Mobile Ad hoc Networks Working
Group
Internet-Draft
Intended status: Standards Track
Expires: November 2, 2011

S. Ratliff
B. Berry
G. Harrison
S. Jury
D. Satterwhite
Cisco Systems
May 2, 2011

Dynamic Link Exchange Protocol (DLEP)
draft-ietf-manet-dlep-01

Abstract

When routing devices rely on modems to effect communications over wireless links, they need timely and accurate knowledge of the characteristics of the link (speed, state, etc.) in order to make forwarding decisions. In mobile or other environments where these characteristics change frequently, manual configurations or the inference of state through routing or transport protocols does not allow the router to make the best decisions. A bidirectional, event-driven communication channel between the router and the modem is necessary.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

There exist today a collection of modem devices that control links of variable bandwidth and quality. Examples of these types of links include line-of-sight (LOS) radios, satellite terminals, and cable/ DSL modems. Fluctuations in speed and quality of these links can occur due to configuration (in the case of cable/DSL modems), or on a moment-to-moment basis, due to physical phenomena like multipath interference, obstructions, rain fade, etc. It is also quite possible that link quality and bandwidth varies with respect to individual neighbors on a link, and with the type of traffic being sent. As an example, consider the case of an 802.11g access point, serving 2 associated laptop computers. In this environment, the answer to the question "What is the bandwidth on the 802.11g link?" is "It depends on which associated laptop we're talking about, and on what kind of traffic is being sent." While the first laptop, being physically close to the access point, may have a bandwidth of 54Mbps for unicast traffic, the other laptop, being relatively far away, or obstructed by some object, can simultaneously have a bandwidth of only 32Mbps for unicast. However, for multicast traffic sent from the access point, all traffic is sent at the base transmission rate (which is configurable, but depending on the model of the access point, is usually 24Mbps or less).

In addition to utilizing variable bandwidth links, mobile networks are challenged by the notion that link connectivity will come and go over time. Effectively utilizing a relatively short-lived connection is problematic in IP routed networks, as routing protocols tend to rely on independent timers at OSI Layer 3 to maintain network convergence (e.g. HELLO messages and/or recognition of DEAD routing adjacencies). These short-lived connections can be better utilized with an event-driven paradigm, where acquisition of a new neighbor (or loss of an existing one) is somehow signaled, as opposed to a timer-driven paradigm.

Another complicating factor for mobile networks are the different methods of physically connecting the modem devices to the router. Modems can be deployed as an interface card in a router's chassis, or as a standalone device connected to the router via Ethernet, USB, or even a serial link. In the case of Ethernet or serial attachment, with existing protocols and techniques, routing software cannot be aware of convergence events occurring on the radio link (e.g. acquisition or loss of a potential routing neighbor), nor can the router be aware of the actual capacity of the link. This lack of awareness, along with the variability in bandwidth, leads to a situation where quality of service (QoS) profiles are extremely difficult to establish and properly maintain. This is especially true of demand-based access schemes such as Demand Assigned Multiple Access (DAMA) implementations used on some satellite systems. With a DAMA-based system, additional bandwidth may be available, but will not be used unless the network devices emit traffic at rate higher than the currently established rate. Increasing the traffic rate does not guarantee additional bandwidth will be allocated; rather, it may result in data loss and additional retransmissions on the link.

In attempting to address the challenges listed above, the authors have developed the Data Link Exchange Protocol, or DLEP. The DLEP protocol runs between a router and its attached modem devices, allowing the modem to communicate link characteristics as they change, and convergence events (acquisition and loss of potential routing neighbors). The diagram below is used to illustrate the scope of DLEP sessions. When a local client (Modem device) detects the presence of a remote neighbor, it sends an indication to its local router via the DLEP session. Upon receipt of the indication, the local router would take appropriate action (e.g. initiation of discovery or HELLO protocols) to converge the network. After notification of the new neighbor, the modem device utilizes the DLEP session to report the characteristics of the link (bandwidth, latency, etc) to the router on an as-needed basis. Finally, the Modem is able to use the DLEP session to notify the router when the remote neighbor is lost, shortening the time required to re-converge the network.

```
   |-----Local Neighbor-----|          |-----Remote Neighbor----|
   |                        |          |     (far-end device)   |

   +--------+       +-------+          +-------+       +--------+
   | Router |=======| Modem |{~~~~~~~~}| Modem |=======| Router |
   |        |       | Device|          | Device|       |        |
   +--------+       +-------+          +-------+       +--------+

            |       |       | Link     |       |       |
            |-DLEP--|       | Protocol |       |-DLEP--|
            |       |       | (e.g.    |       |       |
            |       |       | 802.11)  |       |       |
```

Figure 1: DLEP Network

DLEP exists as a collection of type-length-value (TLV) based messages using [RFC5444] formatting. The protocol can be used for both Ethernet attached modems (utilizing, for example, a UDP socket for transport of the RFC 5444 packets), or in environments where the modem is an interface card in a chassis (via a message passing scheme). DLEP utilizes a session paradigm between the modem device and its associated router. If multiple modem devices are attached to a router, a separate DLEP session MUST exist for each modem. If a modem device supports multiple connections to a router (via multiple interfaces), or supports connections to multiple routers, a separate DLEP session MUST exist for each connection.

1.1 Requirements

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 BCP 14, RFC 2119 [RFC2119].

2. Assumptions

In order to implement discovery in the DLEP protocol (thereby avoiding some configuration), we have defined a first-speaker and a passive-listener scheme. Specifically, the router is defined as the passive-listener, and the modem device defined as the first-speaker (e.g. the initiator for discovery). Borrowing from existing terminology, this document refers to the first-speaker as the 'client', even though there is no client/server relationship in the classic sense.

DLEP assumes that participating modem devices appear to the router as a transparent bridge - specifically, the assumption is that the destination MAC address for data traffic in any frame emitted by the router should be the MAC address of the next-hop router or end-device, and not the MAC address of any of the intervening modem devices.

DLEP assumes that security on the session (e.g. authentication of session partners, encryption of traffic, or both) is dealt with by the underlying transport mechanism for the RFC 5444 packets (e.g. by using a transport such as DTLS [DTLS]).

The optional [RFC5444] message header Sequence Number MUST be included in all DLEP packets. Sequence Numbers start at 1 and are incremented by one for each original and retransmitted message. The unsigned 16-bit Sequence Number rolls over at 65535 to 1. A Sequence Number of 0 is not valid. Peer level Sequence Numbers are unique within the context of a DLEP session. Sequence numbers are used in DLEP to correlate a response to a request.

3. Normal Session Flow

A session between a router and a client is established by exchanging the "Peer Discovery" and "Peer Offer" messages described below.

Once that exchange has successfully occurred, the client informs the router of the presence of a new potential routing partner via the "Neighbor Up" message. The loss of a neighbor is communicated via the "Neighbor Down" message, and link quality is communicated via the "Neighbor Update" message. Note that, due to the issue of metrics varying depending on neighbor (discussed above), DLEP link metrics are expressed within the context of a neighbor relationship, instead of on the link as a whole.

Once the DLEP session has started, the session partners exchange heartbeat messages based on a negotiated time interval. The heartbeat messages are used to assure the session partners are in an appropriate state, and that bidirectional connectivity still exists.

In addition to receiving metrics about the link, DLEP provides for the ability for the router to request a different amount of bandwidth, or latency, for its client via the Link Characteristics Message. This allows the router to deal with requisite increases (or decreases) of allocated bandwidth/latency in demand-based schemes in a more deterministic manner.

4. Generic DLEP Packet Definition

The Generic DLEP Packet Definition follows the format for packets defined in [RFC5444].

The Generic DLEP Packet Definition contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Flags | Packet Sequence Number          | Packet TLV   |
|       |       |                                 | Block...     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Message (Contains DLEP message)...                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Version - Version of RFC 5444 specification on which the packet/messages/TLVs are constructed.

Flags - 4 bit field. All bits MUST be ignored by DLEP implementations.

Packet Sequence Number - If present, the packet sequence number is parsed and ignored. DLEP does NOT use or generate packet sequence numbers.

Packet TLV block - a TLV block which contains packet level TLV information. DLEP implementations MUST NOT use this TLV block.

Message - the packet MAY contain zero or more messages, however, DLEP messages are encoded within an RFC 5444 Message TLV Block.

5. Message Header Format

DLEP utilizes the following format for the RFC 5444 message header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Msg Type   |Msg Flg|AddrLen|          Message Size         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Message Seq Num      |           TLV Block...        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - an 8-bit field which specifies the type of the message. For DLEP, this field contains DLEP_MESSAGE (value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - a 4-bit unsigned integer field encoding the length of all addresses included in this message. DLEP implementations do not use this field; contents SHOULD be ignored.

Message Size - a 16-bit unsigned integer field which specifies the number of octets that make up the message including the message header.

Message Sequence Number - a 16-bit unsigned integer field that contains a sequence number, generated by the originator of the message. Sequence numbers range from 1 to 65535. Sequence numbers roll over at 65535 to 1; 0 is invalid.

TLV Block - TLV Block included in the message.

6. Message TLV Block Format

The DLEP protocol is organized as a set of orders, each with a collection of Sub-TLVs. The Sub-TLVs carry information needed to process and/or establish context (e.g. the MAC address of a far-end router), and the 'tlv-type' field in the message TLV block carries the DLEP order itself. The DLEP orders are enumerated in section 8.1 of this document, and the messages created using these orders are documented in sections 9 through 24.

DLEP uses the following settings for an RFC 5444 Message TLV block:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       TLVs Length             |  TLV Type     | TLV Flags     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Length     |       Value...                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLVs Length - a 16-bit unsigned integer field that contains the total number of octets in all of the immediately following TLV elements (tlvs-length not included).

TLV Type - an 8-bit unsigned integer field specifying the type of the TLV. DLEP uses this field to specify the DLEP order. Valid DLEP orders are defined in section 8.1 of this document.

TLV Flags - an 8-bit flags bit field. Bit 3 (thasvalue) MUST be set; all other bits are not used and MUST be set to '0'.

Length - Length of the 'Value' field of the TLV

Value - A field of length <Length> which contains data specific to a particular TLV type. In the DLEP case, this field will consist of a collection of DLEP sub-TLVs appropriate for the DLEP action specified in the TLV type field.

7. DLEP sub-TLVs

DLEP protocol messages are transported in an RFC 5444 message TLV. All DLEP messages use the RFC 5444 DLEP_MESSAGE value (TBD). The protocol messages consist of a DLEP order, encoded in the 'tlv-type' field in the message TLV block, with the 'value' field of the TLV block containing a collection (1 or more) DLEP sub-TLVs.

The format of DLEP Sub-TLVs is consistent with RFC 5444 in that the Sub-TLVs contain a flag field in addition to the type, length, and value fields. Valid DLEP Sub-TLVs are:

TLV	TLV
Value	Description
TBD	Identification sub-TLV
TBD	DLEP Version sub-TLV
TBD	Peer Type sub-TLV
TBD	MAC Address sub-TLV
TBD	IPv4 Address sub-TLV
TBD	IPv6 Address sub-TLV

TBD	Maximum Data Rate (MDR) sub-TLV
TBD	Current Data Rate (CDR) sub-TLV
TBD	Latency sub-TLV
TBD	Resources sub-TLV
TBD	Relative Link Quality (RLQ) sub-TLV
TBD	Status sub-TLV
TBD	Heartbeat Interval sub-TLV
TBD	Heartbeat Threshold sub-TLV
TBD	Neighbor down ACK timer sub-TLV
TBD	Link Characteristics ACK timer sub-TLV

DLEP sub-TLVs contain the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  TLV Type     |TLV Flags=0x10 | Length        | Value...      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
TLV Type    - an 8-bit unsigned integer field specifying the type
              of the sub-TLV.

TLV Flags   - an 8-bit flags bit field. Bit 3 (thasvalue) MUST be
              set, all other bits are not used and MUST be set to
              '0'.

Length      - an 8-bit length of the value field of the sub-TLV

Value       - A field of length <Length> which contains data
              specific to a particular sub-TLV.
```

7.1 Identification Sub-TLV

This Sub-TLV MUST exist in the TLV Block for all DLEP messages, and MUST be the first Sub-TLV of the message. Further, there MUST be ONLY one Identification Sub-TLV in an RFC 5444 message TLV block. The Identification sub-TLV contains client and router identification information used to establish the proper context for processing DLEP protocol messages.

The Identification sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type = TBD |TLV Flags=0x10 |Length = 8     | Router ID     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Router ID                  | Client ID     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Client ID                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type   - Value TBD

TLV Flags   - 0x10, Bit 3 (thasvalue) is set, all other bits are unused and MUST be set to '0'.

Length   - 8

Router ID   - indicates the router ID of the DLEP session.

Client ID   - indicates the client ID of the DLEP session.

When the client initiates discovery (via the Peer Discovery message), it MUST set the Client ID to a 32-bit quantity that will be used to uniquely identify this session from the client-side. The client MUST set the Router ID to '0'. When responding to the Peer Discovery message, the router MUST echo the Client ID, and MUST supply its own unique 32-bit quantity to identify the session from the router's perspective. After the Peer Discovery/Peer Offer exchange, both the Client ID and the Router ID MUST be set to the values obtained from the Peer DIscovery/Peer Offer sequence.

7.2 DLEP Version Sub-TLV

The DLEP Version Sub-TLV is an OPTIONAL TLV in both the Peer Discovery and Peer Offer messages. The Version Sub-TLV is used to indicate the client or router version of the protocol. The client and router MAY use this information to decide if the peer is running at a supported level.

The DLEP Version Sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD  |TLV Flags=0x10 |Length=4       | Major Version |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Major Version |       Minor Version           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type   - TBD

TLV Flags   - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length   - Length is 4

Major Version - Major version of the client or router protocol.

Minor Version - Minor version of the client or router protocol.

Support of this draft is indicated by setting the Major Version to '1', and the Minor Version to '1' (e.g. Version 1.1).

7.3 Peer Type Sub-TLV

The Peer Type Sub-TLV is used by the router and client to give additional information as to its type. It is an OPTIONAL sub-TLV in both the Peer Discovery Message and the Peer Offer message. The peer type is a string and is envisioned to be used for informational purposes (e.g. display command).

The Peer Type sub-TLV contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =TBD  |TLV Flags=0x10 |Length= peer   |Peer Type Str  |
  |               |               |type string len|Max Len = 80   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   TLV Type         - TBD

   TLV Flags        - 0x10, Bit 3 (thasvalue) is set, all other bits
                      are not used and MUST be set to '0'.

   Length           - length of peer type string (80 bytes maximum)
   Peer Type String - Non-Null terminated peer type string, maximum
                      length of 80 bytes. For example, a satellite
                      modem might set this variable to 'Satellite
                      terminal'.
```

7.4 MAC Address Sub-TLV

The MAC address Sub-TLV MUST appear in all neighbor-oriented messages (e.g. Neighbor Up, Neighbor Up ACK, Neighbor Down, Neighbor Down ACK, Neighbor Update, Link Characteristics Request, and Link Characteristics ACK). The MAC Address sub-TLV contains the address of the far-end (neighbor) router.

