Network Working Group
Request for Comments: 2909
Category: Experimental

P. Radoslavov
D. Estrin
R. Govindan
USC/ISI
M. Handley
ACIRI
S. Kumar
USC/ISI
D. Thaler
Microsoft

September 2000

The Multicast Address-Set Claim (MASC) Protocol

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

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 group-specific distribution trees will be locally-rooted, and that traffic will be sent outside the domain only when and where external receivers exist.

Table of Contents

1. Introduction

This document describes MASC, a protocol for inter-domain multicast address set allocation. The MASC protocol (a Layer-3 protocol in the multicast address allocation architecture [MALLOC]) 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 (a Prefix Coordinator in [MALLOC]) 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 group-specific 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.

1.1. Terminology

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 [RFC2119].

Constants used by this protocol are shown as [NAME_OF_CONSTANT], and summarized in Section 6.

1.2. Definitions

This specification uses a number of terms that may not be familiar to the reader. This section defines some of these and refers to other documents for definitions of others.

MAAS (Multicast Address Allocation Server)
A host providing multicast address allocation services to end users (e.g. via MADCAP [MADCAP]).

MASC server
A node running MASC.

Peer
Other MASC speakers a node directly communicates with.

Multicast
IP Multicast, as defined for IPv4 in [RFC1112] and for IPv6 in [RFC2460].

Multicast Address
An IP multicast address or group address, as defined in [RFC1112] and [RFC2373]. An identifier for a group of nodes.

2. Requirements for Inter-Domain Address Allocation

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

- Efficient address space utilization when space is scare, 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.
- Minimize flux in the allocated address sets (e.g. the address sets should be reused when possible).
- 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 [CIDR] 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 MAASs 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 are configured with their parent MASC domain, and parents are configured with their

children domains. At the top, a number of Top-Level Domains are connected in a (sparse) mesh and share the global multicast address space. To improve the robustness, a pair of children of the same parent domain MAY be configured as siblings with regard to that parent.

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.

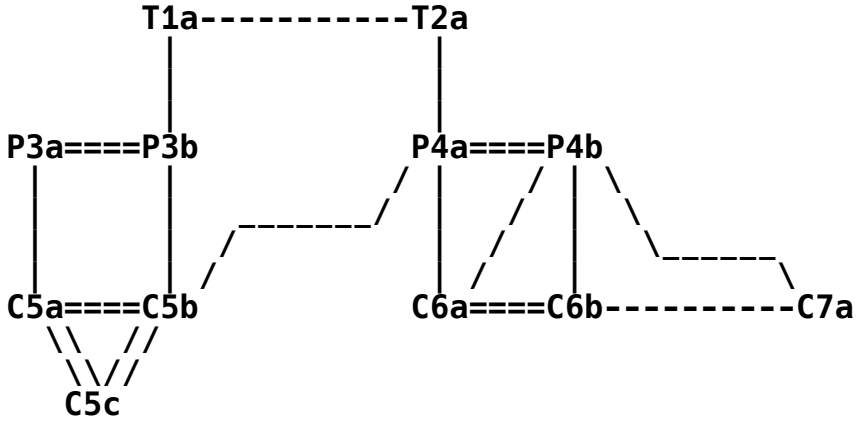


Figure 1: Example MASC Topology

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 (Top-Level Domain SIBLING peering). Also, C6b is a SIBLING in its session with C7a with regard to their common parent P4.

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.

4.1. 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 MAASs 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.

4.2. Prefix Lifetime

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 5.2). 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.

4.3. 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.

4.4. Multi-Parent Sibling-to-Sibling and Internal Peering

Two sibling nodes that have more than one common parent will create and use between them a number of transport-level connections, one per each common parent. The information associated with a parent will be sent over the connection that corresponds to the same parent. Internal peers do not need to open multiple connections between them; a single connection is used for all information.

4.5. Administratively-Scoped Address Allocation

MASC can also be used for sub-allocating prefixes of addresses within an administrative scope zone [SCOPE], but only if the scope is "divisible" (as described in [MALLOC] and [MZAP]). 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.

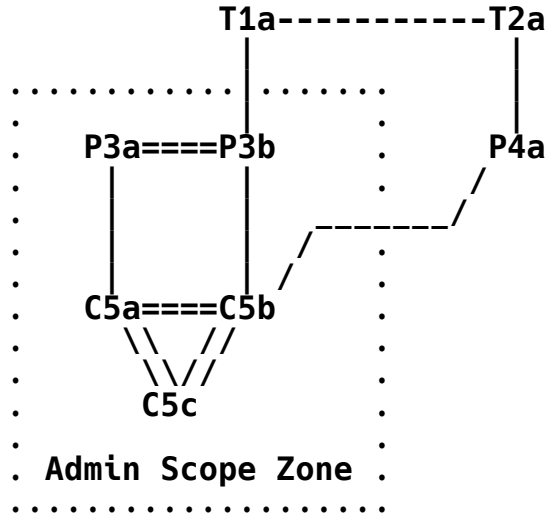


Figure 2: Scope Zone Example

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.

5. Protocol Details

5.1. 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 5.1.1. If the local claim is the loser, a new prefix must be chosen to claim, and the loser claim's Claim-Timer must be canceled. The loser claim can be either explicitly withdrawn, or can be left to expire without taking further actions. 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 can be performed:

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

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

   - 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 (see Section 12.2).

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 5.1.1. The output of the function will define the further processing of that claim (see Section 11).

5.1.1. 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 based on the following precedence:

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 a 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.

5.2. Renewing an Existing Claim

The procedure for extending the lifetime of prefixes already in use is the same as claiming new space (see Section 5.1), except that the claim type must be CLAIM_TO_EXPAND, while the Address and the Mask of the claim (see Section 7.3) must be the same as the already allocated prefix. If the Claim-Timer expires and there is no collision, the desired lifetime is assumed.

5.3. Expanding an Existing Prefix

The procedure for extending the lifetime of prefixes already in use is the same as claiming new space (see Section 5.1), except that the claim type must be CLAIM_TO_EXPAND, while the Address and the Mask of the claim (see Section 7.3) 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.

5.4. 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 resources associated with this prefix 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, all known siblings with the same parent, and all internal peers.

6. Constants

MASC uses the following constants:

[PORT_NUMBER]
: 2587. The TCP port number used to listen for incoming MASC connections, as assigned by IANA.

[WAITING_PERIOD]
: The amount of time (in seconds) that must pass between a NEW_CLAIM (or CLAIM_TO_EXPAND), and a PREFIX_IN_USE for the same prefix. 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 (in seconds) a MASC node must wait before initiating a new claim or a claim for space expansion. This 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 Parent Domain Identifier used by a Top-Level Domain (which has no parent). Must be 0.

