines the following Optional Parameters:

a) Authentication Information (Parameter Type 1):
This optional parameter may be used to authenticate a MASC speaker. The Parameter Value field contains a 1-octet Authentication Code followed by a variable length Authentication Data.



Authentication Code:

This 1-octet unsigned integer indicates the authentication mechanism being used. Whenever an authentication mechanism is specified for use within MASC, three things must be included in the specification:

- o the value of the Authentication Code which indicates use of the mechanism,
- o the form and meaning of the Authentication Data, and
- o the algorithm for computing values of Marker fields.

Note that a separate authentication mechanism may be used in establishing the transport level connection.

Authentication Data:

The form and meaning of this field is a variable-length field depend on the Authentication Code.

b) Parents Information (Parameter Type 2):
This optional parameter may be used to carry the Parents' Domain IDs. The Parameter Value field contains a number of 2-octet fields

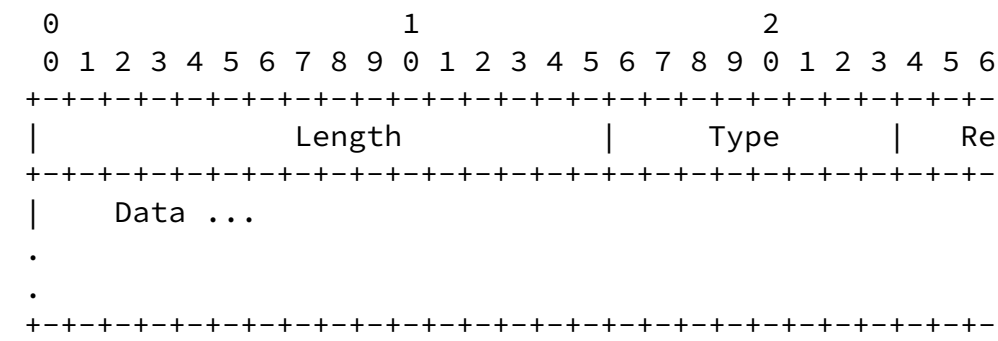
filled with the Domain ID of each MASC parent domain.

8.3. UPDATE Message Format

UPDATE messages are used to transfer Claim/Collision/PrefixManaged information between MASC speakers. The UPDATE message always includes the fixed-size MASC header, and one or more attributes as described below. The minimum length of the UPDATE message is 32 octets (including

the message header).

Each attribute is of the form:



All attributes are 4-byte aligned.

Length:
The Length is the length of the entire attribute, including the length, type, and data fields. If other attributes are nested within the data field, the length includes the size of all such nested attributes.

Type:

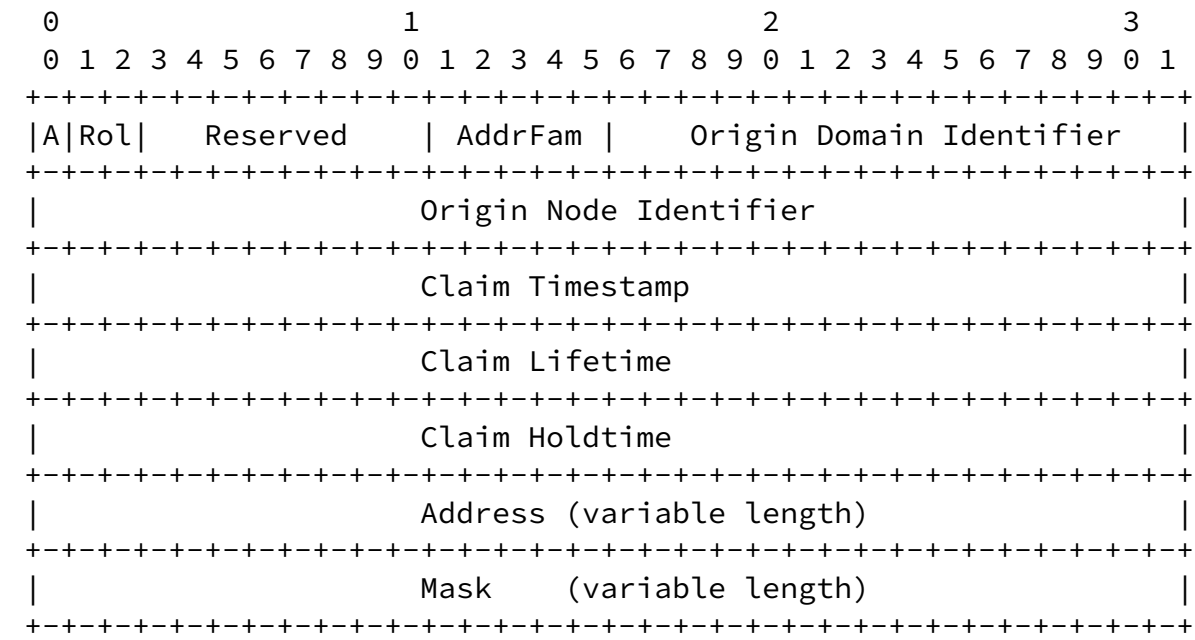
Types 128-255 are reserved for "optional" attributes. If a required attribute is unrecognized, a NOTIFICATION with UPDATE Error Code and Unrecognized Required Attribute subcode will be sent.

Unrecognized optional attributes are simply ignored.

0 = PREFIX_IN_USE (prefix is being used by the origin)
1 = CLAIM_DENIED (origin refuses your claim and will not propagate it)
2 = CLAIM_TO_EXPAND (origin is trying to expand the size of an

existing prefix)
3 = NEW_CLAIM (origin is trying to claim a new prefix)
4 = PREFIX_MANAGED (parent is informing child of space available)
5 = WITHDRAW (origin is withdrawing a previous claim)

Types 0-3 are collectively called "CLAIMs". The message format below describes the encoding of a CLAIM, PREFIX_MANAGED and WITHDRAW.



Reserved/Res.:
Must be zero. Ignored by the receiver.

A-bit:
ACTIVE_PREFIX bit. If set, indicates that the advertised address prefix is Active, otherwise the prefix is Deprecated, i.e. non-active (see Section 5.2).

Rol:
The relationship/role of the Origin of the message to the node sending that message.