The MAC Address sub-TLV contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =TBD  |TLV Flags=0x10 |Length = 6     |MAC Address    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                      MAC Address                              |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | MAC Address   |
  +-+-+-+-+-+-+-+-+
```

```
   TLV Type    - TBD

   TLV Flags   - 0x10, Bit 3 (thasvalue) is set, all other bits are not
                 used and MUST be set to '0'.
```

Length - 6

MAC Address - MAC Address of the far-end router.

7.5 IPv4 Address Sub-TLV

The IPv4 Address Sub-TLV MAY be used in Neighbor Up, Neighbor Update, and Peer Update Messages, if the client is aware of the Layer 3 address. When included in Neighbor messages, the IPv4 Address sub-TLV contains the IPv4 address of the far-end router (neighbor). In the Peer Update message, it contains the IPv4 address of the local router. In either case, the sub-TLV also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv4 Address Sub-TLV contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Type =TBD  |TLV Flags=0x10 |Length = 5     |   Add/Drop    |
   |               |               |               |   Indicator   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     IPv4 Address                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 5

Add/Drop Indicator - Value indicating whether this is a new or existing address (0x01), or a withdrawal of an address (0x02).

IPv4 Address - IPv4 Address of the far-end router.

7.6 IPv6 Address Sub-TLV

The IPv6 Address Sub-TLV MAY be used in Neighbor Up, Neighbor Update, and Peer Update Messages, if the client is aware of the Layer 3 address. When included in Neighbor messages, the IPv6 Address sub-TLV contains the IPv6 address of the far-end router (neighbor). In the Peer Update, it contains the IPv6 address of the local router. In either case, the sub-TLV also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv6 Address sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD   |TLV Flags=0x10 |Length = 17    |   Add/Drop    |
|                |               |               |   Indicator   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         IPv6 Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         IPv6 Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         IPv6 Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         IPv6 Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
TLV Type     - TBD

TLV Flags    - 0x10, Bit 3 (thasvalue) is set, all other bits are not
               used and MUST be set to '0'.

Length       - 17

Add/Drop     - Value indicating whether this is a new or
Indicator      existing address (0x01), or a withdrawal of
               an address (0x02).

IPv6 Address - IPv6 Address of the far-end router.
```

7.7 Maximum Data Rate Sub-TLV

The Maximum Data Rate (MDR) Sub-TLV is used in Neighbor Up, Neighbor Update, and Link Characteristics ACK Messages to indicate the maximum theoretical data rate, in bits per second, that can be achieved on the link. When metrics are reported via the messages listed above, the maximum data rate MUST be reported.

The Maximum Data Rate sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD   |TLV Flags=0x10 |Length = 8     |  MDR (bps)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         MDR (bps)                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         MDR (bps)             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
TLV Type              -  TBD
```

```
   TLV Flags             -  0x10, Bit 3 (thasvalue) is set, all other
                            bits are not used and MUST be set to '0'.

   Length                -  8

   Maximum Data Rate     -  A 64-bit unsigned number, representing the
                            maximum theoretical data rate, in bits per
                            second (bps), that can be achieved on the
                            link.
```

7.8 Current Data Rate Sub-TLV

The Current Data Rate (CDR) Sub-TLV is used in Neighbor Up, Neighbor Update, Link Characteristics Request, and Link Characteristics ACK messages to indicate the rate at which the link is currently operating, or in the case of the Link Characteristics Request, the desired data rate for the link.

The Current Data Rate sub-TLV contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Type =TBD  |TLV Flags=0x10 |Length = 8     |CDR (bps)      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        CDR (bps)                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        CDR (bps)              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   TLV Type              -  TBD

   TLV Flags             -  0x10, Bit 3 (thasvalue) is set, all other
                            bits are not used and MUST be set to '0'.

   Length                -  8

   Current Data Rate     -  A 64-bit unsigned number, representing the
                            current data rate, in bits per second (bps),
                            on the link. When reporting metrics (e.g,
                            in Neighbor Up, Neighbor Down, or Link
                            Characteristics ACK), if there is no
                            distinction between current and maximum
                            data rates, current data rate SHOULD be
                            set equal to the maximum data rate.
```

7.9 Latency Sub-TLV

The Latency Sub-TLV is used in Neighbor Up, Neighbor Update, Link Characteristics Request, and Link Characteristics ACK messages to indicate the amount of latency on the link, or in the case of the Link Characteristics Request, to indicate the maximum latency required (e.g. a should-not-exeed value) on the link.

The Latency Sub-TLV contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =TBD  |TLV Flags=0x10 |Length = 2     |Latency (ms)   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Latency (ms)   |
  +-+-+-+-+-+-+-+-+
```

```
   TLV Type             -  TBD

   TLV Flags            -  0x10, Bit 3 (thasvalue) is set, all other
                           bits are not used and MUST be set to '0'.

   Length               -  2

   Latency              -  the transmission delay that a packet
                           encounters as it is transmitted over the
                           link. In Neighbor Up, Neighbor Update,
                           and Link Characteristics ACK, this value
                           is reported in absolute delay, in
                           milliseconds. The calculation of latency
                           is modem-device dependent. For example,
                           the latency may be a running average
                           calculated from the internal queuing. If
                           the modem device cannot calculate latency,
                           it SHOULD be reported as 0. In the Link
                           Characteristics Request Message, this value
                           represents the maximum delay, in milliseconds,
                           expected on the link.
```

7.10 Resources Sub-TLV

The Resources Sub-TLV is used in Neighbor Up, Neighbor Update, and Link Characteristics ACK messages to indicate a percentage (0-100) amount of resources (e.g. battery power) remaining on the modem device.

The Resources TLV contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =TBD  |TLV Flags=0x10 |Length = 1     |   Resources   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   TLV Type             -  TBD

   TLV Flags            -  0x10, Bit 3 (thasvalue) is set, all other
                           bits are not used and MUST be set to '0'.

   Length               -  1
```

Resources - a percentage, 0-100, representing the amount of remaining resources, such as battery power. If resources cannot be calculated, a value of 100 SHOULD be reported.

7.11 Relative Link Quality Sub-TLV

The Relative Link Quality (RLQ) Sub-TLV is used in Neighbor Up, Neighbor Update, and Link Characteristics ACK messages to indicate the quality of the link as calculated by the modem device.

The Relative Link Quality sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD  |TLV Flags=0x10 |Length = 1     |Relative Link  |
|               |               |               |Quality (RLQ)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

Length - 1

Relative Link Quality - a non-dimensional number, 0-100, representing the relative link quality. A value of 100 represents a link of the highest quality. If the RLQ cannot be calculated, a value of 100 SHOULD be reported.

7.12 Status Sub-TLV

The Status Sub-TLV is sent from either the client or router to indicate the success or failure of a given request

The Status Sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD  |TLV Flags=0x10 |Length = 1     |     Code      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type - TBD

TLV Flags - 0x10, Bit 3 (thasvalue) is set, all other bits are not used and MUST be set to '0'.

```
   Length           - 1

   Termination Code - 0 = Success
                      Non-zero = Failure. Specific values of a non-
                      zero termination code depend on the operation
                      requested (e.g. Neighbor Up, Neighbor Down, etc).
```

7.13 Heartbeat Interval Sub-TLV

The Heartbeat Interval Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired Heartbeat timeout window. If included in the Peer Discovery, the router MUST either accept the timeout interval, or reject the Peer Discovery.

The Heartbeat Interval Sub-TLV contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =TBD  |TLV Flags=0x10 |Length = 1     | Interval      |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   TLV Type         - TBD

   TLV Flags        - 0x10, Bit 3 (thasvalue) is set, all other bits are
                      not used and MUST be set to ’0’.

   Length           - 1

   Interval         - 0 = Do NOT use heartbeats on this peer-to-peer
                      session. Non-zero = Interval, in seconds, for
                      heartbeat messages.
```

7.14 Heartbeat Threshold Sub-TLV

The Heartbeat Threshold Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired number of windows, of time (Heartbeat Interval) seconds, to wait before either peer declares the peer-to-peer session lost. In this case, the overall amount of time before a peer-to-peer session is declared lost is expressed as (Interval * Threshold), where ’Interval’ is the value in the Heartbeat Interval sub-TLV, documented above. If this sub-TLV is included by the client in the Peer Discovery, the client MUST also specify the Heartbeat Interval sub-TLV with a non-zero interval. If this sub-TLV is received during Peer Discovery, the router MUST either accept the threshold, or reject the Peer Discovery.

The Heartbeat Threshold Sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD  |TLV Flags=0x10 |Length = 1     | Threshold     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
TLV Type          - TBD

TLV Flags         - 0x10, Bit 3 (thasvalue) is set, all other bits are
                    not used and MUST be set to '0'.

Length            - 1

Threshold         - 0 = Do NOT use heartbeats on this peer-to-peer
                    session. Non-zero = Number of windows, of
                    Heartbeat Interval seconds, to wait before
                    declaring a peer-to-peer session to be lost.
```

7.15 Link Characteristics ACK Timer Sub-TLV

The Link Characteristic ACK Timer Sub-TLV MAY be sent from the client during Peer Discovery to indicate the desired number of seconds the router should wait for a response to a Link Characteristics Request. If this sub-TLV is received during Peer Discovery, the router MUST either accept the timeout value, or reject the Peer Discovery.

The Link Characteristics ACK Timer Sub-TLV contains the following fields:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TLV Type =TBD  |TLV Flags=0x10 |Length = 1     | Interval      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
TLV Type          - TBD

TLV Flags         - 0x10, Bit 3 (thasvalue) is set, all other bits are
                    not used and MUST be set to '0'.

Length            - 1

Interval          - 0 = Do NOT use timeouts for Link Characteristics
                    requests on this peer-to-peer session.
                    Non-zero = Interval, in seconds, to wait before
                    considering a Link Characteristics Request has
                    been lost.
```

8. DLEP Protocol Messages

DLEP places no additional requirements on the RFC 5444 Packet formats, or the packet header. DLEP does require that the optional 'msg-seq-num' in the message header exist, and defines a set of values for the 'tlv-type' field in the RFC 5444 TLV block. Therefore, a DLEP message, starting from the RFC 5444 Message header, would appear as follows:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |                               |
  | (value TBD)   |       |       |                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      | TLV block length (length of   |
  |                               | DLEP order + Sub-TLVs)        |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Message  |TLV Flags=0x10 | Length        | Start of DLEP |
  | Block value   |               |               | Sub-TLVs...   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

8.1 Message Block TLV Values

As mentioned above, all DLEP messages utilize a single RFC 5444 message type, the DLEP_MESSAGE (TBD). DLEP further identifies protocol messages by using the 'tlv-type' field in the RFC 5444 message TLV block. DLEP defines the following Message-Type-specific values for the tlv-type field:

TLV	TLV
Value	Description
TBD	Attached Peer Discovery
TBD	Detached Peer Discovery
TBD	Peer Offer
TBD	Peer Update
TBD	Peer Update ACK
TBD	Peer Termination
TBD	Peer Termination ACK
TBD	Neighbor Up
TBD	Neighbor Up ACK
TBD	Neighbor Down
TBD	Neighbor Down ACK
TBD	Neighbor Update
TBD	Neighbor Address Update
TBD	Neighbor Address Update ACK
TBD	Heartbeat
TBD	Link Characteristics Request
TBD	Link Characteristics ACK

In all of the diagrams following, the message layouts begin with the RFC 5444 message header.

9. Peer Discovery Messages

There are two different types of Peer Discovery Messages, Attached and Detached. Attached Peer Discovery Messages are sent by the client when it is directly attached to the router (e.g. the client exists as a card in the chassis, or it is connected via Ethernet with no intervening devices). The Detached Peer Discovery message, on the other hand, is sent by a "remote" client -- for example, a client at a satellite hub system might use a Detached Discovery Message in order to act as a proxy for remote ground terminals. To explain in another way, a detached client uses the variable link itself (the radio or satellite link) to establish a DLEP session with a remote router.

9.1 Attached Peer Discovery Message

The Attached Peer Discovery Message is sent by an attached client to a router to begin a new DLEP association. The Peer Offer message is required to complete the discovery process. The client MAY implement its own retry heuristics in the event it (the client) determines the Attached Peer Discovery Message has timed out.

The Attached Peer Discovery Message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   | DLEP_MESSAGE  |  0x1  |  0x0  |       22 + size of opt        |
   | (value TBD)   |       |       |            sub-TLV            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Message Seq Num        |TLVs Length =14 + opt sub-TLVs |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Attached |TLV Flags=0x10 | Length =11 +  | Sub-TLV type= |
   | Peer Discovery|               | opt sub-TLVs  | Identification|
   | (Value TDB)   |               |               | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |           Router ID           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Router ID           |           Client ID           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Client ID           |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP Version   |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 4     |         Major Version         | Minor Version |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Minor Version |Sub-TLV type=  |TLV Flags=0x10 | Length = Len  |
   |               |Peer Type (TBD)|               |of peer string |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               (Continued on next page)                        |
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                (Continued from previous page)                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Peer Type Str |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
   |MaxLen=80 bytes|Heartbeat Int. |               |               |
   |               |(TBD)          |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Heartbeat     |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
   | Interval      |HB Thresh.     |               |               |
   | (seconds)     |(TBD)          |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Heartbeat     |Sub-TLV Type=  |TLV FLags=0x10 | Length = 1    |
   | Threshold     |Link Char. ACK |               |               |
   |(# of windows) |Timer (TBD)    |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Link Char ACK  |
   |Timer (sec)    |
   +-+-+-+-+-+-+-+-+
```

```
    Message Type                    - DLEP_MESSAGE (value TBD)

    Message Flags                   - Set to 0x1 (bit 3, mhasseqnum
                                      bit is set).  No other bits are
                                      used and MUST be set to '0'.

    Message Address Length          - 0x0

    Message Size                    - 22 + size of optional sub-TLVs

    Message Sequence Number         - a 16-bit unsigned integer field
                                      containing a sequence number
                                      generated by the message
                                      originator.

    TLV Block                       - TLVs Length: 14 + size of optional
                                                   sub-TLVs.

                                      DLEP Attached Peer Disc. order
                                      Identification TLV (MANDATORY)
                                      Version Sub-TLV (OPTIONAL)
                                      Peer Type Sub-TLV (OPTIONAL)
                                      Heartbeat Int. Sub-TLV (OPTIONAL)
                                      Heartbeat Threshold Sub-TLV (OPT.)
                                      Link Characteristics ACK Timer
                                      Sub-TLV (OPTIONAL)
```

9.2 Detached Peer Discovery Message

The Detached Peer Discovery Message is sent by a detached client proxy to a router to begin a new DLEP session. The Peer Offer message is required to complete the discovery process. The client MAY implement its own retry heuristics in the event it (the client) determines the Detached Peer Discovery Message has timed out.

The Detached Peer Discovery Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |        22 + size of opt       |
  | (value TBD)   |       |       |            sub-TLV            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length =14 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Detached |TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
  | Peer Discovery|               | opt sub-TLVs  | Identification|
  | (Value TDB)   |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP Version   |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 4     |         Major Version         | Minor Version |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Minor Version |Sub-TLV type=  |TLV Flags=0x10 | Length = Len  |
  |               |Peer Type (TBD)|               |of peer string |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Peer Type Str |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
  |MaxLen=80 bytes|Heartbeat Int. |               |               |
  |               |(TBD)          |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Heartbeat     |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
  | Interval      |HB Thresh.     |               |               |
  | (seconds)     |(TBD)          |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Heartbeat     |Sub-TLV Type=  |TLV FLags=0x10 | Length = 1    |
  | Threshold     |Link Char. ACK |               |               |
  |(# of windows) |Timer (TBD)    |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Link Char ACK  |
  |Timer (sec)    |
  +-+-+-+-+-+-+-+-+
```

Message Type   - DLEP_MESSAGE (value TBD)

```
   Message Flags                  - Set to 0x1 (bit 3,
                                    mhasseqnum bit is set).
                                    All other bits are not used
                                    and MUST be set to '0'.