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

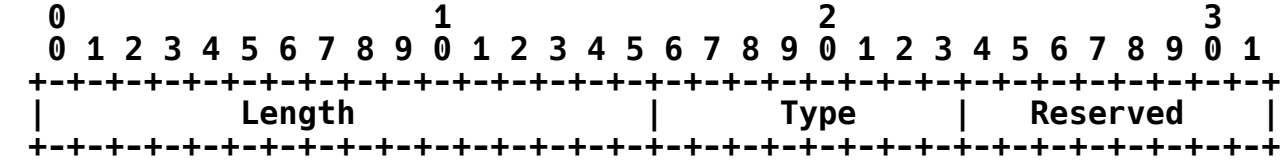
7. Message Formats

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.

7.1. Message Header Format

Each message has a fixed-size (4-octets) 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:

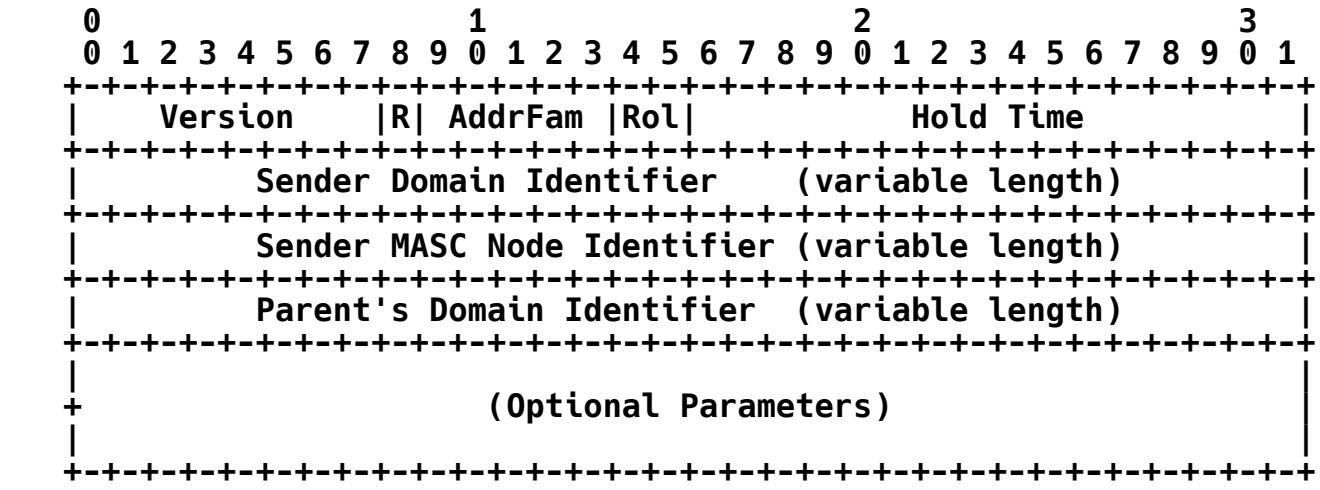
1 - OPEN
2 - UPDATE
3 - NOTIFICATION
4 - KEEPALIVE

Reserved:
This 1-octet field is reserved. MUST be set to zero by the sender, and MUST be ignored by the receiver.

7.2. 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 20 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.

R bit:
This 1-bit field is reserved. MUST be set to zero by the sender, and MUST be ignored by the receiver.

AddrFam:
This 5-bit field is 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)

My Role (Rol):
This 2-bit field indicates the proposed relationship of the sending system to the receiving system:
00 = INTERNAL_PEER (sent from one internal peer to another)
01 = CHILD (sent from a child to its parent)
10 = SIBLING (sent from one sibling to another)
11 = 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 for that peer 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. RECOMMENDED value is [HOLDTIME] seconds.

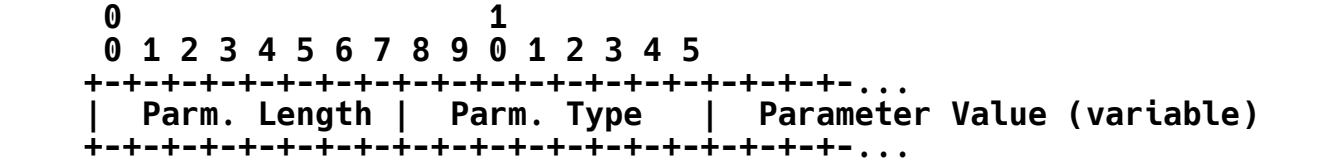
Sender Domain Identifier:
A globally unique identifier. Its length is determined based on the Address Family, and should be treated as an unsigned integer (e.g. a 4-octet integer for IPv4, or a 16-octet integer for IPv6), but must be at least 4 octets long. It should be set to the Autonomous System number of the sender, but the network unicast prefix address is also acceptable.

Sender MASC Node Identifier:
This field's length and format are same as the Sender Domain Identifier field, and 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 or IPv6 address). The value of the MASC Node Identifier is determined on startup and is the same for every MASC session opened.

Parent's Domain Identifier:
This field's length and format are same as the Sender Domain Identifier field, and is set to the Domain Identifier of the sender's parent (e.g. the parent's Autonomous System number, or network prefix address), or is set to [TLD_ID] if the sender is a TLD. Used only when Rol is INTERNAL_PEER or SIBLING, otherwise is ignored. This field is used to determine the common parents between siblings, to associate each sibling-to-sibling connection with a particular parent, and to discover TLD-related

configuration problems among internal peers. If a non-TLD node does not know yet the Domain ID of any of its parents, it can use its own Domain ID in the OPEN messages to its internal peers.

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.



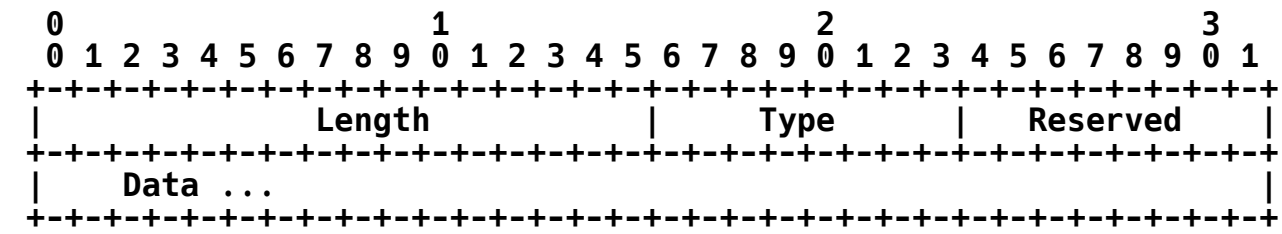
Parameter Length is a one octet field that contains the length of the Parameter Value field in octets. Parameter Type is a one octet field that unambiguously identifies individual parameters. Parameter Value is a variable length field that is interpreted according to the value of the Parameter Type field.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

This document does not define any optional parameters.