0 = INTERNAL (originated by the sender's domain)
1 = CHILD (originated by a child of the sender's domain)
2 = SIBLING (originated by a sibling of the sender's domain)
3 = PARENT (originated by a parent of the sender's domain)

Origin Domain Identifier:
The AS number of the claim originator.

Origin Node Identifier:
The MASC Node ID of the claim originator.

Claim Timestamp:
The timestamp of the claim when it was originated. The timestamp is expressed in number of seconds since midnight (0 hour), January 1, 1970, Greenwich.

Claim Lifetime:
The time in seconds between the Claim Timestamp, and the time at which the prefix will become free.

Claim Holdtime:
The time in seconds between the Claim Timestamp, and the time at which the claim should be deleted from the local cache. For PREFIX_IN_USE and PREFIX_MANAGED claims it should be equal to Claim Lifetime; for CLAIM_TO_EXPAND, NEW_CLAIM, and CLAIM_DENIED it should be equal to [WAITING-PERIOD].

AddrFam:
The IANA-assigned address family number of the encoded prefix [IANA]. These include (among others):

Number	Description
1	IP (IP version 4)
2	IPv6 (IP version 6)

Address:
The address associated with the given prefix to be encoded. The length is determined based on the Address Family (e.g. 4 for IPv4, 16 for IPv6)

Mask:
The mask associated with the given prefix. The length is the same as the Address field and is determined based on the Address Family. The field contains the full bitmask.

8.4. KEEPALIVE Message Format

MASC does not use any transport protocol-based keep-alive mechanism to determine if peers are reachable. Instead, KEEPALIVE messages are exchanged between peers often enough as not to cause the Hold Timer to

expire. A reasonable maximum time between the last KEEPALIVE or UPDATE message sent, and the time at which a KEEPALIVE message is sent, would be one third of the Hold Time interval. KEEPALIVE messages MUST NOT be sent more frequently than one per second. An implementation MAY adjust the rate at which it sends KEEPALIVE messages as a function of the Hold Time interval.

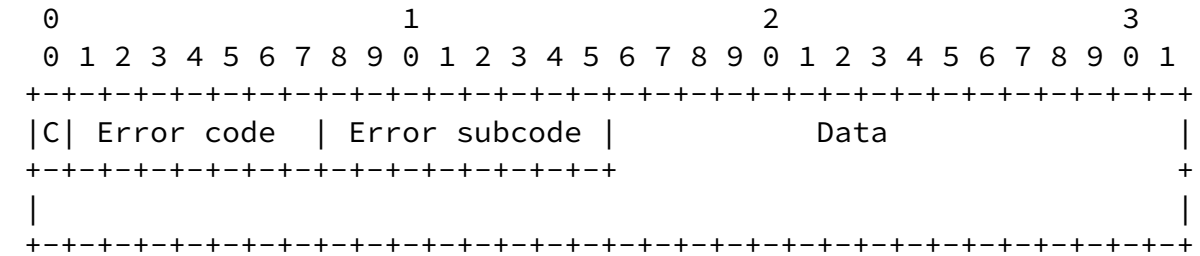
If the negotiated Hold Time interval is zero, then periodic KEEPALIVE messages MUST NOT be sent.

A KEEPALIVE message consists of only a message header, and has a length of 4 octets.

8.5. NOTIFICATION Message Format

A NOTIFICATION message is sent when an error condition is detected. Depending on the error condition, the MASC connection might or must be closed immediately after sending the message. If the connection is to be closed, the C-bit must be set.

In addition to the fixed-size MASC header, the NOTIFICATION message contains the following fields:



C-bit:
Close-bit. If set, the connection will be closed.

Error Code:
This 7-bit unsigned integer indicates the type of NOTIFICATION. The following Error Codes have been defined:

Error Code	Symbolic Name	Reference
1	Message Header Error	Section 9.1
2	OPEN Message Error	Section 9.2

3	CLAIM Message Error	Section 9.3

4	Hold Timer Expired	Section 9.4
5	Finite State Machine Error	Section 9.5
6	NOTIFICATION Message Error	Section 9.6
7	Cease	Section 9.7

Error subcode:

This 1-octet unsigned integer provides more specific information about the nature of the reported error. Each Error Code may have one or more Error Subcodes associated with it. If no appropriate Error Subcode is defined, then a zero (Unspecific) value is used for the Error Subcode field, and the C-bit must be set (i.e. the connection will be closed). The used notation in the error description below is: MC = Must Close connection = C-bit set; CC = Can Close connection = C-bit is not set.

Message Header Error subcodes:

0 - Unspecific	(MC)
1 - Bad Message Length	(MC)
2 - Bad Message Type	(MC)

OPEN Message Error subcodes:

0 - Unspecific	(MC)
1 - Unsupported Version Number	(MC)
2 - Bad Peer AS	(MC)
4 - Unsupported Optional Parameter	(CC)
5 - Authentication Failure	(MC)
6 - Unacceptable Hold Time	(MC)
7 - Invalid Parent Configuration	(MC)
8 - Inconsistent Role	(MC)
9 - Bad Parent Domain ID	(MC)
10 - No Common Parent	(MC)

UPDATE Message Error subcodes:

0 - Unspecific (MC)
1 - Malformed Attribute List (MC)
2 - Unrecognized Required Attribute (CC)
5 - Attribute Length Error (MC)
10 - Invalid Address field (CC)
11 - Invalid Mask field (CC)
12 - Non-Contiguous Mask (CC)

13 - Unrecognized Address Family (MC)
14 - Claim Type Error (CC)
15 - Origin Domain ID Error (CC)
16 - Origin Node ID Error (CC)
17 - Claim Lifetime Too Short (CC)
18 - Claim Lifetime Too Long (CC)
19 - Claim Timestamp Too Old (CC)
20 - Claim Timestamp Too New (CC)
21 - Claim Prefix Size Too Small (CC)
22 - Claim Prefix Size Too Large (CC)
23 - Illegal Origin Role Error (CC)

Hold Timer Expired subcodes (the C-bit is always set):

0 - Unspecific (MC)

Finite State Machine Error subcodes:

0 - Unspecific (MC)
1 - Open/Close MASC Connection FSM Error (MC)

Cease subcodes (the C-bit is always set):

0 - Unspecific (MC)

NOTIFICATION subcodes (the C-bit is always set):

0 - Unspecific (MC)

Data:

This variable-length field is used to diagnose the reason for the NOTIFICATION. The contents of the Data field depend upon the Error Code and Error Subcode. See Section 9 for more details.