   Message Address Length         - 0x0

   Message Size                   - 22 + size of optional
                                    sub-TLVs

   Message Sequence Number        - A 16-bit unsigned integer
                                    field containing a sequence
                                    number, generated by the
                                    message originator.

   TLV Block                      - TLVs Length: 14 + size of
                                     optional sub-TLVs.

                                    DLEP Detached Peer Discovery order
                                    Identification sub-TLV (MANDATORY)
                                    Version sub-TLV (OPTIONAL)
                                    Peer Type Sub-TLV (OPTIONAL)
                                    Heartbeat Interval Sub-TLV (OPTIONAL)
                                    Heartbeat Threshold Sub-TLV (OPTIONAL)
                                    Link Char. ACK Timer Sub-TLV(OPTIONAL)
```

10. Peer Offer Message

The Peer Offer Message is sent by a router to a client or client proxy in response to a Peer Discovery Message. The Peer Offer Message is the response to either of the Peer Discovery messages (either Attached or Detached), and completes the DLEP session establishment.

The Peer Offer Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |        22 + size of opt       |
  | (value TBD)   |       |       |            sub-TLV            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length =14 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |DLEP Peer Offer|TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
  | (Value TBD)   |               | opt sub-TLVs  |Identification |
  |               |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                  (Continued on next page)                     |
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                  (Continued from above)                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP Version   |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Length = 4    |         Major Version         | Minor Version |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Minor Version |Peer Type sub- |TLV Flags=0x10 | Length = Len  |
  |               |TLV = TDB      |               |of peer string |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Peer Type Str |DLEP IPv4 sub- |TLV Flags=0x10 | Length = 5    |
  |MaxLen=80 bytes|TLV (TBD)      |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Add/Drop Ind. |                 IPv4 Address                  |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | IPv4 Address  |DLEP IPv6 sub- |TLV Flags=0x10 | Length = 17   |
  |               |TLV Type = TBD |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Add/Drop Ind. |                 IPv6 Address                  |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |IPv6 Address   |Sub-TLV type=  |TLV Flags=0x10 | Length = 1    |
  |               |Heartbeat Int. |               |               |
  |               |(TBD)          |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Heartbeat    |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
  |  Interval     |Heartbeat      |               |               |
  |  (seconds)    |Threshold (TBD)|               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Heartbeat    |Sub-TLV Type=  |TLV FLags=0x10 | Length = 1    |
  |  Threshold    |Link Char. ACK |               |               |
  | (# of windows)|Timer (TBD)    |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Link Char ACK  |Sub-TLV Type=  |TLV Flags=0x10 | Length = 1    |
  |Timer (sec)    |DLEP Status    |               |               |
  |               |(TBD)          |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Code          |
  +-+-+-+-+-+-+-+-+
```

Message Type   - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to ’0’.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

TLV Block - TLV Length: 14 + size of optional sub-TLVs
DLEP Peer Offer order
Identification sub-TLV (MANDATORY)
DLEP Version sub-TLV (OPTIONAL)
Peer Type sub-TLV (OPTIONAL)
IPv4 Address sub-TLV (OPTIONAL)
IPv6 Address sub-TLV (OPTIONAL)
Status sub-TLV (OPTIONAL)
Heartbeat Interval Sub-TLV (OPTIONAL)
Heartbeat Threshold Sub-TLV (OPTIONAL)
Link Char. ACK Timer Sub-TLV (OPTIONAL)

11. Peer Update Message

The Peer Update message is sent by the router to indicate local Layer 3 address changes. For example, addition of an IPv4 address to the router would prompt a Peer Update message to its attached DLEP clients. If the modem device is capable of understanding and forwarding this information, the address update would prompt any remote DLEP clients (DLEP clients that are on the far-end of the variable link) to issue a "Neighbor Update" message to their local routers, with the address change information. Clients that do not track Layer 3 addresses MUST silently ignore the Peer Update Message. Clients that track Layer 3 addresses MUST acknowledge the Peer Update with a Peer Update ACK message. Routers MAY employ heuristics to retransmit Peer Update messages. Sending of Peer Update Messages SHOULD cease when a router implementation determines that a partner modem device does NOT support Layer 3 address tracking.

The Peer Update Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |        22 + size of opt       |
  | (value TBD)   |       |       |            sub-TLVs           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length =14 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Peer     |TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
  | Update        |               | opt sub-TLVs  | Identification|
  | (Value TDB)   |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP IPv4      |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 5     | Add/Drop Ind. |         IPv4 Address          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |         IPv4 Address          | Sub-TLV type= |TLV Flags=0x10 |
  |                               | DLEP IPv6     |               |
  |                               | sub-TLV (TBD) |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 17    | Add/Drop Ind. |         IPv6 Address          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |        IPv6 Address           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + optional Sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

```
   TLV Block                - TLV Length:  14 + length of optional
                                           sub-TLVs.
                              DLEP Peer Update order
                              Identification sub-TLV (MANDATORY)
                              IPv4 Address Sub-TLV (OPTIONAL)
                              IPv6 Address Sub-TLV (OPTIONAL)
```

12. Peer Update ACK Message

The client sends the Peer Update ACK Message to indicate whether a Peer Update Message was successfully processed.

The Peer Update ACK message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   | DLEP_MESSAGE  |  0x1  |  0x0  |       22 + size of opt        |
   | (value TBD)   |       |       |            sub-TLVs           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |TLVs Length =14 + opt sub-TLVs |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Peer     |TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
   | Update ACK    |               | opt sub-TLVs  | Identification|
   | (Value TDB)   |               |               | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |           Router ID           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Router ID         |           Client ID           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Client ID         |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP Status    |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 1     |      Code     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Message Type             - DLEP_MESSAGE (Value TBD)

   Message Flags            - Set to 0x1 (bit 3, mhasseqnum bit
                              is set). All other bits are unused and
                              MUST be set to '0'.

   Message Address Length   - 0x0

   Message Size             - 22 + size of optional sub-TLVs.

   Message Sequence Number  - A 16-bit unsigned integer field containing
                              the sequence number from the Neighbor Up
                              Message that is being acknowledged.
```

TLV Block - TLV Length: 14 + optional sub-TLVs
DLEP Peer Update ACK order
Identification Sub-TLV (MANDATORY)
Status Sub-TLV (OPTIONAL)

13. Peer Termination Message

The Peer Termination Message is sent by either the client or the router when a session needs to be terminated. Transmission of a Peer Termination ACK message is required to confirm the termination process. The sender of the Peer Termination message is free to define its heuristics in event of a timeout. The receiver of a Peer Termination Message MUST terminate all neighbor relationships and release associated resources. No Neighbor Down messages are sent.

The Peer Termination Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =    |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE   | 0x1   | 0x0   |        22 + size of opt       |
  | (value TBD)    |       |       |            sub-TLVs           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num       |TLVs Length =14 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Peer      |TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
  | Termination    |               | opt sub-TLVs  | Identification|
  | (Value TDB)    |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8      |           Router ID           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID           |           Client ID           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID           |Sub-TLV type=  |TLV Flags=0x10 |
  |                                |DLEP Status    |               |
  |                                |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Length = 1     | Code          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 22 + size of optional sub-TLVs.

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - TLV Length = 14 + optional sub-TLVs
DLEP Peer Termination order
Identification Sub-TLV (MANDATORY)
Status Sub-TLV (OPTIONAL)

14. Peer Termination ACK Message

The Peer Termination Message ACK is sent by either the client or the router when a session needs to be terminated.

The Peer Termination ACK Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |        22 + size of opt       |
  | (value TBD)   |       |       |            sub-TLVs           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length =14 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Peer Term|TLV Flags=0x10 | Length = 11 + | Sub-TLV type= |
  | ACK           |               | opt sub-TLVs  | Identification|
  | (Value TBD)   |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP Status    |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Length = 1    | Code          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to ’0’.

Message Address Length - 0x0

Message Size - 22 + optional sub-TLVs.

Message Sequence Number   - A 16-bit unsigned integer field containing the sequence number in the corresponding Peer Termination Message being acknowledged.

TLV Block   - TLV Length = 14 + optional Sub-TLVs
DLEP Peer Termination ACK order
Identification Sub-TLV (MANDATORY)
Status Sub-TLV (OPTIONAL)

15. Neighbor Up Message

The client sends the Neighbor Up message to report that a new potential routing neighbor has been detected. A Neighbor Up ACK Message is required to confirm a received Neighbor Up. The sender of the Neighbor Up Message is free to define its retry heuristics in event of a timeout.

The Neighbor Up Message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   | DLEP_MESSAGE  | 0x1   | 0x0   |        31 + size of opt       |
   | (value TBD)   |       |       |            sub-TLVs           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |TLVs Length =23 + opt sub-TLVs |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Neighbor |TLV Flags=0x10 | Length =20 +  | Sub-TLV type= |
   | Up (TBD)      |               | opt sub-TLVs  | Identification|
   |               |               |               | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |          Router ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Router ID          |          Client ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP MAC       |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 6     |                  MAC Address                  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              MAC Address                      |Sub-TLV type=  |
   |                                               |DLEP IPv4 (TBD)|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 5     | Add/Drop Ind. | IPv4 Address  |
   |               |               |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                IPv4 Address                   |Sub-TLV type=  |
   |                                               |DLEP IPv6 (TBD)|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   (Continued on next page)                    |
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  (Continued from above)                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 17    | Add/Drop Ind. | IPv6 Address  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        IPv6 Address                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        IPv6 Address                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        IPv6 Address                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                IPv6 Address                   |Sub-TLV type=  |
   |                                               |DLEP MDR (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |         MDR (bps)             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        MDR (bps)                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        MDR (bps)              |sub-TLV Type = |TLV Flags=0x10 |
   |                               |DLEP CDR (TBD) |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 8     |                 CDR (bps)                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        CDR (bps)                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  CDR (bps)    |sub-TLV Type = |TLV Flags=0x10 | Length = 2    |
   |               |Latency (TBD)  |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
            Latency (ms)           |sub-TLV Type = |TLV Flags=0x10 |
   |                               |Resources(TBD) |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Length = 1    |   Resources   |sub-TLV Type = |TLV Flags=0x10 |
   |               |               |DLEP RLQ (TBD) |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Length = 1    |      RLQ      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Message Type             - DLEP_MESSAGE (Value TBD)

   Message Flags            - Set to 0x1 (bit 3, mhasseqnum bit
                              is set). All other bits are unused and
                              MUST be set to '0'.

   Message Address Length   - 0x0

   Message Size             - 31 + optional Sub-TLVs

   Message Sequence Number  - A 16-bit unsigned integer field containing
                              a sequence number generated by the message
                              originator.
   TLV Block                 - TLV Length:  23 + optional Sub-TLVs.
                              DLEP Neighbor Up order
                              Identification Sub-TLV (MANDATORY)
                              MAC Address Sub-TLV (MANDATORY)
                              IPv4 Address Sub-TLV (OPTIONAL)
                              IPv6 Address Sub-TLV (OPTIONAL)
                              Maximum Data Rate Sub-TLV (OPTIONAL)
                              Current Data Rate Sub-TLV (OPTIONAL)
                              Latency Sub-TLV (OPTIONAL)
                              Resources Sub-TLV (OPTIONAL)
                              Relative Link Factor Sub-TLV (OPTIONAL)
```

16. Neighbor Up ACK Message

The router sends the Neighbor Up ACK Message to indicate whether a Neighbor Up Message was successfully processed.

The Neighbor Up ACK message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |                35             |
  | (value TBD)   |       |       |                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length = 27               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Neighbor |TLV Flags=0x10 | Length = 24   | Sub-TLV type= |
  | Up ACK (TBD)  |               |               | Identification|
  |               |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP MAC       |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 6     |                MAC Address                    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              MAC Address                      |Sub-TLV type=  |
  |                                               |DLEP Status
  |                                               |(TBD)          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 | Length = 1     |  Code         |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Message Type              - DLEP_MESSAGE (Value TBD)
```

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to ’0’.

Message Address Length - 0x0

Message Size - 35

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number from the Neighbor Down Message that is being acknowledged.

TLV Block - TLV Length: 27

DLEP Neighbor Up ACK order
Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Status Sub-TLV (MANDATORY)

17. Neighbor Down Message

The client sends the Neighbor Down message to report when a neighbor is no longer reachable from the client. The Neighbor Down message MUST contain a MAC Address TLV. Any other TLVs present MAY be ignored. A Neighbor Down ACK Message is required to confirm the process. The sender of the Neighbor Down message is free to define its retry heuristics in event of a timeout.

The Neighbor Down Message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   |  DLEP_MESSAGE |  0x1  |  0x0  |        31 + optional          |
   |  (value TBD)  |       |       |            sub-TLV            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |  TLVs Length = 23 + optional  |
   |                               |             Sub-TLV           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  TLV Type =   |TLV Flags=0x10 | Length = 20 + | Sub-TLV type= |
   |  DLEP Neighbor|               | optional Sub- |Identification |
   |  Down (TBD)   |               | TLV           | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |          Router ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Router ID            |          Client ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Client ID            |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP MAC       |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   (Continued on next page)                    |
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   (Continued from above)                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 6     |                MAC Address                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              MAC Address                      |Sub-TLV type=  |
   |                                               |DLEP Status
   |                                               |(TBD)          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 | Length = 1    |  Code         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Message Type              - DLEP_MESSAGE (Value TBD)

    Message Flags             - Set to 0x1 (bit 3, mhasseqnum bit
                                 is set). All other bits are unused and
                                 MUST be set to '0'.

    Message Address Length    - 0x0

    Message Size              - 31 + optional TLVs

    Message Sequence Number   - A 16-bit unsigned integer field
                                 containing a sequence number generated
                                 by the message originator.

    TLV Block                 - TLV Length: 23 + optional Sub-TLVs

                                 DLEP Neighbor Down order
                                 Identification Sub-TLV (MANDATORY)
                                 MAC Address Sub-TLV (MANDATORY)
                                 Status Sub-TLV (OPTIONAL)
```

18. Neighbor Down ACK Message

The router sends the Neighbor Down ACK Message to indicate whether a Neighbor Down Message was successfully processed.

The Neighbor Down ACK message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size          |
   | DLEP_MESSAGE  | 0x1   | 0x0   |                35              |
   | (value TBD)   |       |       |                                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |TLVs Length = 27                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Neighbor |TLV Flags=0x10 | Length = 24   | Sub-TLV type= |
   | Down ACK (TBD)|               |               | Identification|
   |               |               |               | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |          Router ID             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Router ID          |          Client ID             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP MAC       |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 6     |                MAC Address                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              MAC Address                      |Sub-TLV type=  |
   |                                               |DLEP Status
   |                                               |(TBD)          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 | Length = 1    |  Code         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Message Type              - DLEP_MESSAGE (Value TBD)

   Message Flags             - Set to 0x1 (bit 3, mhasseqnum bit
                               is set). All other bits are unused and
                               MUST be set to '0'.

   Message Address Length    - 0x0

   Message Size              - 35

   Message Sequence Number   - A 16-bit unsigned integer field containing
                               the sequence number from the Neighbor Down
                               Message that is being acknowledged.