7.3. 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 40 octets (including the message header).

Each attribute is of the form:



All attributes are 4-octets 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.

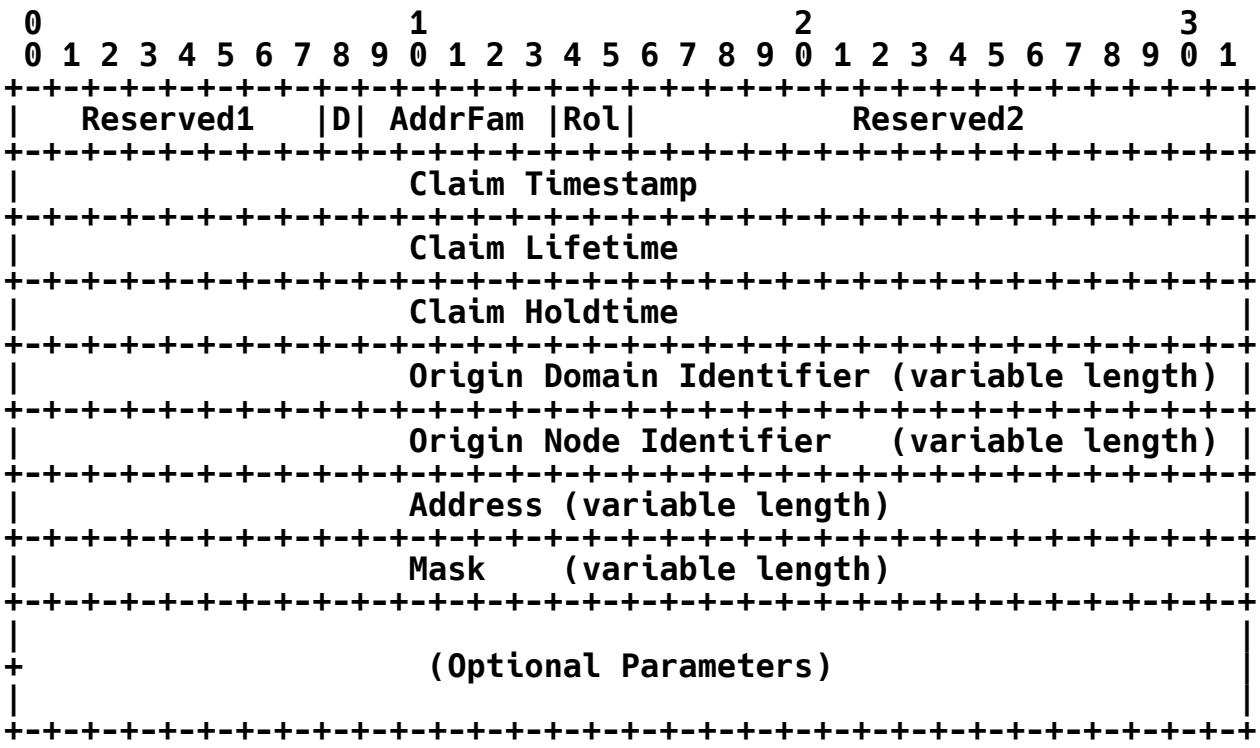
Type:
This 1-octet unsigned integer indicates the type code of the attribute. The following type codes are defined:

0 = PREFIX_IN_USE (prefix is being used by the origin)
1 = CLAIM_DENIED (the claim is refused (probably by the origin's parent domain))
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 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.

Reserved:
This 1-octet field is reserved. MUST be set to zero by the sender, and MUST be ignored by the receiver.

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



Reserved1:
This 1-octet field is reserved. MUST be set to zero by the sender, and MUST be ignored by the receiver.

D-bit:
DEPRECATED_PREFIX bit. If set, indicates that the advertised address prefix is Deprecated, otherwise the prefix is Active (see Section 4.3).

AddrFam:
This 5-bit field is the IANA-assigned address family number of the encoded prefix [IANA].

Rol:
This 2-bit field indicates the relationship/role of the Origin of the message to the node sending that message:

- 00 = INTERNAL (originated by the sender's domain)
- 01 = CHILD (originated by a child of the sender's domain)
- 10 = SIBLING (originated by a sibling of the sender's domain)
- 11 = PARENT (originated by a parent of the sender's domain)

Reserved2:
This 2-octet field is reserved. MUST be set to zero by the sender, and MUST be ignored by the receiver.

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].

Origin Domain Identifier:
The domain identifier of the claim originator. Its length and format definition are same as the Sender Domain Identifier (see Section 7.2).

Origin Node Identifier:
The MASC Node ID of the claim originator. Its length and format definition are same as the Sender MASC Node Identifier (see Section 7.2).

Address:
The address associated with the given prefix to be encoded. The length is determined based on the Address Family (e.g. 4 octets 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.

Optional Parameters:
This field may contain a list of optional parameters, where each parameter is encoded using same format as the optional parameters of an OPEN message (see Section 7.2). Unrecognized optional parameters MUST be silently ignored. This document does not define any optional parameters.

7.4. 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.

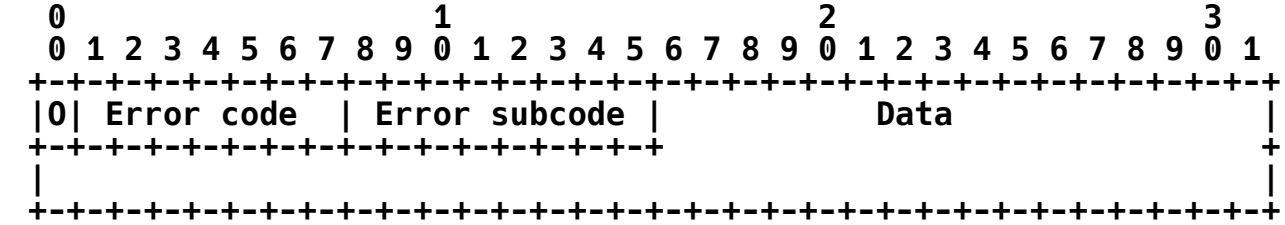
If the negotiated Hold Time interval is zero, then periodic KEEPALIVE messages MUST NOT be sent.

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

7.5. 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 sender of the NOTIFICATION decides that the connection is to be closed, it will indicate this by zeroing the O-bit in the NOTIFICATION message (see below).

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



O-bit:
Open-bit. If zero, it indicates that the sender will close the connection. If '1', it indicates that the sender has chosen to keep the connection open.

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 8.1
2	OPEN Message Error	Section 8.2
3	UPDATE Message Error	Section 8.3
4	Hold Timer Expired	Section 8.4
5	Finite State Machine Error	Section 8.5
6	NOTIFICATION Message Error	Section 8.6
7	Cease	Section 8.7

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 O-bit must be zero (i.e. the connection will be closed). The notation used in the error description below is: MC = Must Close connection = O-bit is zero; CC = Can Close connection = O-bit might be zero.