Note that the length of the Data field can be determined from the message Length field by the formula:

Message Length = 6 + Data Length

The minimum length of the NOTIFICATION message is 6 octets (including message header).

9. MASC Error Handling

This section describes actions to be taken when errors are detected while processing MASC messages. MASC Error Handling is similar to that of BGP [BGP].

When any of the conditions described here are detected, a NOTIFICATION message with the indicated Error Code, Error Subcode, and Data fields is sent. In addition, the MASC connection might be closed. If no Error Subcode is specified, then a zero (Unspecific) must be used.

The phrase "the MASC connection is closed" means that the transport protocol connection has been closed and that all resources for that MASC connection have been deallocated.

Unless specified explicitly, the Data field of the NOTIFICATION message that is sent to indicate an error is empty.

9.1. Message Header error handling

All errors detected while processing the Message Header are indicated by sending the NOTIFICATION message with Error Code Message Header Error. The Error Subcode elaborates on the specific nature of the error.

If the Length field of the message header is less than 4 or greater than 4096, or if the Length field of an OPEN message is less than the minimum length of the OPEN message, or if the Length field of an UPDATE message is less than the minimum length of the UPDATE message, or if the Length field of a KEEPALIVE message is not equal to 4, then the Error Subcode is set to Bad Message Length. The Data field contains the erroneous Length field.

If the Type field of the message header is not recognized, then the Error Subcode is set to Bad Message Type. The Data field contains the erroneous Type field.

9.2. OPEN message error handling

All errors detected while processing the OPEN message are indicated by sending the NOTIFICATION message with Error Code OPEN Message Error. The Error Subcode elaborates on the specific nature of the error.

If the version number contained in the Version field of the received

OPEN message is not supported, then the Error Subcode is set to Unsupported Version Number. The Data field is a 1-octet unsigned integer, which indicates the largest locally supported version number less than the version the remote MASC node bid (as indicated in the received OPEN message).

If the Autonomous System field of the OPEN message is unacceptable, then the Error Subcode is set to Bad Peer AS. The determination of acceptable Autonomous System numbers is outside the scope of this protocol.

If the Hold Time field of the OPEN message is unacceptable, then the Error Subcode MUST be set to Unacceptable Hold Time. An implementation MUST reject Hold Time values of one or two seconds. An implementation MAY reject any proposed Hold Time. An implementation which accepts a Hold Time MUST use the negotiated value for the Hold Time.

If one of the Optional Parameters in the OPEN message is not recognized, then the Error Subcode is set to Unsupported Optional Parameters.

If the OPEN message carries Authentication Information (as an Optional Parameter), then the corresponding authentication procedure is invoked. If the authentication procedure (based on Authentication Code and Authentication Data) fails, then the Error Subcode is set to Authentication Failure.

If the remote system's proposed Role conflicts with its expected role (based on the local system's configured Role), then the Error Subcode is set to Inconsistent Role. The Data field is 1-octet long, and contains the local system's configured Role.

If the remote system's proposed Role is INTERNAL_PEER, and either (but not both) the local system or the remote system's Parent's Domain ID is [TLD-ID], then the Error Subcode is set to Invalid Parent Configuration. The Data field must be filled with all the local system's Parent Domain IDs.

If one of the remote system's Parent Domain IDs is unacceptable, then the Error Subcode is set to Bad Parent Domain ID and the Data field must be filled with all unacceptable Parent Domain IDs. The determination of acceptable Parent Domain ID is outside the scope of this protocol.

If the remote system is supposed to be a sibling, but it does not have a common parent (based on the Parents Domain ID information in the OPEN message), the Error Subcode is set to No Common Parent, and the Data

field is filled with all Parent's Domain IDs of the local MASC domain.

9.3. UPDATE message error handling

All errors detected while processing the UPDATE message are indicated by sending the NOTIFICATION message with Error Code UPDATE Message Error. The error subcode elaborates on the specific nature of the error.

If any recognized attribute has an Attribute Length that conflicts with the expected length (based on the attribute type code), then the Error Subcode is set to Attribute Length Error. The Data field contains the erroneous attribute (type, length and value).

If the Address field includes an invalid address (except 0), then the Error Subcode is set to Invalid Address. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If the Mask field includes an invalid mask (for example, starting with 0), then the Error Subcode is set to Invalid Mask. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If the Mask field includes a non-contiguous bitmask, and that MASC server does not support, or is not configured to use non-contiguous masks, then the Error Subcode is set to Non-Contiguous Mask. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If the Address Family is unrecognized, then the Error Subcode is set to Unrecognized Address Family. In addition, the Data field must contain the first 28 octets of the UPDATE message (excluding the Message Header).

If the Origin Role/Claim Type combination is not one of the following, then the Error Subcode is set to Claim Type Error ("x" = "any", i.e. 0/1)

A	Origin Role	Claim Type
x	ICSP	PREFIX_IN_USE (0)
0	ICSP	CLAIM_DENIED (1)
0	ICSP	CLAIM_TO_EXPAND (2)
0	ICSP	NEW_CLAIM (3)

x	I SP	PREFIX_MANAGED (4)

In addition, the Data Field must contain the whole UPDATE message (excluding the Message Header).

If there is a reason to believe that the Origin Domain ID is invalid (for example, if it is 0), then the Error Subcode is set to Origin

Domain ID Error. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header). The same applies for Origin Node ID (the corresponding error is Origin Node ID Error).

If a node (usually a parent receiving a claim from a child) thinks that the Claim Lifetime is too short (for example, less than 172800, i.e. 48 hours), it SHOULD send an UPDATE Message Error with subcode Claim Lifetime Too Short. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If a node (usually a parent receiving a claim from a child) thinks the Claim Lifetime is too long (for example, more than 15,768,000, i.e. half year), then it SHOULD send a UPDATE Message Error with subcode Claim Lifetime Too Long. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header). Note that usually a parent MASC node should send first CLAIM_DENIED collision messages with Claim Lifetime field filled with the longest acceptable lifetime. If the child refuses to claim with shorter lifetime, then Claim Lifetime Too Long should be sent.