   TLV Block                 - TLV Length:  27

                               DLEP Neighbor Down ACK order
                               Identification Sub-TLV (MANDATORY)
                               MAC Address Sub-TLV (MANDATORY)
                               Status Sub-TLV (MANDATORY)
```

19. Neighbor Update Message

The client sends the Neighbor Update message when a change in link metric parameters is detected for a routing neighbor.

The Neighbor Update Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |         31 + optional         |
  | (value TBD)   |       |       |            sub-TLV            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |  TLVs Length = 23 + optional  |
  |                               |             Sub-TLVs          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Type =     |TLV Flags=0x10 |Length = 20 +  |Sub-TLV type = |
  |DLEP Neighbor  |               |optional Sub-  |Identification |
  |Update (TBD)   |               |TLVs           |Sub-TLV (TBD)  |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP MAC       |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 6     |                MAC Address                    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              MAC Address                      |Sub-TLV type=  |
  |                                               |DLEP MDR (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |           MDR (bps)           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        MDR (bps)                              |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |        MDR (bps)              |Sub-TLV Type = |TLV Flags=0x10 |
  |                               |DLEP CDR (TBD) |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 8     |                 CDR (bps)                     |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        CDR (bps)                              |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |   CDR (bps)   |Sub-TLV Type = |TLV Flags=0x10 | Length = 2    |
  |               |DLEP Latency   |               |               |
  |               |(TBD)          |               |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |        Latency (ms)           |Sub-TLV Type=  |TLV Flags=0x10 |
  |                               |DLEP Resources |               |
  |                               |(TBD)          |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                   (Continued on next page)                    |
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                    (Continued from above)                     |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Length = 1    |   Resources   |Sub-TLV Type=  |TLV FLags=0x10 |
  |               |               |DLEP RLQ (TBD) |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | Length = 1    |      RLQ      |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + optional TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

TLV Block - TLVs Length - 23 + optional Sub-TLVs.

DLEP Neighbor Update order
Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Maximum Data Rate Sub-TLV (OPTIONAL)
Current Data Rate Sub-TLV (OPTIONAL)
Latency Sub-TLV (OPTIONAL)
Resources Sub-TLV (OPTIONAL)
Relative Link Quality Sub-TLV (OPTIONAL)

20. Neighbor Address Update Message

The client sends the Neighbor Address Update message when a change in Layer 3 addressing is detected for a routing neighbor.

The Neighbor Address Update Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|         Message Size          |
  | DLEP_MESSAGE  |  0x1  |  0x0  |      31 + size of opt         |
  | (value TBD)   |       |       |          sub-TLVs             |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |         Message Seq Num       |TLVs Length =23 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                  (Continued on next page)                     |
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                  (Continued from above)                       |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Neighbor |TLV Flags=0x10 | Length =20 +  | Sub-TLV type= |
  | Address Update|               | opt sub-TLVs  | Identification|
  |(TBD)          |               |               | Sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP MAC       |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 6     |                MAC Address                    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              MAC Address                      |Sub-TLV type=  |
  |                                               |DLEP IPv4 (TBD)|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 5     | Add/Drop Ind. | IPv4 Address  |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                IPv4 Address                   |Sub-TLV type=  |
  |                                               |DLEP IPv6 (TBD)|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 17    | Add/Drop Ind. | IPv6 Address  |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        IPv6 Address                           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                IPv6 Address                   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + optional TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number, generated by the message originator.

```
   TLV Block                       - TLVs Length - 23 + optional Sub-TLVs.
                                     DLEP Neighbor Address Update order
                                     Identification Sub-TLV (MANDATORY)
                                     MAC Address Sub-TLV (MANDATORY)
                                     IPv4 Address Sub-TLV (OPTIONAL)
                                     IPv6 Address Sub-TLV (OPTIONAL)
```

21. Neighbor Address Update ACK Message

The router sends the Neighbor Address Update ACK Message to indicate whether a Neighbor Address Update Message was successfully processed.

The Neighbor Address Update ACK message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
  | DLEP_MESSAGE  | 0x1   | 0x0   |                35             |
  | (value TBD)   |       |       |                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length = 27               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Neighbor |TLV Flags=0x10 | Length = 24   | Sub-TLV type= |
  | Address Update|               |               | Identification|
  | ACK (TBD)     |               |               | sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |           Router ID           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |           Client ID           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP MAC       |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 6     |                MAC Address                    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              MAC Address                      |Sub-TLV type=  |
  |                                               |DLEP Status
  |                                               |(TBD)          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 | Length = 1    |  Code         |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   Message Type              - DLEP_MESSAGE (Value TBD)

   Message Flags             - Set to 0x1 (bit 3, mhasseqnum bit
                               is set). All other bits are unused and
                               MUST be set to '0'.

   Message Address Length    - 0x0
   Message Size            - 35

   Message Sequence Number  - A 16-bit unsigned integer field containing
                              the sequence number from the Neighbor Down
                              Message that is being acknowledged.

   TLV Block                - TLV Length:  27

                              DLEP Neighbor Address Update ACK order
                              Identification Sub-TLV (MANDATORY)
                              MAC Address Sub-TLV (MANDATORY)
                              Status Sub-TLV (MANDATORY)
```

22. Heartbeat Message

A Heartbeat Message is sent by a peer every N seconds, where N is defined in the "Heartbeat Interval" field of the discovery message. The message is used by peers to detect when a DLEP session partner is no longer communicating. Peers SHOULD allow some integral number of heartbeat intervals (default 4) to expire with no traffic on the session before initiating DLEP session termination procedures.

The Heartbeat Message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   | DLEP_MESSAGE  |  0x1  |  0x0  |               22              |
   | (value TBD)   |       |       |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |TLVs Length = 14               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Heartbeat|TLV Flags=0x10 | Length = 11   | Sub-TLV type= |
   | (TBD)         |               |               | Identification|
   |               |               |               | sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |          Router ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Router ID         |          Client ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Client ID         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Message Type            - DLEP_MESSAGE (Value TBD)

   Message Flags           - Set to 0x1 (bit 3, mhasseqnum bit is
                             set). All other bits are unused and SHOULD
                             be set to '0'.

   Message Address Length  - 0x0

   Message Size            - 22
   Message Sequence Number - A 16-bit unsigned integer field containing
                             a sequence number generated by the message
                             originator.

   TLV Block -               TLV Length = 14

                             DLEP Heartbeat order
                             Identification Sub-TLV (MANDATORY)
```

23. Link Characteristics Request Message

The Link Characteristics Request Message is sent by the router to the modem device when the router detects that a different set of transmission characteristics is necessary (or desired) for the type of traffic that is flowing on the link. The request contains either a Current Data Rate (CDR) TLV to request a different amount of bandwidth than what is currently allocated, a Latency TLV to request that traffic delay on the link not exceed the specified value, or both. A Link Characteristics ACK Message is required to complete the request. Implementations are free to define their retry heuristics in event of a timeout. Issuing a Link Characteristics Request with ONLY the MAC Address TLV is a mechanism a peer MAY use to request metrics (via the Link Characteristics ACK) from its partner.

The Link Characteristics Request Message contains the following fields:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Msg Type =   |Msg Flg|AddrLen|          Message Size          |
  | DLEP_MESSAGE  | 0x1   | 0x0   |        31 + size of opt        |
  | (value TBD)   |       |       |            sub-TLVs            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Message Seq Num      |TLVs Length =23 + opt sub-TLVs |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  | DLEP Link Char|TLV Flags=0x10 | Length =20 +  | Sub-TLV type= |
  | Request (TBD) |               | opt sub-TLVs  | Identification|
  |               |               |               | Sub-TLV (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |          Router ID             |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Router ID          |          Client ID             |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
  |                               |DLEP MAC       |               |
  |                               |sub-TLV (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 6     |                MAC Address                     |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                (Continued on next page)                       |
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                (Continued from above)                         |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              MAC Address                      |Sub-TLV type=  |
  |                                               |DLEP CDR (TBD) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |TLV Flags=0x10 |Length = 8     |           CDR (bps)           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                        CDR (bps)                              |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |        CDR (bps)              |Sub-TLV Type=  |TLV Flags=0x10 |
  |                               |Latency (TBD)  |               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |Length = 2     |         Latency (ms)            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to ’0’.

Message Address Length - 0x0

Message Size - 31 + length of optional (Current Data Rate and/or Latency) Sub-TLVs

Message Sequence Number - A 16-bit unsigned integer field containing a sequence number generated by the message originator.

TLV Block - Length: 23 + optional Sub-TLVs

DLEP Link Characteristics Request order Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)

Current Data Rate Sub-TLV - if present, this value represents the requested data rate in bits per second (bps). (OPTIONAL)

Latency TLV - if present, this value represents the maximum latency, in milliseconds, desired on the link. (OPTIONAL)

24. Link Characteristics ACK Message

The Link Characteristics ACK Message is sent by the client to the router letting the router know the success (or failure) of the requested change in link characteristics. The Link Characteristics ACK message SHOULD contain a complete set of metric TLVs. It MUST contain the same TLV types as the request. The values in the metric TLVs in the Link Characteristics ACK message MUST reflect the link characteristics after the request has been processed.

The Link Characteristics ACK Message contains the following fields:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg Type =   |Msg Flg|AddrLen|          Message Size         |
   | DLEP_MESSAGE  |  0x1  |  0x0  |       31 + size of opt        |
   | (value TBD)   |       |       |           sub-TLVs            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Message Seq Num      |TLVs Length =23 + opt sub-TLVs |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | DLEP Link Char|TLV Flags=0x10 | Length =20 +  | Sub-TLV type= |
   | ACK (TBD)     |               | opt sub-TLVs  | Identification|
   |               |               |               | Sub-TLV (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |          Router ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Router ID          |          Client ID            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Client ID          |Sub-TLV type=  |TLV Flags=0x10 |
   |                               |DLEP MAC       |               |
   |                               |sub-TLV (TBD)  |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 6     |                MAC Address                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              MAC Address                      |Sub-TLV type=  |
   |                                               |DLEP MDR (TBD) |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |TLV Flags=0x10 |Length = 8     |           MDR (bps)           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        MDR (bps)                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        MDR (bps)              |Sub-TLV Type=  |TLV Flags=0x10 |
   |                               |DLEP CDR (TBD) |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Length = 8     |                  CDR (bps)                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                           CDR (bps)                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  CDR (bps)    |Sub-TLV Type = |TLV Flags=0x10 | Length = 2    |
   |               |Latency (TBD)  |               |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Latency (ms)           |Sub-TLV Type=  |TLV Flags=0x10 |
   |                               |Resources (TBD)|               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Length = 1    |   Resources   |Sub-TLV Type=  |TLV Flags=0x10 |
   |               |               |RLQ (TBD)      |               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Length = 1    |      RLQ      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Message Type - DLEP_MESSAGE (Value TBD)

Message Flags - Set to 0x1 (bit 3, mhasseqnum bit is set). All other bits are unused and MUST be set to '0'.

Message Address Length - 0x0

Message Size - 31 + length of optional (Current Data Rate and/or Latency) TLVs

Message Sequence Number - A 16-bit unsigned integer field containing the sequence number that appeared on the corresponding Link Characteristics Request message.

TLV Block - TLVs Length = 23 + Optional TLVs

DLEP Link Characteristics ACK order
Identification Sub-TLV (MANDATORY)
MAC Address Sub-TLV (MANDATORY)
Maximum Data Rate Sub-TLV (OPTIONAL)

Current Data Rate Sub-TLV - if present, this value represents the NEW (or unchanged, if the request is denied) Current Data Rate in bits per second (bps). (OPTIONAL)

Latency Sub-TLV - if present, this value represents the NEW maximum latency (or unchanged, if the request is denied), expressed in milliseconds, on the link. (OPTIONAL)

Resources Sub-TLV (OPTIONAL)

Relative Link Quality Sub-TLV (OPTIONAL)

25. Security Considerations

The protocol does not contain any mechanisms for security (e.g. authentication or encryption). The protocol assumes that any security would be implemented in the underlying transport (for example, by use of DTLS or some other mechanism), and is therefore outside the scope of this document.

26. IANA Considerations

This section specifies requests to IANA.

26.1 TLV Registrations

This specification defines:

- One TLV types which must be allocated from the 0-223 range of the "Assigned Message TLV Types" repository of [RFC5444].

- A new repository for DLEP orders, with seventeen values currently assigned.

- A new repository for DLEP Sub-TLV assignments with fifteen values currently assigned.

26.2 Expert Review: Evaluation Guidelines

For the registries for TLV type extensions where an Expert Review is required, the designated expert SHOULD take the same general recommendations into consideration as are specified by [RFC5444].

26.3 Message TLV Type Registration

The Message TLV specified below must be allocated from the "Message TLV Types" namespace of [RFC5444].

- DLEP_MESSAGE

26.4 DLEP Order Registration

A new repository must be created with the values of the DLEP orders. Valid orders are:

- Attached Peer Discovery Message
- Detached Peer Discovery Message
- Peer Offer Message
- Peer Update Message
- Peer Update ACK Message
- Peer Termination Message
- Peer Termination ACK Message
- Neighbor Up Message
- Neighbor Up ACK Message
- Neighbor Down Message
- Neighbor Down ACK Message
- Neighbor Update Message
- Neighbor Address Update Message
- Neighbor Address Update ACK Message
- Heartbeat Message
- Link Characteristics Request Message
- Link Characteristics ACK Message

This registry should be created according to the guidelines for 'Message-Type-Specific TLV' registration as specified in section 6.2.1 of [RFC5444].

26.5 DLEP Sub-TLV Type Registrations

A new repository for DLEP Sub-TLVs must be created. Valid Sub-TLVs are:

- Identification Sub-TLV
- DLEP Version Sub-TLV
- Peer Type Sub-TLV
- MAC Address Sub-TLV
- IPv4 Address Sub-TLV
- IPv6 Address Sub-TLV
- Maximum Data Rate Sub-TLV
- Current Data Rate Sub-TLV
- Latency Sub-TLV
- Resources Sub-TLV
- Relative Link Quality Sub-TLV
- Status Sub-TLV
- Heartbeat Interval Sub-TLV
- Heartbeat Threshold Sub-TLV
- Link Characteristics ACK Timer Sub-TLV

It is also requested that the registry allocation contain space reserved for experimental sub-TLVs.

27. Appendix A.

Peer Level Message Flows

*Modem Device (Client) Restarts Discovery

```
   Router                     Client   Message Description
   ====================================================================

   <-------Peer Discovery---------    Modem initiates discovery


    ---------Peer Offer----------->   Router detects a problem, sends
      w/ Non-zero Status TLV          Peer Offer w/ Status TLV indicating
                                      the error.

                                      Modem accepts failure, restarts
                                      discovery process.

   <-------Peer Discovery---------    Modem initiates discovery


    ---------Peer Offer----------->   Router accepts, sends Peer Offer
         w/ Zero Status TLV           w/ Status TLV indicating success.

                                      Discovery completed.
```

*Modem Device Detects Peer Offer Timeout

```
   Router                     Client   Message Description
   ====================================================================

   <-------Peer Discovery---------    Modem initiates discovery,
                                      starts a guard timer.

                                      Modem guard timer expires.
                                      Modem restarts discovery process.

    <-------Peer Discovery---------   Modem initiates discovery,
                                      starts a guard timer.