Message Header Error subcodes:

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

OPEN Message Error subcodes:

0 - Unspecific	(MC)
1 - Unsupported Version Number	(MC)
2 - Bad Peer Domain ID	(MC)
3 - Bad Peer MASC Node ID	(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)
13 - Unrecognized Address Family	(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)
24	No Appropriate Parent Prefix	(CC)
25	No Appropriate Child Prefix	(CC)
26	No Appropriate Internal Prefix	(CC)
27	No Appropriate Sibling Prefix	(CC)
28	Claim Holdtime Too Short	(CC)
29	Claim Holdtime Too Long	(CC)

Hold Timer Expired subcodes (the O-bit is always zero):

0	Unspecific	(MC)

Finite State Machine Error subcodes:

0	Unspecific	(MC)
1	Open/Close MASC Connection FSM Error	(MC)
2	Unexpected Message Type FSM Error	(MC)

Cease subcodes (the O-bit is always zero):

0	Unspecific	(MC)

NOTIFICATION subcodes (the O-bit is always zero):

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 8 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).

8. 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 is empty.

8.1. 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. The Data field contains the erroneous Message (including the message header).

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

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

8.2. 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. The Data field contains the erroneous OPEN Message (excluding the Message Header), unless stated otherwise.

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 Sender Domain Identifier field of the OPEN message is unacceptable, then the Error Subcode is set to Bad Peer Domain ID. The determination of acceptable Domain IDs is outside the scope of this protocol.

If the Sender MASC Node Identifier field of the OPEN message is unacceptable, then the Error Subcode is set to Bad Peer MASC Node ID. The determination of acceptable Node IDs 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 the remote system's proposed Role is INTERNAL_PEER, and either (but not both) the local system or the remote system's Parent 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 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 Parent Domain ID is unacceptable, then the Error Subcode is set to Bad Parent Domain ID, and the Data field is filled with the erroneous Parent Domain ID. 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 with the local system (based on the Parent 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 Domain IDs of the local MASC domain.

If the Address Family is unrecognized, then the Error Subcode is set to Unrecognized Address Family.

8.3. 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. The Data field contains the erroneous UPDATE Message (including the attribute header, but excluding the Message Header), unless stated otherwise.

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.

If any of the mandatory well-known attributes are not recognized, then the Error Subcode is set to Unrecognized Required Attribute.

If the Address field includes an invalid address (except 0), then the Error Subcode is set to Invalid Address.

If the Mask field includes an invalid mask (for example, starting with 0), then the Error Subcode is set to Invalid Mask.

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.

If the Address Family is unrecognized, then the Error Subcode is set to Unrecognized Address Family.

If the Origin Role/Claim Type combination is not one of the following, then the Error Subcode is set to Claim Type Error.

Origin Role	Claim Type	
ICS	PREFIX_IN_USE	(0)
I P	CLAIM_DENIED	(1)
ICS	CLAIM_TO_EXPAND	(2)
ICS	NEW_CLAIM	(3)
I P	PREFIX_MANAGED	(4)
ICSP	WITHDRAW	(5)

If there is a reason to believe that the Origin Domain ID is invalid, then the Error Subcode is set to Origin Domain ID Error. 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) decides that the Claim Lifetime is too short (for example, less than 172800, i.e. 48 hours), it MAY send an UPDATE Message Error with subcode Claim Lifetime Too Short.

If a node (usually a parent receiving a claim from a child) decides that the Claim Lifetime is too long (for example, more than 15,768,000, i.e. half year), then it MAY send an UPDATE Message Error with subcode Claim Lifetime Too Long. 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) decides that 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 MUST send an UPDATE Message Error with subcode Claim Timestamp Too Old. Claim Timestamp Too New is defined similarly.

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

If a node (usually a parent receiving a claim from a child) decides that the prefix size implied by the Mask field is too large, then it MAY send an UPDATE Message Error with subcode Claim Prefix Size Too Large. 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 in Section 11.1), then the Error Subcode is set to Illegal Origin Role Error.

If the received UPDATE message needs to be associated with a parent's prefix, but the association is not successful, then the Error Subcode is set to No Appropriate Parent Prefix. The No Appropriate Child Prefix, No Appropriate Internal Prefix, and No Appropriate Sibling Prefix Error Subcodes are defined similarly.

If a node decides that the Claim Holdtime is too short (for example, just few seconds), it MAY send an UPDATE Message Error with subcode Claim Holdtime Too Short.

If a node decides that the Claim Holdtime is too long (for example, more than 15,768,000, i.e. half year), then it SHOULD send an UPDATE Message Error with subcode Claim Holdtime Too Long.

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

8.4. 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.

8.5. 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.

8.6. NOTIFICATION Message Error Handling

If a node sends a NOTIFICATION message, and there is an error in that message, and the O-bit of that message is not zero, a NOTIFICATION with O-bit zeroed, 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 O-bit zeroed, then any error, such as an unrecognized Error Code or Error Subcode, should be noticed, logged

locally, and brought to the attention of the administrator of the remote node. The means to do this, however, lies outside the scope of this document.

8.7. 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.

8.8. 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. Note that if the nodes were siblings, and each of those connections was associated with a different parent, then we do not consider this situation as collision (see Section 4.4).

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, and, in case of a sibling-to-sibling connection, the Parent Domain ID of that connection 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). Comparing MASC Node Identifiers is done by treating them as unsigned integers (e.g. 4-octets long for IPv4 and 16-octets long for IPv6).

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 the 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).

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 10).

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

9. 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.

10. MASC Finite State Machine

This section specifies MASC operation in terms of a Finite State Machine (FSM). The FSM and the operations are peer peering session. Following is a brief summary and overview of MASC operations by state as determined by this FSM.

Initially the peering session is in the Idle state.

10.1. Open/Close MASC Connection FSM

Idle state:

In this state MASC refuses all incoming MASC connections from the peer. 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 a 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, but shall not be longer than 24 hours.

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 the 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 listening for a transport protocol connection initiated by the remote node.

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] seconds 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 a 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 8.2), or a connection

collision (see Section 8.8) 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 there is no parent domain with Domain ID equal to the Parent Domain ID 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 7.2). 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 MASC Domain ID, 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 a 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 a 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 a NOTIFICATION message with error code Hold Timer Expired and changes its state to Idle.

If the local system receives a NOTIFICATION message with the O-bit zeroed, 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 a 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, or NOTIFICATION message with O-bit set, it restarts its Hold Timer, if the negotiated Hold Time value is non-zero.

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

If the local system receives an UPDATE message and the UPDATE message error handling procedure (see Section 8.3) detects an error, the local system sends a NOTIFICATION message and, if the O-bit was zeroed, 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 (see Section 11.8).