If a node (usually a parent receiving a claim from a child) thinks the Claim Timestamp is too small, i.e. too old (for example, if a node is self-confident that its clock is quite accurate), then it SHOULD send a UPDATE Message Error with subcode Claim Timestamp Too Old. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If a node (usually a parent receiving a claim from a child) thinks the Claim Timestamp is too large, i.e. too new (for example, if a node is self-confident that its clock is quite accurate), then it SHOULD send a UPDATE Message Error with subcode Claim Timestamp Too New. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If a node (usually a parent receiving a claim from a child) thinks that the prefix size implied by the Mask field is too small (for example, smaller than 16 addresses), then it SHOULD send a UPDATE Message Error with subcode Claim Prefix Size Too Small. In addition, the Data field

must contain the whole UPDATE message (excluding the Message Header).

If a node (usually a parent receiving a claim from a child) thinks that the prefix size implied by the Mask field is too large, then it COULD send a UPDATE Message Error with subcode Claim Prefix Size Too Large. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header). Note that usually a parent MASC node should send first CLAIM_DENIED collision messages for some subrange of the child's large claimed address range. If the child refuses to shrink the claim size, then Claim Prefix Size Too Large should be sent.

If the received UPDATE message's computed Updated Origin Role is illegal (see Table 1), then the Error Subcode is set to Illegal Origin Role Error. In addition, the Data field must contain the whole UPDATE message (excluding the Message Header).

If any other error is encountered when processing attributes, then the Error Subcode is set to Malformed Attribute List, and the problematic attribute is included in the data field.

9.4. Hold Timer Expired error handling

If a system does not receive successive KEEPALIVE and/or UPDATE and/or NOTIFICATION messages within the period specified in the Hold Time field of the OPEN message, then the NOTIFICATION message with Hold Timer Expired Error Code must be sent and the MASC connection closed.

9.5. Finite State Machine error handling

Any error detected by the MASC Finite State Machine (e.g., receipt of an unexpected event) is indicated by sending the NOTIFICATION message with Error Code Finite State Machine Error. The Error Subcode elaborates on the specific nature of the error.

9.6. NOTIFICATION message error handling

If a node sends a NOTIFICATION message, and there is an error in that message, and the C-bit of that message is not set, a NOTIFICATION with C-bit set, Error Code of NOTIFICATION Error, and subcode Unspecific must be sent. In addition, the Data field must include the erratic NOTIFICATION message. However, if the erratic NOTIFICATION message had

the C-bit set, then any error, such as an unrecognized Error Code or Error Subcode, should be noticed, logged locally, and brought to the attention of the administration of the remote node. The means to do this, however, lies outside the scope of this document.

9.7. Cease

In absence of any fatal errors (that are indicated in this section), a MASC node may choose at any given time to close its MASC connection by sending the NOTIFICATION message with Error Code Cease. However, the Cease NOTIFICATION message must not be used when a fatal error indicated by this section does exist.

9.8. Connection Collision Detection

If a pair of MASC speakers try simultaneously to establish a TCP connection to each other, then two parallel connections between this pair of speakers might well be formed. We refer to this situation as connection collision. Clearly, one of these connections must be closed.

Based on the value of the MASC Node Identifier a convention is established for detecting which MASC connection is to be preserved when a connection collision does occur. The convention is to compare the MASC Node Identifiers of the remote nodes involved in the collision and to retain only the connection initiated by the MASC speaker with the higher-valued MASC Node Identifier.

Upon receipt of an OPEN message, the local system must examine all of its connections that are in the OpenConfirm state. A MASC speaker may also examine connections in an OpenSent state if it knows the MASC Node Identifier of the remote node by means outside of the protocol. If among these connections there is a connection to a remote MASC speaker whose MASC Node Identifier equals the one in the OPEN message, then the local system performs the following connection collision resolution procedure:

1. The MASC Node Identifier of the local system is compared to the MASC Node Identifier of the remote system (as specified in the OPEN message).

2. If the value of the local MASC Node Identifier is less than the remote one, the local system closes MASC connection that already exists (the one that is already in the OpenConfirm state), and

accepts MASC connection initiated by the remote system.

3. Otherwise, the local system closes the newly created MASC connection (the one associated with the newly received OPEN message), and continues to use the existing one (the one that is already in the OpenConfirm state).

Comparing MASC Node Identifiers is done by treating them as (4-octet long) unsigned integers.

A connection collision with an existing MASC connection that is in the Established state causes unconditional closing of the newly created connection. Note that a connection collision cannot be detected with connections that are in Idle, or Connect, or Active states (see Section 11).

Closing the MASC connection (that results from the collision resolution procedure) is accomplished by sending the NOTIFICATION message with the Error Code Cease.

10. MASC Version Negotiation

MASC speakers may negotiate the version of the protocol by making multiple attempts to open a MASC connection, starting with the highest version number each supports. If an open attempt fails with an Error Code OPEN Message Error, and an Error Subcode Unsupported Version Number, then the MASC speaker has available the version number it tried, the version number the remote node tried, the version number passed by the remote node in the NOTIFICATION message, and the version numbers that it supports. If the two MASC speakers do support one or more common versions, then this will allow them to rapidly determine the highest common version. In order to support MASC version negotiation, future versions of MASC must retain the format of the OPEN and NOTIFICATION messages.

11. MASC Finite State machine

This section specifies MASC operation in terms of a Finite State Machine (FSM). Following is a brief summary and overview of MASC operations by

state as determined by this FSM.

Initially MASC is in the Idle state.

11.1. Open/Close MASC Connection FSM

Idle state:

In this state MASC refuses all incoming MASC connections. No resources are allocated to the remote node. In response to the Start event (initiated by either system or operator) the local system initializes all MASC resources, starts the ConnectRetry timer, initiates a transport connection to the remote node, while listening for connection that may be initiated by the remote MASC node, and changes its state to Connect. The exact value of the ConnectRetry timer is a local matter, but should be sufficiently large to allow TCP initialization.