    ---------Peer Offer----------->   Router accepts, sends Peer Offer
         w/ Zero Status TLV           w/ Status TLV indicating success.

                                      Discovery completed.
```

*Router Peer Offer Lost

```
   Router                    Client   Message Description
   ====================================================================

   <-------Peer Discovery---------    Modem initiates discovery,
                                      starts a guard timer.

    ---------Peer Offer-------||      Router offers availability

                                      Modem times out on Peer Offer,
                                      restarts discovery process.

   <-------Peer Discovery---------    Modem initiates discovery

    ---------Peer Offer----------->   Router detects subsequent discovery,
                                      internally terminates the previous,
                                      accepts the new association, sends
                                      Peer Offer w/ Status TLV indicating
                                      success.


                                      Discovery completed.
```

*Discovery Success

```
   Router                    Client   Message Description
   ====================================================================

   <-------Peer Discovery---------    Modem initiates discovery

    ---------Peer Offer----------->   Router offers availability

    -------Peer Heartbeat--------->

   <-------Peer Heartbeat---------

    -------Peer Heartbeat--------->

   <==============================>   Neighbor Sessions

   <-------Peer Heartbeat---------

    -------Peer Heartbeat--------->

    --------Peer Term Req--------->   Terminate Request

   <--------Peer Term Res---------    Terminate Response
```

*Router Detects a Heartbeat timeout

```
   Router                    Client   Message Description
   ====================================================================

   <-------Peer Heartbeat---------

    -------Peer Heartbeat--------->

      ||---Peer Heartbeat---------

           ~ ~ ~ ~ ~ ~ ~

    -------Peer Heartbeat--------->

      ||---Peer Heartbeat---------
                                      Router Heartbeat Timer expires,
                                      detects missing heartbeats. Router
                                      takes down all neighbor sessions
                                      and terminates the Peer association.

    ------Peer Terminate --------->   Peer Terminate Request

                                      Modem takes down all neighbor
                                      sessions, then acknowledges the
                                      Peer Terminate

   <----Peer Terminate ACK---------   Peer Terminate ACK
```

*Modem Detects a Heartbeat timeout

```
   Router                    Client   Message Description
   ====================================================================

   <-------Peer Heartbeat---------

    -------Peer Heartbeat------||

   <-------Peer Heartbeat---------

           ~ ~ ~ ~ ~ ~ ~

    -------Peer Heartbeat------||

   <-------Peer Heartbeat---------
                                      Modem Heartbeat Timer expires,
                                      detects missing heartbeats. Modem
                                      takes down all neighbor sessions
                                      and terminates the Peer association.
    <-------Peer Terminate--------    Peer Terminate Request

                                      Router takes down all neighbor
                                      sessions, then acknowledges the
                                      Peer Terminate

    ------Peer Terminate ACK----->    Peer Terminate ACK
```

*Peer Terminate (from Modem) Lost

```
   Router                    Client   Message Description
   =====================================================================

     ||------Peer Terminate--------   Modem Peer Terminate Request

                                      Router Heartbeat times out,
                                      terminates association.

    --------Peer Terminate------->    Router Peer Terminate

    <-----Peer Terminate ACK------    Modem sends Peer Terminate ACK
```

*Peer Terminate (from router) Lost

```
   Router                    Client   Message Description
   =====================================================================

    -------Peer Terminate-------->    Router Peer Terminate Request

                                      Modem HB times out,
                                      terminates association.

    <------Peer Terminate--------     Modem Peer Terminate

    ------Peer Terminate ACK----->    Peer Terminate ACK
```

Neighbor Level Message Flows

*Modem Neighbor Up Lost

```
   Router                    Client   Message Description
   ====================================================================

    ||-----Neighbor Up ------------   Modem sends Neighbor Up

                                      Modem timesout on ACK

    <------Neighbor Up ------------   Modem sends Neighbor Up

    ------Neighbor Up ACK--------->   Router accepts the neighbor
                                      session

   <------Neighbor Update---------    Modem Neighbor Metrics
          . . . . . . . .
   <------Neighbor Update---------    Modem Neighbor Metrics
```

*Router Detects Duplicate Neighbor Ups

```
   Router                    Client   Message Description
   ====================================================================

    <------Neighbor Up ------------   Modem sends Neighbor Up

    ------Neighbor Up ACK-------||    Router accepts the neighbor
                                      session

                                      Modem timesout on ACK

    <------Neighbor Up ------------   Modem resends Neighbor Up

                                      Router detects duplicate
                                      Neighbor, takes down the
                                      previous, accepts the new
                                      Neighbor.

    ------Neighbor Up ACK--------->   Router accepts the neighbor
                                      session

   <------Neighbor Update---------    Modem Neighbor Metrics
          . . . . . . . .
   <------Neighbor Update---------    Modem Neighbor Metrics
```

*Neighbor Up, No Layer 3 Addresses

```
   Router                    Client    Message Description
   ====================================================================

    <-------Neighbor Up ------------   Modem sends Neighbor Up

    ------Neighbor Up ACK--------->   Router accepts the neighbor
                                      session

                                      Router ARPs for IPv4 if defined.
                                      Router drives ND for IPv6 if
                                      defined.

   <------Neighbor Update---------    Modem Neighbor Metrics
          . . . . . . . .
   <------Neighbor Update---------    Modem Neighbor Metrics
```

*Neighbor Up with IPv4, No IPv6

```
   Router                    Client   Message Description
   ====================================================================

    <------Neighbor Up ------------   Modem sends Neighbor Up with
                                      the IPv4 TLV

    ------Neighbor Up ACK--------->   Router accepts the neighbor
                                      session

                                      Router drives ND for IPv6 if
                                      defined.

   <------Neighbor Update---------    Modem Neighbor Metrics
          . . . . . . . .
   <------Neighbor Update---------    Modem Neighbor Metrics
```

*Neighbor Up with IPv4 and IPv6

```
   Router                    Client   Message Description
   ====================================================================

    <-------Neighbor Up ------------   Modem sends Neighbor Up with
                                      the IPv4 and IPv6 TLVs

    ------Neighbor Up ACK--------->   Router accepts the neighbor
                                      session

   <------Neighbor Update---------    Modem Neighbor Metrics
          . . . . . . . .
   <------Neighbor Update---------    Modem Neighbor Metrics
```

*Neighbor Session Success

```
   Router                    Client   Message Description
   =====================================================================


    ---------Peer Offer----------->   Router offers availability

    -------Peer Heartbeat--------->


   <------Neighbor Up -----------      Modem

    ------Neighbor Up ACK-------->     Router

   <------Neighbor Update---------     Modem
          . . . . . . . .
   <------Neighbor Update---------     Modem

                                       Modem initiates the terminate

   <------Neighbor Down ----------     Modem

    ------Neighbor Down ACK------->    Router

                                       or

                                       Router initiates the terminate

    ------Neighbor Down ---------->    Router

   <------Neighbor Down ACK-------     Modem
```

Acknowledgements

The authors would like to acknowledge the influence and contributions of Chris Olsen and Teco Boot.

Normative References

[RFC5444] Clausen, T., Ed,. "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, Februar, 2009.

[RFC5578] Berry, B., Ed., "PPPoE with Credit Flow and Metrics", RFC 5578, February 2010.

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

Informative References

[DTLS] Rescorla, E., Ed,. "Datagram Transport Layer Security", RFC 4347, April 2006.

Author's Addresses

Stan Ratliff
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: sratliff@cisco.com

Bo Berry
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: boberry@cisco.com

Greg Harrison
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
EMail: greharri@cisco.com

Shawn Jury
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
Email: sjury@cisco.com

Darryl Satterwhite
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA
Email: dsatterw@cisco.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: July 23, 2011

S. Harnedy
Booz Allen Hamilton
R. Cole
US Army CERDEC
I. Chakeres
CenGen

January 19, 2011

Definition of Managed Objects for the DYMO Manet Routing Protocol
draft-ietf-manet-dymo-mib-04

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of the DYMO routing process. The DYMO-MIB also reports state information, performance information, and notifications. In addition to configuration, this additional state, performance and notification information is useful to management operators troubleshooting DYMO routing problems.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of a Dynamic MANET On-demand (DYMO) routing [I-D.ietf-manet-dymo] process. The DYMO-MIB also reports state information, performance metrics, and notifications. In addition to configuration, this additional state, performance and notification information is useful to management stations troubleshooting routing problems.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Conventions

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].

4. Overview

The Dynamic MANET On-demand (DYMO) routing protocol [I-D.ietf-manet-dymo] is intended for use by mobile nodes in wireless, multihop networks. DYMO determines unicast routes among DYMO routers within the network in an on-demand fashion, offering improved convergence in dynamic topologies.

A DYMO router's MIB contains DYMO process configuration parameters (e.g. interfaces), state information (e.g. sequence number), performance counters (e.g. number of control messages), and notifications.

4.1. DYMO Management Model

This section describes the management model for the DYMO routing protocol.

The MIB is comprised of four groups, i.e., Notifications, Configuration, State and Performance. The configuration of the managed devices is controlled by the objects in the Configuration Group. These are divided into Nodal and Interface objects. The bulk of the DYMO configuration is in the Nodal objects which control protocol behavior. The Interface objects merely identify/configure interfaces to enable DYMO routing over their interface. The Nodal objects are further divided into routing (or protocol) objects and Gateway objects. Gateway objects define other routing prefixes for which the node acts as a routing proxy on behalf of these non-local prefixes.

The Configuration Objects drive the behavior of the managed DYMO device and hence determines the information in the remaining groups, i.e., State, Performance and Notifications. The State objects primarily present the resulting forwarding table objects. The Performance group primarily is comprised of counters for monitoring the number of DYMO routing messages received locally, per node and per interface. The Notifications group contains objects which monitor changes to the interface configuration and the gateway prefixes configuration.

See the below diagram outlining the DYMO-MIB device management model.

```
   +-------------------------------------------+
   |  CONFIGURATION GROUP                      |
   |                                           |
   |  Nodal                        Interface   |
   |                                           |
   |  +-------+  +-------+         +-------+   |
   |  |Gateway|  |Routing|         |Routing|   |
   |  +-------+  +-------+         +-------+   |
   |                                           |
   +-------------------------------------------+
        ||           ||                 ||
        ||           ||                 ||
        ||           ||                 ||
       \  /         \  /               \  /
        \/           \/                 \/
   +-------+  +-------------+  +---------------+
   | STATE |  | PERFORMANCE |  | NOTIFICATIONS |
   | GROUP |  | GROUP       |  | GROUP         |
   +-------+  +-------------+  +---------------+
```

4.2. Terms

The following definitions apply throughout this document:

- Configuration Objects - switches, tables, objects which are initialized to default settings or set through the management interface defined by this MIB.

- Tunable Configuration Objects - objects whose values affect timing or attempt bounds on the DYMO protocol.

- State Objects - automatically generated values which define the current operating state of the DYMO protocol process in the router.

- Performance Objects - automatically generated values which help an operator or automated tool to assess the performance of the DYMO protocol process on the router and the overall routing performance within the DYMO routing domain.

5. Structure of the MIB Module

This section presents the structure of the DYMO MIB module. The objects are arranged into the following groups:

- dymoMIBNotifications - defines the notifications associated with the DYMO-MIB. These are currently limited to notifications of interface state changes and gateway prefix changes.

- o dymoMIBObjects - defines the objects forming the basis for the DYMO-MIB. These objects are divided up by function into the following groups:

- o

  - * Configuration Group - This group contains the DYMO objects that configure specific options that determine the overall performance and operation of the routing protocol for the router device and its interfaces.

  - * State Group - Contains information describing the current state of the DYMO process such as the DYMO routing table.

  - * Performance Group - Contains objects which help to characterize the performance of the DYMO process, typically statistics counters. There are two types of DYMO statistics: global counters and per interface counters.

- o dymoMIBConformance - defines minimal and full conformance of implementations to this DYMO-MIB.

5.1. Textual Conventions

The textual conventions used in the DYMO-MIB are as follows. The RowStatus and TruthValue textual conventions are imported from RFC 2579 [RFC2579]. The DymoInterfaceOperStatus is defined within the DYMO-MIB. This contains the current operational status of the DYMO interface.

5.2. The Configuration Group

The DYMO device is configured with a set of controls. The list of configuration controls for the DYMO device follow.

Protocol Configuration Parameters:

- o DID

- o MSG_HOPLIMIT

- o ROUTE_TIMEOUT

- o ROUTE_AGE_MIN_TIMEOUT

- o ROUTE_SEQNUM_AGE_MAX_TIMEOUT

- ROUTE_USED_TIMEOUT

- ROUTE_DELETE_TIMEOUT

- ROUTE_RREQ_WAIT_TIME

- UNICAST_MESSAGE_SENT_TIMEOUT

- MSG_HOPLIMIT

- DISCOVERY_ATTEMPTS_MAX

Protocol Configuration Tables:

- Responsible Hosts - If RESPONSIBLE_ADDRESSES is set to other than self address, then the DYMO router must be configured with the set of host addresses for which it is to generate RREP messages.

- Interfaces - If DYMO_INTERFACES is set to other than all, then the DYMO router must be told which interfaces to run the DYMO protocol over. This is a table containing the interfaces and associated information.

5.3. The State Group

The State Subtree reports current state information. State information from the DYMO-MIB is primarily contained in the 'Routing' Table.

5.3.1. Routing Table

The DYMO routing table contains information related to IP forwarding entries found by the node's DYMO processes.

5.4. The Performance Group

The Performance subtree reports primarily counters that relate to DYMO protocol activity. The DYMO performance objects consists of per node and per interface objects:

- OwnSequenceNumber

- RREQ initiated

- RREQ sent

- RREQ received

- RREP initiated
- RREP sent
- RREP received
- RRER initiated
- RRER sent
- RRER received
- Per interface statistics table with the following entries:
-
    - RREQ initiated
    - RREQ sent
    - RREQ received
    - RREP initiated
    - RREP sent
    - RREP received
    - RRER initiated
    - RRER sent
    - RRER received

5.5. The Notifications Group

The Notifications Subtree contains the list of notifications supported within the DYMO-MIB and their intended purpose or utility. This group is currently contains two notification objects, one related to status changes in DYMO interfaces and one related to changes in the gateway prefixes table.

6. Relationship to Other MIB Modules

The text of this section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. Definitions imported from other MIB modules and other MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

6.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The DYMO-MIB does not duplicate those objects.

6.2. MIB modules required for IMPORTS

The DYMO-MIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF [RFC2580], INET-ADDRESS-MIB [RFC4001] and IF-MIB [RFC2863].

7. Definitions

```
   MANET-DYMO-MIB DEFINITIONS ::= BEGIN

   IMPORTS

      MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
      Counter32, Integer32, Unsigned32, mib-2
         FROM SNMPv2-SMI                          -- [RFC2578]

      TEXTUAL-CONVENTION, RowStatus, TruthValue
         FROM SNMPv2-TC                           -- [RFC2579]

      MODULE-COMPLIANCE, OBJECT-GROUP,
      NOTIFICATION-GROUP
         FROM SNMPv2-CONF                         -- [RFC2580]

      InetAddress, InetAddressType,
      InetAddressPrefixLength
         FROM INET-ADDRESS-MIB                    -- [RFC4001]

      InterfaceIndexOrZero
         FROM IF-MIB                              -- [RFC2863]
      ;

   manetDymoMIB MODULE-IDENTITY
      LAST-UPDATED "201101191200Z"  -- January 19, 2011
      ORGANIZATION "IETF MANET Working Group"
      CONTACT-INFO
         "WG E-Mail: manet@ietf.org
```

```
           WG Chairs: ian.chakeres@gmail.com
                      jmacker@nrl.navy.mil

           Editors:   Sean Harnedy
                      Booz Allen Hamilton
                      333 City Boulevard West
                      Orange, CA 92868
                      USA
                      +1 714 938-3898
                      harnedy_sean@bah.com

                      Robert G. Cole
                      US Army CERDEC
                      Space and Terrestrial Communications
                      328 Hopkins Road
                      Aberdeen Proving Ground, MD 21005
                      USA
                      +1 410 278-6779
                      robert.g.cole@us.army.mil

                      Ian D Chakeres
                      CenGen
                      9250 Bendix Road North
                      Columbia, Maryland  21045
                      USA
                      ian.chakeres@gmail.com"

      DESCRIPTION
         "This MIB module contains managed object definitions for
          the Dynamic MANET On-demand (DYMO) routing protocol as
          defined in: Chakeres,I., and C. Perkins, Dynamic MANET
          On-demand (DYMO) Routing, draft-ietf-manet-dymo-21,
          July 26, 2010.