In response to any other event, the local system sends a NOTIFICATION message with Error Code Finite State Machine Error with the O-bit zeroed 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.

11. 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, it checks whether to accept or reject the message, and if it is not rejected, the UPDATE is processed based on its type.

If an UPDATE message must be associated with a parent domain, then there must be a PREFIX_MANAGED by some parent domain for a prefix that covers the prefix of the particular UPDATE.

11.1. Accept/Reject an 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, where Locally-Configured Role is the local configuration with regard to the peer-forwarder of the message. 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

After the Origin Role is updated, the following additional processing needs to be applied:

- 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 both Locally-Configured Role, and Origin Role are equal to PARENT, and the Origin Domain ID is the same as the local MASC domain, the Updated Origin Role is changed to INTERNAL. This is necessary to allow a parent to receive its own UPDATEs through its own children, although the parent might drop those UPDATEs if it has a reason not to believe its children.

- If both Locally-Configured Role, and Origin Role are equal to PARENT, and the Origin Domain ID is the same as the remote MASC domain, and the UPDATE type is CLAIM_DENIED, the Updated Origin Role is changed to INTERNAL. This is necessary to allow a parent to receive the CLAIM_DENIED it has originated through the child whose claim was denied. If the Origin Domain ID is not same as the remote MASC domain, but is same as some of the other MASC children domains, the Updated Origin Role still should be changed to INTERNAL, although the parent might drop this UPDATE if it has a reason not to believe a third party child.

If the Updated Origin Role is INTERNAL, but the Origin Domain ID differs from the local Domain ID, a NOTIFICATION of <UPDATE Message Error, Illegal Origin Role> must be sent back, and the claim is rejected.

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:

- Role, D-bit, Type
- AddrFam
- Claim Timestamp
- Claim Lifetime
- Claim Holdtime
- Origin Domain Identifier

- Origin Node Identifier
- Address
- Mask

Further processing of an UPDATE is based on its type and the Updated Origin Role.

11.2. PREFIX_IN_USE Message Processing

11.2.1. PREFIX_IN_USE by PARENT

The claim is rejected, and a NOTIFICATION of <UPDATE Message Error, Illegal Origin Role> should be sent back.

11.2.2. PREFIX_IN_USE by SIBLING

If the claim cannot be associated with any parent's PREFIX_MANAGED, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken.

If the claim collides with some of the local domain's pending claims, the local 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 some of the local domain's allocated prefixes, resolve the clash according to Section 12.4. Finally, the claim must be propagated further to all INTERNAL_PEERs, all MASC nodes from the corresponding parent MASC domain and all known siblings with the same parent domain.

11.2.3. PREFIX_IN_USE by CHILD

If the claim's prefix is not a subrange of any of the local domain's PREFIX_MANAGED, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken. Otherwise, the claim must be propagated further to all INTERNAL_PEERs and all MASC children domains.

11.2.4. PREFIX_IN_USE by INTERNAL_PEER

If the MASC node decides that the local domain does not need that prefix any more, it may be withdrawn, otherwise, the claim is processed as PREFIX_MANAGED.

11.3. CLAIM_DENIED Message Processing

11.3.1. CLAIM_DENIED by CHILD or SIBLING

The message is rejected, and a NOTIFICATION of <UPDATE Message Error, Illegal Origin Role> should be sent back.

11.3.2. CLAIM_DENIED by INTERNAL_PEER

Propagate to all INTERNAL_PEERs and all MASC children nodes.

11.3.3. CLAIM_DENIED by PARENT

If the Origin Domain ID is not same as the local domain ID, and the UPDATE cannot be associated with any parent domain, the message is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken.

If the Origin Domain ID is not same as the local domain ID, and the UPDATE can be associated with a parent domain, the message is propagated to all nodes from that parent domain, all INTERNAL_PEERs, and all known SIBLINGs with regard to that parent.

If the Origin Domain ID is same as the local domain ID, and there is no 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), a NOTIFICATION of <UPDATE Message Error, No Appropriate Internal Prefix> must be sent back and no further actions should be taken. Otherwise, the matching NEW_CLAIM or CLAIM_TO_EXPAND's Claim-Timer must be canceled and the claim must not be considered further. Finally, the received CLAIM_DENIED must be propagated to all INTERNAL_PEERs, all MASC nodes from the corresponding parent MASC domain, and all known SIBLINGs with regard to that parent.

11.4. CLAIM_TO_EXPAND Message Processing

11.4.1. CLAIM_TO_EXPAND by PARENT

The claim is rejected, and a NOTIFICATION of <UPDATE Message Error, Illegal Origin Role> should be sent back.

11.4.2. CLAIM_TO_EXPAND by SIBLING

If the claim cannot be associated with any parent's PREFIX_MANAGED, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken.

If there is no overlapping PREFIX_IN_USE by the same MASC domain, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Sibling Prefix> must be sent back and no further actions should be taken.

If the claim 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 canceled. Also, the received claim must be propagated further to all INTERNAL_PEERs, all MASC nodes from the corresponding parent MASC domain and all known siblings with the same parent domain.

11.4.3. CLAIM_TO_EXPAND by CHILD

If the claim cannot be associated with any of the local domain's PREFIX_MANAGED, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken.

If there is no overlapping PREFIX_IN_USE by the same MASC domain, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Child Prefix> must be sent back and no further actions should be taken.

Otherwise, the claim has to be propagated to all INTERNAL_PEERs. If the lifetime of the claim is longer than the lifetime of the corresponding prefix managed by the local domain, or if there is an administratively configured reason to prevent the child from succeeding allocating the claimed prefix, a CLAIM_DENIED must be sent to all MASC children nodes that have same Domain ID as Origin Domain ID in the received message. The CLAIM_DENIED must be the same as the received claim, except RoI=INTERNAL, and Claim Lifetime should be set to the maximum allowed lifetime. Otherwise, propagate the claim to all children as well.

11.4.4. CLAIM_TO_EXPAND by INTERNAL_PEER

If the claim cannot be associated with any parent's PREFIX_MANAGED, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further action should be taken.

If there is no overlapping PREFIX_IN_USE by the local MASC domain, the claim is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Internal Prefix> must be sent back and no further actions should be taken.

If the MASC node decides that the local domain does not need that pending claim any more, it MAY be withdrawn. Otherwise, the claim must be propagated to all INTERNAL_PEERs and all MASC nodes from the corresponding parent MASC domain.

11.5. NEW_CLAIM Message Processing

If the claim's Address field is 0 (i.e. a hint by a child to a parent to obtain more space), the claim should be propagated only among the nodes that belong to the child Origin Domain and the parent domain.

Otherwise, process like CLAIM_TO_EXPAND, except that no check for overlapping PREFIX_IN_USE needs to be performed.