If a MASC speaker detects an error, it shuts down the connection and changes its state to Idle. Getting out of the Idle state requires generation of the Start event. If such an event is generated automatically, then persistent MASC errors may result in persistent flapping of the speaker. To avoid such a condition it is recommended that Start events should not be generated immediately for a node that was previously transitioned to Idle due to an error. For a node that was previously transitioned to Idle due to an error, the time between consecutive generation of Start events, if such events are generated automatically, shall exponentially increase. The value of the initial timer shall be 60 seconds. The time shall be doubled for each consecutive retry.

Any other event received in the Idle state is ignored.

Connect state:

In this state MASC is waiting for the transport protocol connection to be completed.

If the transport protocol connection succeeds, the local system

clears the ConnectRetry timer, completes initialization, sends an OPEN message to the remote node, and changes its state to OpenSent. If the transport protocol connect fails (e.g., retransmission timeout), the local system restarts the ConnectRetry timer, continues to listen for a connection that may be initiated by the remote MASC node, and changes its state to Active state.

In response to the ConnectRetry timer expired event, the local system restarts the ConnectRetry timer, initiates a transport connection to other MASC node, continues to listen for a connection that may be

initiated by the remote MASC node, and stays in the Connect state.

The Start event is ignored in the Connect state.

In response to any other event (initiated by either system or operator), the local system releases all MASC resources associated with this connection and changes its state to Idle.

Active state:

In this state MASC is trying to acquire a remote node by initiating a transport protocol connection.

If the transport protocol connection succeeds, the local system clears the ConnectRetry timer, completes initialization, sends an OPEN message to the remote node, sets its Hold Timer to a large value, and changes its state to OpenSent. A Hold Timer value of [HOLDTIME] minutes is suggested.

In response to the ConnectRetry timer expired event, the local system restarts the ConnectRetry timer, initiates a transport connection to other MASC node, continues to listen for a connection that may be initiated by the remote MASC node, and changes its state to Connect.

If the local system detects that a remote node is trying to establish MASC connection to it, and the IP address of the remote node is not an expected one, the local system restarts the ConnectRetry timer, rejects the attempted connection, continues to listen for a connection that may be initiated by the remote MASC node, and stays

in the Active state.

The Start event is ignored in the Active state.

In response to any other event (initiated by either system or operator), the local system releases all MASC resources associated with this connection and changes its state to Idle.

OpenSent state:

In this state MASC waits for an OPEN message from the remote node. When an OPEN message is received, all fields are checked for correctness. If the MASC message header checking or OPEN message checking detects an error (see Section 9.2), or a connection collision (see Section 9.8) the local system sends a NOTIFICATION message and, if the connection is to be closed, it changes its state

to Idle.

If the locally configured role is SIBLING and if there is no common Parent Domain ID between the local and the remote node (among the included in the OPEN message), the local system sends a NOTIFICATION Open Message Error with Error Subcode set to No Common Parent, the connection must be closed, and the state of the local system must be changed to Idle.

If there are no errors in the OPEN message, MASC sends a KEEPALIVE message and sets a KeepAlive timer. The Hold Timer, which was originally set to a large value (see above), is replaced with the negotiated Hold Time value (see Section 8.2). If the negotiated Hold Time value is zero, then the Hold Time timer and KeepAlive timers are not started. If the value of the MASC Domain ID field is the same as the local Autonomous System number, and if the Role field of the OPEN message is set to INTERNAL_PEER, then the connection is an "internal" connection; otherwise, it is "external". Finally, the state is changed to OpenConfirm.

If a disconnect notification is received from the underlying transport protocol, the local system closes the MASC connection, restarts the ConnectRetry timer, while continue listening for

connection that may be initiated by the remote MASC node, and goes into the Active state.

If the Hold Timer expires, the local system sends NOTIFICATION message with error code Hold Timer Expired and changes its state to Idle.

In response to the Stop event (initiated by either system or operator) the local system sends NOTIFICATION message with Error Code Cease and changes its state to Idle.

The Start event is ignored in the OpenSent state.

In response to any other event the local system sends a NOTIFICATION message with Error Code Finite State Machine Error and Error Subcode Open/Close MASC Connection FSM Error, and changes its state to Idle.

Whenever MASC changes its state from OpenSent to Idle, it closes the MASC (and transport-level) connection and releases all resources associated with that connection.

OpenConfirm state:

In this state MASC waits for a KEEPALIVE or NOTIFICATION message.

If the local system receives a KEEPALIVE message, it changes its state to Established.

If the Hold Timer expires before a KEEPALIVE message is received, the local system sends NOTIFICATION message with error code Hold Timer Expired and changes its state to Idle.

If the local system receives a NOTIFICATION message with the C-bit set, it changes its state to Idle.

If the KeepAlive timer expires, the local system sends a KEEPALIVE message and restarts its KeepAlive timer.

If a disconnect notification is received from the underlying transport protocol, the local system changes its state to Idle.

In response to the Stop event (initiated by either system or operator) the local system sends NOTIFICATION message with Error Code Cease and changes its state to Idle.

The Start event is ignored in the OpenConfirm state.

In response to any other event the local system sends a NOTIFICATION message with Error Code Finite State Machine Error and Error Subcode Unspecific, and changes its state to Idle.

Whenever MASC changes its state from OpenConfirm to Idle, it closes the MASC (and transport-level) connection and releases all resources associated with that connection.

Established state:

In the Established state MASC can exchange UPDATE, NOTIFICATION, and KEEPALIVE messages with the remote node.

If the local system receives an UPDATE or KEEPALIVE message, it restarts its Hold Timer, if the negotiated Hold Time value is non-zero.

If the local system receives a NOTIFICATION message, with the C-bit set, it changes its state to Idle.

If the local system receives an UPDATE message and the UPDATE message

error handling procedure (see Section 9.3) detects an error, the local system sends a NOTIFICATION message and, if the C-bit was set, changes its state to Idle.

If a disconnect notification is received from the underlying transport protocol, the local system changes its state to Idle.

If the Hold Timer expires, the local system sends a NOTIFICATION message with Error Code Hold Timer Expired and changes its state to Idle.

If the KeepAlive timer expires, the local system sends a KEEPALIVE message and restarts its KeepAlive timer.