          Copyright (C) The IETF Trust (2008). This version
          of this MIB module is part of RFC xxxx; see the RFC
          itself for full legal notices."

      -- Revision History
      REVISION    "201101191200Z"   -- January 19, 2011
      DESCRIPTION
         "Fifth draft of this MIB module published as
          draft-ietf-manet-dymo-mib-04.txt.
          Changes include:
          - Incorporated the DYMO ID by adding Instance
            Table.
          - Added dymoSetNotification for improved control
            of DYMO Notifications.
          - Updated various object names to be consistent
            with current draft-ietf-manet-dymo-21.
         "
      REVISION    "200910251200Z"   -- October 25, 2009
      DESCRIPTION
         "Fourth draft of this MIB module published as
          draft-ietf-manet-dymo-mib-03.txt.
          - Minor changes to textual material, including
            additions to the IMPORTS text.
          - Added DEFVAL clauses to all read-write
            configuration objects with defaults identified
            in the DYMO draft."
      REVISION    "200902241200Z"   -- February 24, 2009
      DESCRIPTION
         "Third draft of this MIB module published as
          draft-ietf-manet-dymo-mib-02.txt.
          - Minor changes to dymoInterfacesTable and
            dymoResponsibleAddrTable.
          - Added global dymoAdminStatus and interface
            specific dymoIfAdminStatus.
          - Imported InterfaceIndexOrZero type from
            IF-MIB."
      REVISION    "200811031200Z"   -- November 03, 2008
      DESCRIPTION
         "Second draft of this MIB module published as
          draft-ietf-manet-dymo-mib-01.txt. Minor changes to
          dymoInterfacesTable and dymoResponsibleAddrTable."
      REVISION    "200805141200Z"   -- May 14, 2008
      DESCRIPTION
         "Initial draft of this MIB module published as
          draft-ietf-manet-dymo-mib-00.txt."
      -- RFC-Editor assigns XXXX
      ::= { mib-2 999 }   -- to be assigned by IANA

   --
   -- TEXTUAL CONVENTIONs
   --

   Status ::= TEXTUAL-CONVENTION
       STATUS       current
       DESCRIPTION
          "An indication of the operability of a DYMO
          function or feature.  For example, the status
          of an interface: 'enabled' indicates that
          it is willing to communicate with other DYMO routers,
          and 'disabled' indicates that it is not."
       SYNTAX  INTEGER { enabled (1), disabled (2) }
   --
   -- Top-Level Object Identifier Assignments
   --

   dymoMIBNotifications OBJECT IDENTIFIER ::= { manetDymoMIB 0 }
   dymoMIBObjects       OBJECT IDENTIFIER ::= { manetDymoMIB 1 }
   dymoMIBConformance   OBJECT IDENTIFIER ::= { manetDymoMIB 2 }

   --
   -- dymoConfigurationGroup
   --
   --    This group contains the DYMO objects that configure specific
   --    options that determine the overall performance and operation
   --    of the routing protocol for the router device and its
   --    interfaces.
   --

   dymoConfigurationGroup  OBJECT IDENTIFIER ::= { dymoMIBObjects 1 }

   --
   -- DYMO Global Router Configuration Group
   --

  dymoRouterConfigGroup OBJECT IDENTIFIER ::= {dymoConfigurationGroup 1}

   dymoInstanceTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF DymoInstanceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The DYMO Instance Table describes the DYMO
          ...."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. The DID."
   ::= { dymoRouterConfigGroup 1 }

   dymoInstanceEntry OBJECT-TYPE
      SYNTAX      DymoInstanceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The DYMO instance entry describes one DYMO
          process as indexed by its DID."
      INDEX { dymoInstanceIndex }
   ::= { dymoInstanceTable 1 }

   DymoInstanceEntry ::=
      SEQUENCE {
         dymoInstanceIndex
            Integer32,
         dymoInstanceDid
            Integer32,
         dymoInstanceAdminStatus
            Status,
         dymoInstanceRowStatus
            RowStatus
         }

   dymoInstanceIndex  OBJECT-TYPE
      SYNTAX      Integer32 (0..255)
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The instance index for this DYMO process."
      ::= { dymoInstanceEntry 1 }

   dymoInstanceDid OBJECT-TYPE
       SYNTAX       Integer32 (0..255)
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
          "The DYMO ID of this instance of the
           DYMO process.
          "
   ::= { dymoInstanceEntry 2 }

   dymoInstanceAdminStatus OBJECT-TYPE
       SYNTAX       Status
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
          "The administrative status of this DYMO
          process in the router.  Multiple processes are
          allowed.  The value 'enabled' denotes that the
          DYMO Process is active on at least one interface;
          'disabled' disables it on all interfaces.

          This object is persistent and when written
          the entity SHOULD save the change to non-volatile storage."
   ::= { dymoInstanceEntry 3 }

   dymoInstanceRowStatus  OBJECT-TYPE
      SYNTAX      RowStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "This object permits management of the table
          by facilitating actions such as row creation,
          construction, and destruction. The value of
          this object has no effect on whether other
          objects in this conceptual row can be
          modified."
   ::= { dymoInstanceEntry 4 }

   dymoMaxHopLimit  OBJECT-TYPE
      SYNTAX      Unsigned32 (0..255)
      UNITS       "hops"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The maximum number of hops. The suggested value
          default is 10 hops. This is the DYMO MSG_HOPLIMIT
          parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 10 }
   ::= { dymoRouterConfigGroup 2 }

   dymoRouteTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The route timeout value. The suggested default
          value is 5000 milliseconds. This is the
          DYMO ROUTE_TIMEOUT parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 5000 }
   ::= { dymoRouterConfigGroup 3 }

   dymoRouteAgeMinTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The minimum route age timeout value. The
```

```
          suggested default value is 1000 milliseconds.
          This is the DYMO ROUTE_AGE_MIN_TIMEOUT parameter
          value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 1000 }
   ::= { dymoRouterConfigGroup 4 }

   dymoRouteSeqnumAgeMaxTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The maximum route age timeout value. The
          suggested default value is 60,000 milliseconds.
          This is the DYMO ROUTE_SEQNUM_AGE_MAX_TIMEOUT
          parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 60000 }
   ::= { dymoRouterConfigGroup 5 }

   dymoRouteUsedTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The route used timeout value. The
          suggested default value is to set this
          to the dymoRouteTimeout object value
          (whose default is 5000 milliseconds). This
          is the DYMO ROUTE_USED_TIMEOUT parameter
          value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 5000 }
   ::= { dymoRouterConfigGroup 6 }

   dymoRouteDeleteTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The route delete timeout value. The
          suggested default value is 2 * dymoRouteTimeout
          value (which is equal to 10000 milliseconds
          if using the default value for the
          dymoRouteTimeout value). This is the
          DYMO ROUTE_DELETE_TIMEOUT parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 10000 }
   ::= { dymoRouterConfigGroup 7 }

   dymoRouteRreqWaitTime  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The Route Request wait time. The suggested default
          value is 2000 milliseconds. This is the DYMO
          ROUTE_RREQ_WAIT_TIME parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 2000 }
   ::= { dymoRouterConfigGroup 8 }

   dymoDiscoveryAttemptsMax  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..16)
      UNITS       "attempts"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The number of Route Request retry attempts. The
          suggested default value is 3. This is the
          DYMO DISCOVERY_ATTEMPTS_MAX parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 3 }
   ::= { dymoRouterConfigGroup 9 }
   dymoUnicastMsgSentTimeout  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The message sent timeout value for unicast packets.
          The suggested default value is 1000 milliseconds.
          This is the DYMO UNICAST_MESSAGE_SENT_TIMEOUT
          parameter value."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
      DEFVAL { 1000 }
   ::= { dymoRouterConfigGroup 10 }



   --
   -- DYMO Interfaces Configuration Table
   --

   dymoInterfaceTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF DymoInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The DYMO Interface Table describes the DYMO
          interfaces that are participating in the
          DYMO routing protocol. The ifIndex is from
          the interfaces group defined in the Interfaces
          Group MIB."
      REFERENCE
         "RFC 2863 - The Interfaces Group MIB, McCloghrie,
          K., and F. Kastenholtz, June 2000."
   ::= { dymoConfigurationGroup 2 }

   dymoInterfaceEntry OBJECT-TYPE
      SYNTAX      DymoInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The DYMO interface entry describes one DYMO
          interface as indexed by its ifIndex."
      INDEX { dymoIfIndex }
   ::= { dymoInterfaceTable 1 }
   DymoInterfaceEntry ::=
      SEQUENCE {
         dymoIfIndex
            InterfaceIndexOrZero,
         dymoIfAdminStatus
            Status,
         dymoIfRowStatus
            RowStatus
         }

   dymoIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The ifIndex for this DYMO interface."
      ::= { dymoInterfaceEntry 1 }

   dymoIfAdminStatus OBJECT-TYPE
      SYNTAX      Status
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
          "The DYMO interface's administrative status.
          The value 'enabled' denotes that the interface
          is running the DYMO routing protocol.
          The value 'disabled' denotes that the interface is
          external to DYMO."
      ::= { dymoInterfaceEntry 2 }

   dymoIfRowStatus  OBJECT-TYPE
      SYNTAX      RowStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "This object permits management of the table
          by facilitating actions such as row creation,
          construction, and destruction. The value of
          this object has no effect on whether other
          objects in this conceptual row can be
          modified."
   ::= { dymoInterfaceEntry 3 }


   --
   -- DYMO Responsible Address Table
   --
   dymoResponsibleAddrTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF DymoResponsibleAddrEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The DYMO Responsible Address Table is a
          list of IP address prefixes, and their
          associated prefix length for which the
          DYMO router is responsible."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoConfigurationGroup 3 }

   dymoResponsibleAddrEntry  OBJECT-TYPE
      SYNTAX      DymoResponsibleAddrEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "A single host address range. Information
          in this table is persistent and when this object
          is written, the entity SHOULD save the change to
          non-volatile storage."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
      INDEX { dymoResponsibleAddrIndex }
   ::= { dymoResponsibleAddrTable 1 }

   DymoResponsibleAddrEntry ::=
      SEQUENCE {
         dymoResponsibleAddrIndex
            Unsigned32,
         dymoResponsibleAddrType
            InetAddressType,
         dymoResponsibleAddr
            InetAddress,
         dymoResponsibleAddrPrefixLen
            InetAddressPrefixLength,
         dymoResponsibleAddrRowStatus
            RowStatus
         }

   dymoResponsibleAddrIndex  OBJECT-TYPE
      SYNTAX      Unsigned32
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "This object is the index into this table."
   ::= { dymoResponsibleAddrEntry 1 }

   dymoResponsibleAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "The type of the dymoResponsibleAddr, as defined
          in the InetAddress MIB [RFC 4001]."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoResponsibleAddrEntry 2 }

   dymoResponsibleAddr  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "The destination IP address of this route. The type
          of this address is determined by the value of the
          dymoResponsibleAddrType object."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoResponsibleAddrEntry 3 }

   dymoResponsibleAddrPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "Indicates the number of leading one bits that form the
         mask to be logical-AND'd with the destination address
         before being compared to the value in the dymoResonsibleAddr
          field."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoResponsibleAddrEntry 4 }

   dymoResponsibleAddrRowStatus  OBJECT-TYPE
```

```
      SYNTAX      RowStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "This object permits management of the table
          by facilitating actions such as row creation,
          construction, and destruction. The value of
          this object has no effect on whether other
          objects in this conceptual row can be
          modified."
   ::= { dymoResponsibleAddrEntry 5 }


   --
   -- dymoStateGroup
   --
   --    Contains information describing the current state of the DYMO
   --    process such as the DYMO routing table.
   --

   dymoStateGroup  OBJECT IDENTIFIER ::= { dymoMIBObjects 2 }

   dymoCurrentSeqNum  OBJECT-TYPE
      SYNTAX      Unsigned32 (1..65535)
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The current DYMO sequence number. The DYMO sequence
          numbers allow nodes to judge the freshness of routing
          information and ensures loop freedom. If the sequence
          number has been assigned to be the largest possible
          number representable as a 16-bit unsigned integer
          (i.e., 65,535), then the sequence number is set to
          256 when incremented.  Setting the sequence number
          to 256 allows other nodes to detect that the number
          has rolled over and the node has not lost its sequence
          number (e.g., via reboot)."
   ::= { dymoStateGroup 1 }

   --
   -- DYMO Routing Table
   --

   dymoRoutingTable  OBJECT-TYPE
      SYNTAX       SEQUENCE OF DymoRoutingEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
```

```
         "The DYMO Routing Table describes the
          current routing information learned
          via DYMO control messages."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 2 Suggested
          Parameter Values."
   ::= { dymoStateGroup 2 }

   dymoRoutingEntry  OBJECT-TYPE
      SYNTAX       DymoRoutingEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The DYMO routing entry contains a
          piece of routing information for a
          particular set of addresses."
      INDEX { dymoRoutingIpAddrType,
              dymoRoutingIpAddr,
              dymoRoutingPrefixLen }
   ::= { dymoRoutingTable 1 }

   DymoRoutingEntry ::=
      SEQUENCE {
         dymoRoutingIpAddrType
            InetAddressType,
         dymoRoutingIpAddr
            InetAddress,
         dymoRoutingPrefixLen
            InetAddressPrefixLength,
         dymoRoutingSeqNum
            Unsigned32,
         dymoRoutingNextHopIpAddrType
            InetAddressType,
         dymoRoutingNextHopIpAddress
            InetAddress,
         dymoRoutingNextHopInterface
            InterfaceIndexOrZero,
         dymoRoutingForwardingFlag
            TruthValue,
         dymoRoutingBrokenFlag
            TruthValue,
         dymoRoutingDist
            Unsigned32
         }

   dymoRoutingIpAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The routing table address IP address type."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoRoutingEntry 1 }

   dymoRoutingIpAddr  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The routing table Inet IPv4 or IPv6 address."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoRoutingEntry 2 }

   dymoRoutingPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The prefix length. This is a decimal value that
          indicates the number of contiguous, higher-order
          bits of the address that make up the network
          portion of the address."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing, Chakeres,
          I., and C. Perkins, July 2010. Table 3 Important
          Settings."
   ::= { dymoRoutingEntry 3 }

   dymoRoutingSeqNum  OBJECT-TYPE
      SYNTAX       Unsigned32 (1..65535)
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The interface sequence number. This
          is the DYMO SeqNum associated with this
          routing information."
   ::= { dymoRoutingEntry 4 }

   dymoRoutingNextHopIpAddrType OBJECT-TYPE
```

```
      SYNTAX      InetAddressType
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The IP address type of the next hop."
      ::= { dymoRoutingEntry 5 }

   dymoRoutingNextHopIpAddress OBJECT-TYPE
      SYNTAX       InetAddress
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The IP address of the next hop."
      ::= { dymoRoutingEntry 6 }

   dymoRoutingNextHopInterface OBJECT-TYPE
      SYNTAX       InterfaceIndexOrZero
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The interface ifIndex for sending
          packets toward the destination route
          address."
      ::= { dymoRoutingEntry 7 }

   dymoRoutingForwardingFlag OBJECT-TYPE
      SYNTAX       TruthValue
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The Forwarding Flag indicates whether
          this route can be used for forwarding
          data packets. A value 'true(1)'
          indicates that this route is being used
          for forwarding of data packets, while
          a value 'false(2)' indicates that it is
          not being used for forwarding."
      ::= { dymoRoutingEntry 8 }

   dymoRoutingBrokenFlag OBJECT-TYPE
      SYNTAX       TruthValue
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The Broken Flag indicates whether
          this Route is broken.  This flag is set
          if the next-hop becomes unreachable or
          in response to processing a RERR. A value
          'true(1)' indicates that this route is
          broken, while a value 'false(2)'
          indicates that it is not broken."
      ::= { dymoRoutingEntry 9 }

   dymoRoutingDist OBJECT-TYPE
      SYNTAX       Unsigned32 (0..65535)
      UNITS        "hops"
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The distance to the destination address's
          DYMO router. This is a metric of the
          distance a message or piece of information
          has traversed. The minimum value of distance
          is the number of IP hops traversed. The
          maximum value is 65,535.