11.6. PREFIX_MANAGED Message Processing.

11.6.1. PREFIX_MANAGED by PARENT

If the Origin Domain ID matches one of the parents' domain ID's, the prefix is recorded, and can be used by the address allocation algorithm for allocating subranges. Also, the message is propagated to all MASC nodes of the corresponding parent domain, all INTERNAL_PEERs, and SIBLINGs with same parent.

11.6.2. PREFIX_MANAGED by CHILD or SIBLING

The message is rejected, and a NOTIFICATION of <UPDATE Message Error, Illegal Origin Role> should be sent back.

11.6.3. PREFIX_MANAGED by 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);

11.7. WITHDRAW Message Processing

11.7.1. WITHDRAW by CHILD

If the WITHDRAW cannot be associated with any of the child domain's PREFIX_IN_USE (i.e. no child's PREFIX_IN_USE covers WITHDRAW's range), or if the WITHDRAW does not match any of the child domain's NEW_CLAIM or CLAIM_TO_EXPAND (i.e. there is no child's claim with same Address, Mask and Timestamp), the message is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Child Prefix> must be sent back and no further actions should be taken. Otherwise, propagate to all INTERNAL_PEERs and children.

11.7.2. WITHDRAW by SIBLING

If the WITHDRAW cannot be associated with any of the siblings' PREFIX_IN_USE (i.e. no sibling's PREFIX_IN_USE covers WITHDRAW's range), or if the WITHDRAW does not match any of the sibling domain's NEW_CLAIM or CLAIM_TO_EXPAND (i.e. there is no sibling's claim with same Address, Mask and Timestamp), the message is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Sibling Prefix> must be sent back and no further actions should be taken. Otherwise, propagate to all INTERNAL_PEERs, all MASC nodes from the same parent MASC domain and all known siblings with the same parent domain.

11.7.3. WITHDRAW by INTERNAL

If the WITHDRAW cannot be associated with any of the local domain's PREFIX_IN_USE or PREFIX_MANAGED (i.e. no local domain's prefix covers WITHDRAW's range), or if the WITHDRAW does not match any of the local domain's NEW_CLAIM or CLAIM_TO_EXPAND (i.e. there is no local domain's claim with same Address, Mask and Timestamp) the message is dropped, a NOTIFICATION of <UPDATE Message Error, No Appropriate Internal Prefix> must be sent back and no further actions should be taken.

Otherwise, propagate to all INTERNAL_PEERs, all MASC nodes of the corresponding parent domain of that prefix, all known siblings with that parent domain, and all children. If the WITHDRAW can be associated with some of local domain's PREFIX_IN_USE or PREFIX_MANAGED, 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 been originated by the local domain because of a clash, and the range specified in WITHDRAW is a subrange of the local 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 the G-RIB. If the WITHDRAW matches a local domain's NEW_CLAIM or CLAIM_TO_EXPAND, cancel the matching claim's Claim-Timer.

11.7.4. WITHDRAW by PARENT

If the WITHDRAW cannot be associated with any parent domain, a NOTIFICATION of <UPDATE Message Error, No Appropriate Parent Prefix> must be sent back and no further actions should be taken.

Otherwise, propagate to all INTERNAL_PEERs and all known siblings with the same parent domain. Also, 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 at least equal to the Claim Holdtime in the WITHDRAW message received from the parent; the Origin Node ID should be the same as the particular PREFIX_MANAGED/PREFIX_IN_USE.

11.8. UPDATE Message Ordering

To simplify consistency and sanity check implementations, if there is more than one UPDATE message that needs to be send to a peer (for example, after a connection (re)establishment), some of the UPDATEs must be sent before others.

The rules that always apply are:

- PREFIX_IN_USE must always be sent BEFORE CLAIM_TO_EXPAND, NEW_CLAIM, and WITHDRAW by the same MASC domain

- WITHDRAW must always be sent AFTER PREFIX_IN_USE, CLAIM_TO_EXPAND, NEW_CLAIM, and PREFIX_MANAGED by the same MASC domain

Any further ordering is defined below by the roles of the sender and the receiver.

11.8.1. Parent to Child

Messages are sent in the following order:

1) Parent's PREFIX_MANAGED and WITHDRAWs.

2) All children's PREFIX_IN_USE, CLAIM_TO_EXPAND, and NEW_CLAIMs. CLAIMs from third party children that are hints for more space (i.e. address = 0) should not be propagated; if propagated, the child should drop them.

3) Parent initiated CLAIM_DENIED and children initiated WITHDRAWs. CLAIM_DENIED regarding third party children's claims/hints with address = 0 should not be propagated; if propagated, the child should drop them.

11.8.2. Child to Parent

Messages are sent in the following order:

1) Parent's PREFIX_MANAGED and WITHDRAWs.

2) All PREFIX_IN_USE, CLAIM_TO_EXPAND, and NEW_CLAIMSs from that parent's space, initiated by that child and all its siblings.

3) Parent's initiated CLAIM_DENIED, and all WITHDRAWSs that can be associated with that parent's space and are initiated by the local domain or all known siblings with that parent.

11.8.3. Sibling to Sibling

Messages are sent in the following order:

1) All common parent's PREFIX_MANAGED and WITHDRAWs.

2) PREFIX_IN_USE, CLAIM_TO_EXPAND, and NEW_CLAIMs, initiated by siblings.

3) CLAIM_DENIEDs initiated by common parent, and WITHDRAWs initiated by local domain and all known siblings with that parent.

11.8.4. Internal to Internal

Messages are sent in the following order:

1) All parents' PREFIX_MANAGED and WITHDRAWs.

2) Local domain's and all siblings' PREFIX_IN_USE, CLAIM_TO_EXPAND, and NEW_CLAIMs. CLAIMs from siblings that are hints for more space (i.e. address = 0) should not be propagated; if propagated, the recipient should drop them.

3) CLAIM_DENIEDs initiated by all parents, and WITHDRAWs initiated by local domain and all known siblings.

4) All children's PREFIX_IN_USE, CLAIM_TO_EXPAND, and NEW_CLAIMs.

5) All local domain initiated CLAIM_DENIED regarding children claims and all children initiated WITHDRAWs.

12. Operational Considerations

12.1. Bootup Operations

To learn about its parent domains' IDs and prefixes, a MASC node SHOULD try to establish connections to its PARENT nodes before initiating a connection to a SIBLING node. To avoid learning about its own PREFIX_MANAGED from its children or siblings, a MASC node SHOULD try to establish connections to its PARENT nodes and INTERNAL_PEER nodes before initiating a connection to a CHILD or SIBLING node.