Each time the local system sends a KEEPALIVE or UPDATE message, it restarts its KeepAlive timer, unless the negotiated Hold Time value is zero.

In response to the Stop event (initiated by either system or operator), the local system sends a NOTIFICATION message with Error Code Cease and changes its state to Idle.

The Start event is ignored in the Established state.

After entering the Established state, if the local system has UPDATE messages that are to be sent to the remote node, they must be sent immediately.

In response to any other event, the local system sends NOTIFICATION message with Error Code Finite State Machine Error with the C-bit set and Error Subcode Unspecific, and changes its state to Idle.

Whenever MASC changes its state from Established to Idle, it closes the MASC (and transport-level) connection, releases all resources associated with that connection, and deletes all state derived from that connection.

12. UPDATE Message Processing

The UPDATE message are accepted only when the system is in the Established state.

In the text below, a MASC domain is considered a child of itself with regard to the claims that are related to the address space with local

usage purpose (i.e. to be used by the MAASs within that domain). For example, a NEW_CLAIM initiated by a MASC node to obtain more space for local usage from a prefix managed by that domain will have field Role = CHILD.

If an UPDATE is to be propagated further, it should not be sent back to the node that UPDATE was received from, unless there is an indication that the connection to that node was down and then restored.

If the local system receives an UPDATE message, and there is no indication for error, the following actions are taken:

1. Accept/reject the UPDATE
   The Origin Role field is first compared against the local system's configured Role, according to Table 1, to determine the relationship of the origin to the local system. A result of "---" means that receiving such an UPDATE is illegal and should generate a NOTIFICATION. Any other result is the value to use as the "Updated" Origin Role when propagating the UPDATE to others. (This is analogous to updating a metric upon receiving a route, based on the metric of the link.)

	Locally-Configured Role			
Origin Role	INTERNAL_PEER	CHILD	SIBLING	PARENT
INTERNAL	INTERNAL_PEER	PARENT	SIBLING	CHILD
CHILD	CHILD	SIBLING	---	---
SIBLING	SIBLING	---	SIBLING	CHILD
PARENT	PARENT	---	PARENT	---

Table 1: Updated Origin Role Computation

   If the output from the Updated Origin Role Computation is SIBLING, but the Origin Domain ID is the same as the local MASC domain, the Updated Origin Role is changed to INTERNAL. This is necessary in case a MASC node receives from a parent or sibling its own UPDATEs after reboot, or if because of internal partitioning, the INTERNAL_PEERs are exchanging UPDATEs via other MASC domains (either parent or sibling(s)).

   If Claim Timestamp and Claim Holdtime indicate that the claim has expired (e.g. Timestamp + Claim Holdtime <= CurrentTime), the UPDATE is silently dropped and no further actions are taken.

Each new arrival UPDATE is compared with all claims in the local cache. The following fields are compared, and if all of them are the same, the message is silently rejected and no further actions are taken:

- o A-bit, Role, Type
- o AddrFam
- o Origin Domain Identifier
- o Origin Node Identifier
- o Claim Timestamp
- o Claim Lifetime
- o Claim Holdtime
- o Address
- o Mask

2. PREFIX_IN_USE message processing

- o If the Updated Origin Role is PARENT, the claim is rejected, and a NOTIFICATION with Error Code UPDATE Message Error and Error Subcode Illegal Origin Role should be sent back.
- o If the Updated Origin Role is SIBLING, and the claim collides with some of the local domain's pending claims, the loser claims must not be considered further, and the Claim-Timer of each of them must be canceled. If the received PREFIX_IN_USE claim clashes with and wins over from some of the local domain's allocated prefixes, resolve the clash according to Section 13.1. Finally, the claim must be propagated further to all INTERNAL_PEERs, all MASC nodes from the corresponding parent MASC domain and all known siblings of the same parent domain.
- o If the Updated Origin Role is CHILD, the received claim must be propagated further to all INTERNAL_PEERs and all MASC children domains.
- o If the Updated Origin Role is INTERNAL_PEER, but the Origin Domain ID differs from the local Domain ID, a NOTIFICATION with Error Code

UPDATE Message Error and Error Subcode Illegal Origin Role must be sent back, and the claim is rejected. If the MASC node decides that the local domain does not need anymore that prefix, it must be withdrawn, otherwise, the claim is processed as PREFIX_MANAGED.

3. CLAIM_DENIED message processing

- If the Updated Origin Role is CHILD or SIBLING, the message is rejected, and a NOTIFICATION with Error Code UPDATE Message Error and Error Subcode Illegal Origin Role should be sent back.

- If the Updated Origin Role is INTERNAL_PEER, propagate to all other INTERNAL_PEERs, and all MASC children nodes that have same Domain ID as Origin Domain ID in the received CLAIM_DENIED message.

- If the Updated Origin Role is PARENT, propagate to all other INTERNAL_PEERs. If there is a corresponding pending claim originated by the local MASC domain (i.e. a NEW_CLAIM or CLAIM_TO_EXPAND with same AddrFam, Origin Domain ID, Claim Timestamp, Address and Mask), its Claim-Timer must be cancel and the claim must not be considered further.

4. CLAIM_TO_EXPAND message processing

- If the Updated Origin Role is PARENT, the claim is rejected, and a NOTIFICATION with Error Code UPDATE Message Error and Error Subcode Illegal Origin Role should be sent back.

- If the Updated Origin Role is SIBLING and if the received CLAIM_TO_EXPAND collides with and wins over some of the local domain's pending claims, the loser claims must not be considered further, and the Claim-Timer of the each of them must be cancel. Also, the received claim must be propagated further to all INTERNAL_PEERs, all MASC nodes from the same parent MASC domain and all known siblings of the same parent domain.

- If the Updated Origin Role is CHILD, propagate the claim to all INTERNAL_PEERs. If the claimed prefix is not managed by the local domain, or if the lifetime of the claim is longer than the lifetime of the corresponding prefix managed by the local domain, or if the corresponding prefix managed by the local domain is deprecated, or there is an administratively configured reason to prevent the child from succeeding allocating the claimed prefix, a CLAIM_DENIED

must be send to all MASC children nodes that have same Domain ID as Origin Domain ID in the received CLAIM_TO_EXPAND message. The

CLAIM_DENIED must be the same as the received claim, except Rol=INTERNAL, and Claim Lifetime should be set to the maximum allowed lifetime. Otherwise, propagate the claim to all children as well.