          This parameter is an optional field in the
          DYMO routing table.  If the DYMO Route.Dist
          is not supported by this device, then this
          object should be set to '0'."
      REFERENCE
         "Dynamic MANET On-demand (DYMO) Routing,
          Chakeres, I., and C. Perkins, April
          2008. Section 3 Terminology."
      ::= { dymoRoutingEntry 10 }




   --
   -- DYMO Performance Group (Performance Management)
   --
   --    Contains objects which help to characterize the
   --    performance of the DYMO process, typically statistics
   --    counters. There are two types of DYMO statistics:
   --    global counters and per interface counters.
   --

   dymoPerformanceGroup  OBJECT IDENTIFIER ::= { dymoMIBObjects 3 }

   dymoGlobalPerfGroup  OBJECT IDENTIFIER ::= { dymoPerformanceGroup 1 }

   dymoRreqOriginated  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          device has initiated."
   ::= { dymoGlobalPerfGroup 1 }

   dymoRreqForwarded  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          device has forwarded, i.e., this
          device neither originated or
          terminated the RREQ message."
   ::= { dymoGlobalPerfGroup 2 }

   dymoRreqReceived  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          device has received as the
          target of the message."
   ::= { dymoGlobalPerfGroup 3 }

   dymoRrepOriginated  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          device has initiated."
   ::= { dymoGlobalPerfGroup 4 }

   dymoRrepForwarded  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          device has forwarded, i.e, this
          device neither originated or
          terminated the RREP message."
   ::= { dymoGlobalPerfGroup 5 }

   dymoRrepReceived   OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          device has received as the
          target of the message."
   ::= { dymoGlobalPerfGroup 6 }

   dymoRrerOriginated  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          device has initiated."
   ::= { dymoGlobalPerfGroup 7 }

   dymoRrerForwarded  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          device has forwarded, i.e., this
          device neither originated or
          terminated the RRER message."
   ::= { dymoGlobalPerfGroup 8 }

   dymoRrerReceived  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          device has received as the
          target of the message."
   ::= { dymoGlobalPerfGroup 9 }
   --
   -- Per DYMO Interface Performance Table
   --

   dymoInterfacePerfGroup OBJECT IDENTIFIER ::= {dymoPerformanceGroup 2}

   dymoInterfacePerfTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF DymoInterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The DYMO Interface Performance Table
          describes the DYMO statistics per
          interface."
   ::= { dymoInterfacePerfGroup 1 }

   dymoInterfacePerfEntry OBJECT-TYPE
      SYNTAX       DymoInterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The DYMO Interface Performance entry
          describes the statistics for a particular
          DYMO interface."
      INDEX { dymoIfPerfIfIndex }
   ::= { dymoInterfacePerfTable 1 }

   DymoInterfacePerfEntry ::=
      SEQUENCE {
         dymoIfPerfIfIndex
            InterfaceIndexOrZero,
         dymoIfRreqOriginated
            Counter32,
         dymoIfRreqForwarded
            Counter32,
         dymoIfRreqReceived
            Counter32,
         dymoIfRrepOriginated
            Counter32,
         dymoIfRrepForwarded
            Counter32,
         dymoIfRrepReceived
            Counter32,
         dymoIfRrerOriginated
            Counter32,
         dymoIfRrerForwarded
            Counter32,
         dymoIfRrerReceived
```

```
            Counter32
            }

   dymoIfPerfIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The ifIndex for this DYMO interface
          that is collecting this set of
          performance management statistics."
   ::= { dymoInterfacePerfEntry 1 }

   dymoIfRreqOriginated  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          interface has initiated."
   ::= { dymoInterfacePerfEntry 2 }

   dymoIfRreqForwarded  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          interface has forwarded, i.e., this
          interface neither originated nor
          terminated the RREQ message."
   ::= { dymoInterfacePerfEntry 3 }

   dymoIfRreqReceived  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREQ messages that this DYMO
          interface has received as the
          target of the message."
   ::= { dymoInterfacePerfEntry 4 }

   dymoIfRrepOriginated  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          interface has initiated."
   ::= { dymoInterfacePerfEntry 5 }

   dymoIfRrepForwarded  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          interface has forwarded, i.e., this
          interface neither originated nor
          terminated the RREP message."
   ::= { dymoInterfacePerfEntry 6 }

   dymoIfRrepReceived   OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RREP messages that this DYMO
          interface has received as the
          target of the message."
   ::= { dymoInterfacePerfEntry 7 }

   dymoIfRrerOriginated  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          interface has initiated."
   ::= { dymoInterfacePerfEntry 8 }

   dymoIfRrerForwarded  OBJECT-TYPE
      SYNTAX        Counter32
      MAX-ACCESS    read-only
      STATUS        current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          interface has forwarded, i.e., this
          interface neither originated nor
          terminated the RRER message."
   ::= { dymoInterfacePerfEntry 9 }

   dymoIfRrerReceived  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          RRER messages that this DYMO
          interface has received as the
          target of the message."
   ::= { dymoInterfacePerfEntry 10 }




   --
   -- Notifications
   --

   dymoMIBNotifControl OBJECT IDENTIFIER ::= { dymoMIBNotifications 1 }
   dymoMIBNotifObjects OBJECT IDENTIFIER ::= { dymoMIBNotifications 2 }

   -- dymoMIBNotifControl

   dymoSetNotification OBJECT-TYPE
          SYNTAX       OCTET STRING (SIZE(4))
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A 4-octet string serving as a bit map for
             the notification events defined by the DYMO
             notifications. This object is used to enable
             and disable specific DYMO notifications where
             a 1 in the bit field represents enabled. The
             right-most bit (least significant) represents
             notification 0.

             This object is persistent and when written
             the entity SHOULD save the change to
             non-volatile storage.
             "
           ::= { dymoMIBNotifControl 1 }
   -- dymoMIBNotifObjects

   dymoInstanceAdminStatusChange  NOTIFICATION-TYPE
      OBJECTS      { dymoInstanceAdminStatus,
                     dymoInstanceDid
                   }
      STATUS       current
      DESCRIPTION
         "This notification is generated when the
          administrative status of a DYMO process changes."
   ::= { dymoMIBNotifObjects 1 }

   dymoInterfaceAdminStatusChange  NOTIFICATION-TYPE
      OBJECTS      { dymoIfAdminStatus }
      STATUS       current
      DESCRIPTION
         "This notification is generated when the
          administrative status of a DYMO interface changes."
   ::= { dymoMIBNotifObjects 2 }

   dymoResponsibleAddrEntryChange  NOTIFICATION-TYPE
      OBJECTS      { dymoResponsibleAddrRowStatus }
      STATUS       current
      DESCRIPTION
         "This notification is generated when the status
          of an entry in the DYMO Responsible Address
          Table changes. This includes the creation or
          deletion of a row."
   ::= { dymoMIBNotifObjects 3 }

   --
   -- Compliance Statements
   --

   dymoCompliances  OBJECT IDENTIFIER ::= { dymoMIBConformance 1 }
   dymoMIBGroups    OBJECT IDENTIFIER ::= { dymoMIBConformance 2 }

   dymoBasicCompliance  MODULE-COMPLIANCE
      STATUS current
      DESCRIPTION "The basic implementation requirements for
                   managed network entities that implement
                   the DYMO routing protocol."
      MODULE  -- this module
      MANDATORY-GROUPS { dymoConfigObjectsGroup }
   ::= { dymoCompliances 1 }

   dymoFullCompliance MODULE-COMPLIANCE
      STATUS current
      DESCRIPTION "The full implementation requirements for managed
                   network entities that implement the DYMO routing
                   protocol."
      MODULE  -- this module
      MANDATORY-GROUPS { dymoConfigObjectsGroup,
                         dymoStateObjectsGroup,
                         dymoPerfObjectsGroup,
                         dymoNotifObjectsGroup,
                         dymoNotificationGroup }
   ::= { dymoCompliances 2 }

   --
   -- Units of Conformance
   --

   dymoConfigObjectsGroup OBJECT-GROUP
      OBJECTS {
              dymoInstanceAdminStatus,
              dymoInstanceDid,
              dymoInstanceRowStatus,
              dymoMaxHopLimit,
              dymoRouteTimeout,
              dymoRouteAgeMinTimeout,
              dymoRouteSeqnumAgeMaxTimeout,
              dymoRouteUsedTimeout,
              dymoRouteDeleteTimeout,
              dymoRouteRreqWaitTime,
              dymoDiscoveryAttemptsMax,
              dymoUnicastMsgSentTimeout,
              dymoIfAdminStatus,
              dymoIfRowStatus,
              dymoResponsibleAddrType,
              dymoResponsibleAddr,
              dymoResponsibleAddrPrefixLen,
              dymoResponsibleAddrRowStatus
      }
      STATUS  current
      DESCRIPTION
         "Set of DYMO configuration objects implemented
          in this module."
   ::= { dymoMIBGroups 1 }

   dymoStateObjectsGroup  OBJECT-GROUP
      OBJECTS {
              dymoCurrentSeqNum,
              dymoRoutingSeqNum,
              dymoRoutingNextHopIpAddrType,
              dymoRoutingNextHopIpAddress,
              dymoRoutingNextHopInterface,
              dymoRoutingForwardingFlag,
              dymoRoutingBrokenFlag,
              dymoRoutingDist
      }
      STATUS  current
      DESCRIPTION
         "Set of DYMO state objects implemented
          in this module."
   ::= { dymoMIBGroups 2 }

  dymoPerfObjectsGroup  OBJECT-GROUP
      OBJECTS {
              dymoRreqOriginated,
              dymoRreqForwarded,
              dymoRreqReceived,
              dymoRrepOriginated,
              dymoRrepForwarded,
              dymoRrepReceived,
              dymoRrerOriginated,
              dymoRrerForwarded,
              dymoRrerReceived,
              dymoIfRreqOriginated,
              dymoIfRreqForwarded,
              dymoIfRreqReceived,
              dymoIfRrepOriginated,
              dymoIfRrepForwarded,
              dymoIfRrepReceived,
              dymoIfRrerOriginated,
              dymoIfRrerForwarded,
              dymoIfRrerReceived
      }
      STATUS  current
      DESCRIPTION
         "Set of DYMO statistic objects implemented
          in this module for performance management."
   ::= { dymoMIBGroups 3 }

   dymoNotifObjectsGroup OBJECT-GROUP
      OBJECTS {
         dymoSetNotification
      }
      STATUS  current
      DESCRIPTION
         "Set of DYMO notifications objects implemented
          in this module."
   ::= { dymoMIBGroups 4 }
   dymoNotificationGroup NOTIFICATION-GROUP
      NOTIFICATIONS {
         dymoInstanceAdminStatusChange,
         dymoInterfaceAdminStatusChange,
         dymoResponsibleAddrEntryChange
      }
      STATUS  current
      DESCRIPTION
         "Set of DYMO notifications implemented in this
          module."
   ::= { dymoMIBGroups 5 }

   END
```

8. Security Considerations

[TODO] Each specification that defines one or more MIB modules MUST contain a section that discusses security considerations relevant to those modules. This section MUST be patterned after the latest approved template (available at http://www.ops.ietf.org/mib-security.html). Remember that the objective is not to blindly copy text from the template, but rather to think and evaluate the risks/vulnerabilities and then state/ document the result of this evaluation.

[TODO] if you have any read-write and/or read-create objects, please include the following boilerplate paragraph.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- [TODO] writable MIB objects that could be especially disruptive if abused MUST be explicitly listed by name and the associated security risks MUST be spelled out; RFC 2669 has a very good example.

- [TODO] list the writable tables and objects and state why they are sensitive.

[TODO] else if there are no read-write objects in your MIB module, use the following boilerplate paragraph.

There are no management objects defined in this MIB module that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB module is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB module via direct SNMP SET operations.

[TODO] if you have any sensitive readable objects, please include the following boilerplate paragraph.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- [TODO] you must explicitly list by name any readable objects that are sensitive or vulnerable and the associated security risks MUST be spelled out (for instance, if they might reveal customer information or violate personal privacy laws such as those of the European Union if exposed to unauthorized parties)

- [TODO] list the tables and objects and state why they are sensitive.

[TODO] discuss what security the protocol used to carry the information should have. The following three boilerplate paragraphs should not be changed without very good reason. Changes will almost certainly require justification during IESG review.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

9. IANA Considerations

[TODO] In order to comply with IESG policy as set forth in http://www.ietf.org/ID-Checklist.html, every Internet-Draft that is submitted to the IESG for publication MUST contain an IANA Considerations section. The requirements for this section vary depending what actions are required of the IANA. see RFC4181 section 3.5 for more information on writing an IANA clause for a MIB module document.

[TODO] select an option and provide the necessary details.

Option #1:

> The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
sampleMIB	{ mib-2 XXX }

Option #2:

Editor's Note (to be removed prior to publication): the IANA is requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

Note well: prior to official assignment by the IANA, a draft document MUST use place-holders (such as "XXX" above) rather than actual numbers. See RFC4181 Section 4.5 for an example of how this is done in a draft MIB module.

Option #3:

This memo includes no request to IANA.

10. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

11. Acknowledgements

12. References

12.1. Normative References

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

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

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[I-D.ietf-manet-dymo] Chakeres, I. and C. Perkins, "Dynamic MANET On-demand (DYMO) Routing", draft-ietf-manet-dymo-21 (work in progress), July 2010.

12.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

Appendix A. Change Log

This section identifies the changes that have been made from draft-ietf-manet-dymo-mib-00 .

These changes were made from draft-ietf-manet-dymo-mib-00 to draft-ietf-manet-dymo-mib-01.

1. Only minor changes of a typographic nature, e.g., read-only to read-write on MAX_ACCESS clauses of a few configuration objects.

These changes were made from draft-ietf-manet-dymo-mib-01 to draft-ietf-manet-dymo-mib-02.

1. Added the ForwardingFlag and BrokenFlag objects to the DYMO Routing Table.

2. Added the TruthValue Textual Convention to handle the new Routing Table objects.

3. Added the DYMO device management model to the introductory sections of this draft.

4. General clean up of the introductory sections of this draft.

These changes were made from draft-ietf-manet-dymo-mib-02 to draft-ietf-manet-dymo-mib-03.

1. Minor changes to the textual material and added to the IMPORTS text in the introductory material.

2. Added DEFVAL clauses to all read-write configuration objects having default values identified in the DYMO specification.

These changes were made from draft-ietf-manet-dymo-mib-03 to draft-ietf-manet-dymo-mib-04.

1. Incorporated the DID into the Configuration Group by changing the dymoAdminStatus object to an Instance Table. This allows for the presence of multiple DYMO processes concurrent on the same router.

2. Added the dymoNotifObjectsGroup and its dymoSetNotifications object to allow for individual control of the DYMO Notifications. Updated the Conformance sections accordingly.

3. Renamed several of the Configuration Objects to be consistent with the naming within the current draft-ietf-manet-dymo-21.

Appendix B. Open Issues

This section contains the set of open issues related to the development and design of the DYMO-MIB. This section will not be present in the final version of the MIB and will be removed once all the open issues have been resolved.

1. Work on the Security Section. This MIB does have settable objects, but not sensitive objects (true?).

2. Work on the relationship to other MIBs, IF-MIB, NHDP-MIB.

3. Cleanup all the [TODOs] from the MIB template.

Appendix C.