12.2. Leaf and Non-leaf MASC Domain Operation

A non-leaf MASC domain (i.e. a domain that has children domains) should advertise its PREFIX_MANAGED addresses to its children, and should claim from that space the sub-ranges that would be advertised to the internal MAASs (the claim wait time SHOULD be equal to [WAITING_PERIOD]). A MASC node that belongs to a non-leaf MASC domain should perform dual functions by being a child of itself with regard to the claiming and management of the sub-ranges for local usage. A leaf MASC domain should advertise all PREFIX_MANAGED addresses to its MAASs without explicitly claiming them for internal usage. A MASC node can assume that it belongs to a leaf domain if it simply does not have any UPDATEs by children domains. If an UPDATE by a child is received, the domain MUST switch from "leaf" to "non-leaf" mode, and if it needs more addresses for internal usage, it MUST claim them from that domain's PREFIX_MANAGED. After the last UPDATE originated by a child expires, the domain can switch back to "leaf" mode.

12.3. Clock Skew Workaround

Each UPDATE has "Claim Timestamp" field that is set to the absolute time of the MASC node that originated that UPDATE. The timestamp is used for two purposes: to resolve collisions, and to define how long an UPDATE should be kept in the local cache of other MASC nodes. A skew in the clock could result in unfair collision decision such that the claims originated by nodes that have their clock behind the real time will always win; however, because collisions are presumably rare, this will not be an issue. Skew in the clock however might result in expiring an UPDATE earlier than it really should be expired, and a node might assume too early that the expired UPDATE/prefix is free for allocation. To compensate for the clock skew, an UPDATE message should be kept longer than the amount of time specified in the Claim Holdtime. For example, keeping UPDATEs for an additional 24 hours will compensate for clock skew for up to 24 hours.

12.4. 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 5.1.1) 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 11.7.

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 occurred. 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.

12.5. 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 the root of multicast distribution trees 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.

12.6. Debugging

12.6.1. Prefix-to-Domain Lookup

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

12.6.2. Domain-to-Prefix Lookup

We can find the address space allocated to a particular MASC domain by directly querying one of the MASC servers within that domain, by observing the state in parents, siblings, or children MASC domains, or by observing the G-RIB information originated by that domain. From those three methods, the first method can provide the most detailed information. Finding the address of one of the MASC nodes within a particular domain is outside the scope of MASC.

13. 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 the Layer 2 protocol/mechanism (e.g., ([AAP]) as described in [MALLOC] 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 speaker has local storage, it should use it instead of the Layer 2 protocol/mechanism. Claims that are in progress do not have to be saved by using the Layer 2 protocol/mechanism.

14. Security Considerations

IPsec [IPSEC] can be used to address security concerns between two MASC peering nodes. However, because of the store-and-forward nature of the UPDATE messages, it is possible that if a non-trustworthy MASC node can connect to some point of the MASC topology, then this node can undetectably inject malicious UPDATEs that may disturb the normal operation of other MASC nodes. To address this problem, each MASC node should allow peering only with trustworthy nodes.

After a reboot, a MASC node/domain can restore its state from its neighbors (internal peers, parents, siblings, children). Typically, the state received from a parent or internal peer will be trustworthy, but a node may choose to drop its own UPDATEs that were received through a sibling or a child.

A misbehaving node may attempt a Denial of Service attack by sending a large number of colliding messages that would prevent any of its siblings from allocating more addresses. A single mis-behaving node can easily be identified by all of its siblings, and all of its UPDATEs can be ignored. A Denial of Service attack that uses multiple origin addresses can be prevented if a third-party UPDATE (e.g. by a non-directly connected sibling) is accepted only if it is sent via the common parent domain, and the MASC nodes in the parent domain accept children UPDATEs only if they come via an internal peer, or come directly from a child node that is same as the Origin Node ID.

15. IANA Considerations

This document defines several number spaces (MASC message types, MASC OPEN message optional parameters types, MASC UPDATE message attribute types, MASC UPDATE message optional parameters types, and MASC NOTIFICATION message error codes and subcodes). For all of these number spaces, certain values are defined in this specification. New values may only be defined by IETF Consensus, as described in [IANA-CONSIDERATIONS]. Basically, this means that they are defined by RFCs approved by the IESG.

16. Acknowledgments

The authors would like to thank the participants of the IETF for their assistance with this protocol.

17. APPENDIX A: Sample Algorithms

DISCLAIMER: This section describes some preliminary suggestions by various people for algorithms which could be used with MASC.

17.1. Claim Size and Prefix Selection Algorithm

This section covers the algorithms used by a MASC node (on behalf of a MASC domain) to satisfy the demand for multicast addresses. The allocated addresses should be aggregatable, the address utilization should be reasonably high, and the allocation latency to the MAASs should be shorter than [WAITING_PERIOD] whenever possible.

17.1.1. Prefix Expansion

For ease of implementation and troubleshooting, MASC should use contiguous masks to specify the address ranges, i.e. prefixes. (Research indicates that sufficiently good results can be achieved using contiguous masks only.) The chosen prefixes should be as expandable as possible. The method used to choose the children sub-prefixes from the parent's prefix is the so called Reverse Bit Ordering (idea by Dave Thaler; inspired by Kampai [KAMPAI]). For example, if the parent's prefix width is four bits, the addresses of the sub-prefixes are chosen in the following order:

```
Parent:       xxxx

Child A:      0000
Child B:      1000
Child C:      0100
Child D:      1100
```

If some of the children need to expand their sub-prefix, they try to double the corresponding sub-prefix starting from the right:

```
Child A:      000x
Child A:      00xx
Child D:      110x
Child D:      11xx
```

and so on.

However, because the address ordering is very strict, to reduce the probability for collision, when a new sub-prefix has to be chosen, the choice should be random among all candidates with the same potential for expandability. For example, if the free sub-prefixes are 01xx, 10xx, 110x, then the new prefix to claim should be chosen with probability of 50% for 01xx and 50% for 10xx for example.

17.1.2. Reducing Allocation Latency

To reduce the allocation latency, a MASC node uses pre-allocation. It constantly monitors the demand for addresses from its children (or MAASs), and predicts what would be the address usage after [WAITING_PERIOD]. Only if the available addresses will be used up within [WAITING_PERIOD], a MASC node claims more addresses in advance.

17.1.3. Address Space Utilization

Because every prefix size is a power of two, if a node tries to allocate just a single prefix, the utilization at that node (i.e. at that node's domain) can be as low as 50%. To improve the utilization, a MASC node can have more than one prefix allocated at a time (typically, each of them with different size). By using a pre-allocation and allocating several prefixes of different size (see below), a MASC node should try to keep its address utilization in the range 70-90%.