- o If the Updated Origin Role is INTERNAL_PEER, but the Origin Domain ID differs from the local Domain ID, a NOTIFICATION with Error Code UPDATE Message Error and Error Subcode Illegal Origin Role must be sent back, and the claim is rejected. If the MASC node decides that the local domain does not need anymore that pending claim, it MAY be withdrawn. Otherwise, the claim must be propagated to all INTERNAL_PEERs and all MASC nodes from the parent MASC domain that has advertised PREFIX_MANAGED that covers the claimed prefix.

5. NEW_CLAIM message processing

Process like CLAIM_TO_EXPAND.

6. PREFIX_MANAGED message processing.

- o If the Updated Origin Role is SIBLING or CHILD, the message is rejected, and a NOTIFICATION with Error Code UPDATE Message Error and Error Subcode Illegal Origin Role should be sent back.

- o If the Updated Origin Role is PARENT, and if it matches one of the parents' domain ID, the prefix is recorded and can be used by the address allocation algorithm for allocating subranges. Also, the message is propagated to all INTERNAL_PEERs.

- o If the Updated Origin Role is INTERNAL_PEER, the prefix is recorded as allocated to the local domain, propagated to all INTERNAL_PEERs, and can be used for (all items apply):

  a) address ranges/prefixes advertisements to all MASC children and local domain's MAASs;

  b) injection into G-RIB;

c) further expansion by the address allocation algorithm (see Appendix A); If the Updated Origin Role is INTERNAL_PEER and if there is already in the local cache a WITHDRAW message that overlaps with the received PREFIX_MANAGED, the range of that WITHDRAW cannot be used for advertisements to the local domain's MAASs [AAP] and for injection into G-RIB. In the special case when there is an indication that the WITHDRAW has being

originated by the local domain because of a clash, and the range specified in WITHDRAW is a subrange of PREFIX_MANAGED, and the Claim Holdtime of WITHDRAW is shorter than the Claim Holdtime of PREFIX_MANAGED, the WITHDRAW's range should not be withdrawn from G-RIB.

7. WITHDRAW message processing

- o If the Updated Origin Role is CHILD, propagate to all INTERNAL_PEERs and children.

- o If the Updated Origin Role is SIBLING, propagate to all INTERNAL_PEERs, all MASC nodes from the same parent MASC domain and all known siblings of the same parent domain.

- o If the Updated Origin Role is INTERNAL, propagate to all INTERNAL_PEERs, all MASC nodes of the parent domain that manages the corresponding parent's space, all known siblings of that parent domain and all children. If there are overlapping PREFIX_MANAGED or PREFIX_IN_USE originated/owned by the local MASC domain, stop advertising the WITHDRAW range to the MAASs and withdraw that range from the G-RIB database. In the special case when there is an indication that the WITHDRAW has being originated by the local domain because of clash, and the range specified in WITHDRAW is a subrange of PREFIX_MANAGED, and the Claim Holdtime of WITHDRAW is shorter than the Claim Holdtime of PREFIX_MANAGED, the WITHDRAW's range should not be withdrawn from G-RIB.

- o If the Updated Origin Role is PARENT, propagate to all INTERNAL_PEERs and all known siblings of the same parent domain. Finally, originate a WITHDRAW message for each intersection of a

locally owned PREFIX_MANAGED/PREFIX_IN_USE and the received WITHDRAW. The locally originated WITHDRAW message's Claim Holdtime should be equal to the received from the parent's WITHDRAW Claim Holdtime; the Origin Node ID should be the same as the particular PREFIX_MANAGED/PREFIX_IN_USE.

13. Operational Considerations

13.1. Clash Resolving Mechanism

If a MASC node receives a PREFIX_IN_USE claim originated by a sibling and the claim overlaps with some of the local prefixes, the clash must

be resolved. Two MASC domains should not manage overlapping address ranges, unless the domains have an ancestor-descendant (e.g. parent-child) relationship in the MASC hierarchy. Also, two MASC domains should not have locally-allocated overlapping address ranges. The clashed address ranges should not be advertised to the MAASs and allocated to multicast applications/sessions. If a clashed address has being allocated to an application, the application should be informed to stop using that address and switch to a new one.

The G-RIB database must be consistent, such that it does not have ambiguous entries. "Ambiguous G-RIB entries" are those entries that might cause the multicast routing protocol to loop or lose connectivity. In MASC the WITHDRAW message is used to solve this problem. When a clashing PREFIX_IN_USE is received, it is compared (using the function describe in Section 6.1.1) against all prefixes allocated to the local domain. If the local PREFIX_IN_USE is the winner, no further actions are taken. If the local PREFIX_IN_USE is the loser, the clashing address range must be withdrawn by initiating a WITHDRAW message. The message must have Role = INTERNAL, Origin Node ID and Origin Domain ID must be the same as the corresponding local PREFIX_IN_USE message, while Claim Timestamp, Claim Lifetime, Claim Holdtime, Address and Mask must be the same as the received winning PREFIX_IN_USE. The initiated WITHDRAW message must be processed as described in Section 12.7.

If a cached WITHDRAW times out and the local MASC domain owns an

overlapping PREFIX_MANAGED or PREFIX_IN_USE, the overlapping PREFIX ranges can be injected back into the G-RIB database. Similarly, the address ranges that were not advertised to the local domain's MAASs due to the WITHDRAW, can now be advertised again.

In addition to the automatic resolving of clashes, a MASC implementation should support manual resolving of clashes. For example, after a clash is detected, the network administrator should be informed that a clash has occured. The specific manual mechanisms are outside the scope of this protocol.

A MASC node must be configured to operate using either manual or automatic clash resolution mechanisms.

13.2. Changing network providers

If a MASC domain changes a network provider, such that the old provider cannot be used to provide connectivity, any traffic for sessions that are in progress and use that MASC domain as a BGMP Root Domain will not

be able to reach that domain.

If the new network provider is willing to carry the traffic for the old sessions rooted at the customer domain, then it must propagate the customer's old prefixes through the G-RIB. However, at least one MASC node in the customer domain must maintain a TCP connection to one of the old network provider's MASC nodes. Thus, it can continue to "defend" the customer's prefixes, and should continue until the old prefixes' lifetimes expire.

If the new network provider is not willing to propagate the old prefixes, then the customer should remove its prefixes from the G-RIB. If BGMP is in use, the old network provider's domain will automatically become the Root Domain for the customer's old groups due to the lack of a more specific group route. MASC nodes in the customer domain MAY still connect with the old provider's MASC nodes to defend their allocation.

13.3. Debugging

13.3.1. Prefix-to-domain lookup

Use mtrace [MTRACE] to find the BGMP/MASC root domain for a group address chosen from that prefix.

13.3.2. Domain-to-prefix lookup

We can find the address space allocated to a particular MASC domain by directly quering one of the MASC servers within that domain. TODO: How to find the address of one of the MASC nodes within a particular domain? Find some of the BGMP routers there, but how?

14. MASC storage

In general, MASC will be run by a border routers, which, in general do not have stable storage. In this case, MASC must use AAP ([AAP]) to store the important information (the prefixes allocated by the local domain) in the domain's MAASs who should have stable storage. If the MASC/BGMP router has local storage, it should use it instead of AAP. Claims that are in progress do not have to be saved by using AAP.

15. Security Considerations

TODO

16. Open Issues

- MASC port number
- Startup and reading from MAASs, initial delay for waiting, FSM, etc.

17. APPENDIX A: Sample algorithms

DISCLAIMER: This section describes some preliminary suggestions by various people for algorithms which could be used with MASC. They are mentioned here merely to stimulate discussion, and are not to be taken as a specification.

17.1. Start-up rules

17.1.1. Top-Level Domains and Global Space Injectors

All TLDs are siblings of each other and share the same space. Unlike the rest of the hierarchy, TLDs do not have explicit parent(s); instead, flooding the TLD mesh is used to propagate the claims. However, one or few "space injectors" are necessary to replace some of the functions of the MASC parents. These "injectors" initiate the advertisement of the globally available address space to the TLDs. If the prefix 225/8 (for example) were designated as globally allocatable at a given exchange, then the space injectors at that exchange would inject 225/8 as a PREFIX_MANAGED into the TLDs mesh. These advertisements would have a very long lifetime, of the order of at least 6 (TODO) months. The injectors should periodically renew the lifetime of the advertised global space. If for some reason some address range that is part of the global address space should not longer be globally allocated, then the space injectors will:

a) stop re-expanding the lifetime of that address range, AND

b) advertise that address range as non-active for the rest of its lifetime.

The global address space advertised by Global Space Injectors must be chosen by IANA (TODO).

17.1.2. Default initial claim size

The default initial claim size is 256 addresses.

One alternate suggestion was to claim a number of addresses equal to 1/16th the domain's unicast address space. However, this may be problematic since multicast space usage may not mirror unicast usage, and claiming a large amount of multicast space without adequate demand could cause enormous wastage. Instead, better performance results from

demand-driven claims, and hence a fixed initial claim size is suggested

instead, with claims being expanded as needed due to demand.

If the MASC server has been running in the past, and if it has saved information about the demand pattern, that information should be used to decide the default initial claim size.

17.1.3. Default initial claim lifetime

The default initial claim lifetime is 30 (TODO) days.

If there is information available from the past, or if claims from the children MASC domains have been received, the longest appropriate lifetime should be used instead of the default 30 days.

17.2. Claim Size and Prefix Selection Algorithm

TODO

17.2.1. Prefix expansion

TODO

17.2.2. Address space utilization

TODO

17.2.3. Prefix selection after increase of demand

TODO

17.2.4. Prefix selection after decrease of demand

TODO

17.2.5. Lifetime extension algorithm

TODO

18. Authors' Addresses

Deborah Estrin
Computer Science Department/ISI
University of Southern California
Los Angeles, CA 90089
USA
Email: estrin@usc.edu

Ramesh Govindan
USC/ISI
4676 Admiralty Way
Marina Del Rey, CA 90292
USA
Email: govindan@isi.edu

Mark Handley
USC/ISI
c/o MIT LCS
545 Technology Square
Cambridge, MA 02141
USA
Email: mjh@isi.edu

Satish Kumar
Computer Science Department/ISI
University of Southern California
Los Angeles, CA 90089
USA
Email: kkumar@usc.edu

Pavlin Ivanov Radoslavov
Computer Science Department/ISI
University of Southern California
Los Angeles, CA 90089
USA
Email: pavlin@catarina.usc.edu

David Thaler
Microsoft
One Microsoft Way
Redmond, WA 98052
USA
Email: dthaler@microsoft.com

19. References

[AAP]
Handley, M., "Multicast Address Allocation Protocol (AAP)", http://north.east.isi.edu/malloc/aap-01.txt July 1998.

[API]
Finlayson, Ross, "An Abstract API for Multicast Address Allocation", draft-ietf-malloc-api-01.txt, July 1998.

[BGMP]
Thaler, D., Estrin, D. and D. Meyer., "Border Gateway Multicast Protocol (BGMP): Protocol Specification", draft-ietf-idmr-gum-02.txt, March 1998.

[BGP]
Rekhter, Y., and T. Li, "A Border Gateway Protocol 4 (BGP-4)", RFC 1771, March 1995.

[IANA]
Reynolds, J. and J. Postel, "Assigned Numbers", RFC 1700, October 1994.

[MALLOC]
Handley, M., Thaler, D. and D. Estrin, "The Internet Multicast Address Allocation Architecture", draft-handley-malloc-arch-00.txt, December 1997.

[MBGP]
Bates, T., Chandra, R., Katz, D., and Y. Rekhter., "Multiprotocol Extensions for BGP-4", RFC 2283, September 1997.

[MTRACE]
Fenner, W., and S. Casner, "A ''traceroute'' facility for IP Multicast", draft-ietf-idmr-traceroute-ipm-02.txt, November 1997.

[MZAP]
Handley, M, "Multicast-Scope Zone Announcement Protocol", draft-ietf-mboned-mzap-00.txt, December 1997.

[SCOPE]

Meyer, D., "Administratively Scoped IP Multicast", RFC 2365, July 1998.

20. Full Copyright Statement

Table of Contents