17.1.4. Prefix Selection After Increase of Demand

To additionally reduce the allocation latency by reducing the probability for collision, and to improve the aggregability of the allocated addresses, a MASC node carefully chooses the prefixes to claim. The first prefix is chosen at random among all reasonably expandable candidates. If a node chooses to allocate another, smaller prefix, then, instead of doubling the size of the first one which might reduce significantly the address utilization, a second "neighbor" prefix is chosen. For example, if prefix 224.0/16 was already allocated, and the MASC domain needs 256 more addresses, the second prefix to claim will be 224.1.0/24. If the domain needs more addresses, the second prefix will eventually grow to 224.1/16, and then both prefixes can be automatically aggregated into 224.0/15. Only if 224.0.1/24 could not be allocated, a MASC node will choose another prefix (eventually random among the unused prefixes).

If the number of allocated prefixes increases above some threshold, and none of them can be extended when more addresses are needed, then, to reduce the amount of state, a MASC node should claim a new larger prefix and should stop re-claiming the older non-expandable prefixes. Research results show that up to three prefixes per MASC domain is a reasonable threshold, such that the address utilization can be in the range 70-90%, and at the same time the prefix flux will be reasonably low.

17.1.5. Prefix Selection After Decrease of Demand

If the demand for addresses decreases, such that its address space is under-utilized, a MASC node implicitly returns the unused prefixes after their lifetimes expire, or re-claims some smaller sub-prefixes. For example, if prefix 224.0/15 is 50% used by the MAASs and/or children MASC domains, and the overall utilization is such that approximately 2^16 (64K) addresses should be returned, a MASC node should stop reclaiming 224.0/15 and should start reclaiming either 224.0/16 or 224.1/16 (whichever sub-prefix utilization is higher).

17.1.6. Lifetime Extension Algorithm

If the demand for addresses did not decrease, then a MASC node re-claims the prefixes it has allocated before their lifetime expires. Each prefix (or sub-prefix if the demand has decreased) should be re-claimed every 48 hours.

18. APPENDIX B: Strawman Deployment

At the moment of writing, 225.0.0.0-225.255.255.255 is temporarily allocated to MALLOC. Presumably this block of addresses will be used for experimental deployment and testing.

If MASC were widely deployed on the Internet, we might expect numbers similar to the following:

- Initially will have approximately 128 Top-Level Domains
- Assume initially approximately 8192 level-2 MASC domains; on average, a TLD will have approximately 64 children domains.
- MASC managed global addresses:

  The following (large) ranges are not allocated yet (2^N represents the size of the contiguous mask prefixes):

  ```
  225.0.0.0 - 231.255.255.255 = 2^26 + 2^25 + 2^24
  234.0.0.0 - 238.255.255.255 = 2^25 + 2^25 + 2^24
  ---------------------------
  Total:   12*2^24 addresses
  ```

  Initially, the range 228.0.0.0 - 231.255.255.255 (4*2^24 = 2^26 = 64M) could be used by MASC as the global addresses pool. The rest (8*2^24) should be reserved. Part of it could be added later to MASC, or can be used to enlarge the pool of administratively scoped addresses (currently 239.X.X.X), or the pool for static allocation (233.X.X.X).

- If the multicast addresses are evenly distributed, each TLD would have a maximum of 2^19 (512K) addresses, while each level-2 MASC domain would have 8192 addresses.

- Initial claim size: 256 addresses/MASC domain

- Could use soft and hard thresholds to specify the maximum amount of claimed+allocated addresses per domain. For example, trigger a warning message if claimed+allocated addresses by a domain is >= 1.0*average_assumed_per_domain (a strawman default soft threshold):

  * if a TLD claim+allocation >= 512K
  * if a second level MASC domain claim+allocation >= 8K

  The hard threshold (for example, 2.0*average_assumed_per_domain) can be enforced by sending an explicit DENIED message.

  The TLDs thresholds (with regard to the claims by the second level MASC domains) is a private matter and is a part of the particular TLD policy: the thresholds could be per customer, and the warnings to the administrators could be a signal that it is time to change the policy.

- Initial claim lifetime is of the order of 30 days. Prefix lifetime is periodically (every 48 hours) reclaimed/extended, unless the prefix is under-utilized (see APPENDIX A). Because the allocation is demand-driven, the allocated prefix lifetime will be automatically extended if the MAASs need longer prefix lifetime (e.g. 3-6 months).

- A level-2 MASC domain could have children (i.e. level-3) MASC domains.

- If a level-2 or level-3 MASC domain uses less than 128 addresses, a Layer 2 protocol/mechanism (e.g. AAP) should be run among that domain and its parent MASC domain.

19. Authors' Addresses

Pavlin Radoslavov
Computer Science Department
University of Southern California/ISI
Los Angeles, CA 90089
USA

EMail: pavlin@catarina.usc.edu

Deborah Estrin
Computer Science Department
University of Southern California/ISI
Los Angeles, CA 90089
USA

EMail: estrin@isi.edu

Ramesh Govindan
University of Southern California/ISI
4676 Admiralty Way
Marina Del Rey, CA 90292
USA

EMail: govindan@isi.edu

Mark Handley
AT&T Center for Internet Research at ISCI (ACIRI)
1947 Center St., Suite 600
Berkeley, CA 94704
USA

EMail: mjh@aciri.org

Satish Kumar
Computer Science Department
University of Southern California/ISI
Los Angeles, CA 90089
USA

EMail: kkumar@usc.edu

David Thaler
Microsoft
One Microsoft Way
Redmond, WA 98052
USA

EMail: dthaler@microsoft.com

20. References

[AAP] Handley, M. and S. Hanna, "Multicast Address Allocation Protocol (AAP)", Work in Progress.

[API] Finlayson, R., "An Abstract API for Multicast Address Allocation", RFC 2771, February 2000.

[BGMP] Thaler, D., Estrin, D. and D. Meyer, "Border Gateway Multicast Protocol (BGMP): Protocol Specification", Work in Progress.

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

[CIDR] Rekhter, Y. and C. Topolcic, "Exchanging Routing Information Across Provider Boundaries in the CIDR Environment", RFC 1520, September 1993.

[IANA] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994.

[IANA-CONSIDERATIONS] Alvestrand, H. and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[IPSEC] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[KAMPAI] Tsuchiya, P., "Efficient and Flexible Hierarchical Address Assignment", INET92, June 1992, pp. 441--450.

[MADCAP] Hanna, S., Patel, B. and M. Shah, "Multicast Address Dynamic Client Allocation Protocol (MADCAP)", RFC 2730, December 1999.

[MALLOC] Thaler, D., Handley, M. and D. Estrin, "The Internet Multicast Address Allocation Architecture", RFC 2908, September 2000.

[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", Work in Progress.

[MZAP] Handley, M, Thaler, D. and R. Kermode "Multicast-Scope Zone Announcement Protocol (MZAP)", RFC 2776, February 2000.

[RFC1112] Deering, S., "Host Extensions for IP Multicasting", STD 5, RFC 1112, August 1989.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2373] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

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

21. Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.