```
***************************************************************
* Note to the RFC Editor (to be removed prior to publication) *
*                                                             *
* 1) The reference to RFCXXXX within the DESCRIPTION clauses  *
* of the MIB module point to this draft and are to be         *
* assigned by the RFC Editor.                                 *
*                                                             *
* 2) The reference to RFCXXX2 throughout this document point  *
* to the current draft-ietf-manet-dymo-xx.txt.  This          *
* need to be replaced with the XXX RFC number.                *
*                                                             *
***************************************************************
```

Authors' Addresses

Sean Harnedy
Booz Allen Hamilton
333 City Boulevard West
Orange, California 92868
USA

Phone: +1 714 938-3898
EMail: harnedy_sean@bah.com

Robert G. Cole
US Army CERDEC
328 Hopkins Road, Bldg 245
Aberdeen Proving Ground, Maryland 21005
USA

Phone: +1 410 278 6779
EMail: robert.g.cole@us.army.mil
URI: http://www.cs.jhu.edu/~rgcole/

Ian D Chakeres
CenGen
9250 Bendix Road North
Columbia, Maryland 21045
USA

EMail: ian.chakeres@gmail.com
URI: http://www.ianchak.com/

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: March 9, 2012

U. Herberg
LIX, Ecole Polytechnique
R. Cole
US Army CERDEC
I. Chakeres
CenGen

September 6, 2011

Definition of Managed Objects for the Neighborhood Discovery Protocol
draft-ietf-manet-nhdp-mib-10

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring parameters of the Neighborhood Discovery Protocol (NHDP) process on a router. The MIB defined in this memo, denoted NHDP-MIB, also reports state, performance information and notifications. This additional state and performance information is useful to troubleshoot problems and performance issues during neighbor discovery.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring parameters of the Neighborhood Discovery Protocol [RFC6130] process on a router. The MIB defined in this memo, denoted NHDP-MIB, also reports state, performance information and notifications. This additional state and performance information is useful to troubleshoot problems and performance issues during neighbor discovery.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to Section 7 of [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in [RFC2578], [RFC2579] and [RFC2580].

3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and OPTIONAL" in this document are to be interpreted as described in [RFC2119].

4. Overview

[RFC6130] allows a router in a Mobile Ad Hoc Network (MANET) to discover and track topological information of routers up to two hops away by virtue of exchanging HELLO messages. This information is useful for routers running various routing and multicast flooding protocols developed within the IETF MANET Working Group.

4.1. Terms

The following definitions apply throughout this document:

- Notification Objects - triggers and associated notification messages allowing for asynchronous tracking of pre-defined events on the managed router.

- Configuration Objects - switches, tables, objects which are initialized to default settings or set through the management interface defined by this MIB.

- State Objects - automatically generated values which define the current operating state of the NHDP protocol process in the router.

- Performance Objects - automatically generated values which help an administrator or automated tool to assess the performance of the NHDP protocol process on the router and the overall discovery performance within the MANET.

5. Structure of the MIB Module

This section presents the structure of the NHDP-MIB module. The MIB is arranged into the following structure:

- nhdpNotifications - objects defining NHDP-MIB notifications.

- nhdpObjects - defining objects within this MIB. The objects are arranged into the following groups:

  * Configuration Group - defining objects related to the configuration of the NHDP instance on the router.

  * State Group - defining objects which reflect the current state of the NHDP instance running on the router.

  * Performance Group - defining objects which are useful to a management station when characterizing the performance of NHDP on the router and in the MANET.

- nhdpConformance - defining the minimal and maximal conformance requirements for implementations of this MIB.

5.1. Notifications

This section describes the use of notifications, and mechanisms to enhance the ability to manage NHDP networks.

5.1.1. Introduction

Notifications can be emitted by an NHDP router as a reaction to a specific event. This allows a network manager to efficiently determine the source of problems or significant changes of configuration or topology, instead of polling a possibly large number of NHDP routers.

5.1.2. Notification Generation

When an exception event occurs, the application notifies the local agent, which sends a notification to the appropriate SNMP management stations. The message includes the notification type and may include a list of notification-specific variables. Section 7 contains the notification definitions, which includes the variable lists. At least one IP address of the NHDP router that originates the notification is included in the variable list so that the network manager may determine the source of the notification.

5.1.3. Limiting Frequency of Notifications

To limit the frequency of notifications, the following additional mechanisms are suggested, similar to those in [RFC4750]:

5.1.3.1. Ignoring Initial Activity

The majority of critical events occur when NHDP is enabled on a router, at which time the symmetric neighbors and two-hop neighbors of the NHDP router are discovered. During this initial period, a potential flood of notifications is unnecessary since the events are expected. To avoid unnecessary notifications, a router should not originate expected notifications until a certain time interval has elapsed, which is to be predefined by the network manager.

5.1.3.2. Throttling Traps

The mechanism for throttling the notifications is the same as in [RFC4750] (i.e. the amount of transmitted notifications per time is bounded).

Appropriate values for the window time and upper bound are to be selected by the network manager and depend on the deployment of the MANET.

5.1.3.3. One Notification per Event

Similar to the according mechanism in [RFC4750], only one notification is sent per event.

5.2. The Configuration Group

The NHDP router is configured with a set of controls. The authoritative list of configuration controls within the NHDP-MIB are found within the MIB module itself. Generally, an attempt was made in developing the NHDP-MIB module to support all configuration objects defined in [RFC6130]. For all of the configuration parameters, the same constraints and default values of these parameters as defined in [RFC6130] are followed.

5.3. The State Group

The State Group reports current state information of a router running [RFC6130]. The NHDP-MIB State Group tables were designed to contain the complete set of state information defined within the information bases specified in Section 6, Section 7 and Section 8 of [RFC6130].

Two constructs, i.e., TEXTUAL CONVENTIONs, are defined to support of the tables in the State Group. The NHDP protocol stores and indexes information through sets of (dynamically defined) addresses, i.e., address sets. Within SMIv2 it is not possible to index tables with variably defined address sets. Hence, these TEXTUAL CONVENTIONS are defined to provide a local mapping between NHDP managed address sets and SMIv2 table indexing. These constructs are the NeighborIfIndex and NeighborRouterIndex. These are locally (to the NHDP router) defined, unique identifiers of virtual neighbors and neighbor interfaces. Due to the nature of the NHDP protocol, the local router may have identified distinct address sets but is not able to associate these as a single interface. Hence, two or more NeighborIfIndexes pointing to multiple distinct address sets may in fact be related to a common neighbor interface. This ambiguity may also hold with respect to the assignment of the NeighborRouterIndex. The local MIB agent is responsible for managing, aggregating and retiring the defined indexes, and in updating MIB tables using these indexes as the local router learns more about its neighbors' topology. These constructs are used to define indexes to the appropriate State Group tables and to correlate table entries to address sets, virtual neighbor interfaces and virtual neighbors within the MANET.

5.4. The Performance Group

The Performance Group reports values relevant to system performance. This section lists objects for NHDP performance monitoring, some of which are explicitly defined in the NHDP-MIB and others which can be estimated through a combination of base objects from this MIB. Throughout this section, those objects will be pointed out that are intended as base objects which are explicitly defined within this MIB and those objects which can be estimated.

Unstable neighbors or 2-hop neighbors and frequent changes of sets can have a negative influence on the performance of NHDP. The following objects allow management applications to acquire information related to the stability and performance of NHDP:

The following objects return statistics related to HELLO messages:

- Total number of sent HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageXmits

  Object type: Counter32

- Total number of received HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageRecvd

  Object type: Counter32

- Total number of sent periodic HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessagePeriodicXmits

  Object type: Counter32

- Total number of sent triggered HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageTriggeredXmits

  Object type: Counter32

- Acquire history of HELLO message scheduling instances for a given time duration on an interface

  It is desirable to develop the history of the exact timestamps of each HELLO message that has been sent as well as the type of the message (triggered or periodical). The list of events starts at the given point of time t0 and ends at the given time t1.

  This is a Derived Object estimated from the NHDP-MIB. It is derived from, e.g., the nhdpIfHelloMessagePeriodicXmits Base Object from the NHDP-MIB.

- Histogram of the intervals between HELLO messages on an interface

  It is desirable to track the values (in a 2-dimensional array) that represent a histogram of intervals between HELLO messages. The histogram would display the distribution of intervals between two consecutive HELLOs using a given bin size. It includes all HELLOs that have been sent after the given time t0 and before the given time t1.

  This is a Derived Object to be estimated from objects within the NHDP-MIB. It can be estimated from, e.g., the nhdpIfHelloMessagePeriodicXmits Base Object from the NHDP-MIB. The network management application could convert this information into the desired histogram.

- Changes of the frequency of the message scheduling on an interface

  This object will divide the given time interval from t0 to t1 into a given number of equal parts. It then creates a histogram for each part and calculates the distances (e.g. using the Bhattacharyya distance) between each two adjacent histograms in time. A higher value between two histograms means more difference between the histograms. For instance, this is representative of an event that suddenly sends many triggered HELLO messages, whereas before there have been only very few such triggered messages.

  This is a Derived Object estimated from objects within the NHDP-MIB, as previously discussed, albeit this is a bit more complex with respect to the management application.

- Average number of sent HELLO messages per second between the given time t0 and t1 on an interface

  This is a Derived Object estimated from, e.g., the nhdpIfHelloMessageXmits Base Object.

- Average number of received HELLO messages per second on an interface between the given time t0 and t1

  This is a Derived Object estimated from the NHDP-MIB. See the previous discussion.

- Total accumulated size in octets of sent HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageXmitAccumulatedSize

  Object type: Counter64

- Total accumulated size in octets of received HELLO messages on an interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageRecvdAccumulatedSize

  Object type: Counter64

- Average size in octets of sent HELLO messages between the given time t0 and t1 on an interface

  This is a Derived Object estimated from, e.g., the nhdpIfHelloMessageRecvdAccumulatedSize Base Object from this NHDP-MIB.

- Average size in octets of received HELLO messages between the given time t0 and t1 on an interface

  This is a Derived Object estimated from the NHDP-MIB. See previous discussion.

- Total accumulated number of advertised symmetric neighbors in HELLOs on that interface.

  This is a Base Object.

  Object name: nhdpIfHelloMessageXmitAccumulatedSymmetricNeighborCount

  Object type: Counter32

- Total accumulated number of advertised heard neighbors in HELLOs on that interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageXmitAccumulatedHeardNeighborCount

Object type: Counter32

- Total accumulated number of advertised lost neighbors in HELLOs on that interface

  This is a Base Object.

  Object name: nhdpIfHelloMessageXmitAccumulatedLostNeighborCount

  Object type: Counter32

- Number of expected packets from a given neighbor based on the packet sequence number on an interface

  This is a Base Object.

  Object name: nhdpDiscIfExpectedPackets

  Object type: Counter32

- Success rate of received packets (number of received packets divided by number of expected packets based on the packet sequence number)

  This is a Derived Object to be pulled from this NHDP-MIB. It is derived from, e.g., the nhdpDiscIfRecvdPackets and the nhdpDiscIfExpectedPackets Base Objects defined in this MIB. This metric is then computed by the network management application.

The following objects inspect the frequency of all Neighbor Set changes:

- Number of Neighbor Set changes

  This object counts each Neighbor Set change. A change occurs whenever a new Neighbor Tuple has been added, a Neighbor Tuple has been removed or any entry of a Neighbor Tuple has been modified.

  This is a Base Object.

  Object name: nhdpNibNeighborSetChanges

  Object type: Counter32

- Acquire history of Neighbor Set changes

  This object returns the history of the exact timestamps of each time the Neighbor Set has been changed.

  This is a Derived Object estimated from the NHDP-MIB. It is derived from the previously discussed Base Object.

- Histogram of the intervals between Neighbor Set changes

  Returns the values (in a 2-dimensional array) that represent a histogram of intervals between Neighbor Set changes.

  This is a Derived Object estimated from the previously discussed Base Object.

- Changes of the frequency of the Neighbor Set changes

  This object will divide the given time interval from t0 to t1 into a given number of equal parts. It then creates a histogram for each part and calculates the distances (e.g. using the Bhattacharyya distance) between each two adjacent histograms in time. A higher value between two histograms means more difference between the histograms.

  This is a Derived Object estimated from the previously discussed Base Object.

The next objects examine the uptime of a given neighbor (as listed in the Neighbor Set):

- Number of changes of a Neighbor Tuple

  Returns the number of changes to the given Neighbor Tuple.

  This is a Base Object.

  Object name: nhdpDiscNeighborNibNeighborSetChanges

  Object type: Counter32

- Neighbor uptime

  Returns the number of hundredths of a second since the Neighbor Tuple corresponding to the given neighbor exists.

This is a Base Object.

Object name: nhdpDiscNeighborNibNeighborSetUpTime

Object type: TimeTicks

- Acquire history of change of the 'nbrup' status of a given neighbor

  This object returns the history of the exact timestamps of each time the neighbor (as listed in the Neighbor Set) becomes 'nbrup' or 'nbrdown'. A neighbor is said to become 'nbrup' if a new Neighbor Tuple is created that corresponds to the given neighbor. It becomes 'nbrdown' if such a Neighbor Tuple has been deleted. The existence of a Lost Neighbor Tuple for that previous neighbor does not mean that the neighbor is still 'nbrup'.

  This is a Derived Object estimated from, e.g., the nhdpDiscNeighborNibNeighborSetChanges Base Object defined in this MIB.

- Histogram of the intervals between a change of the 'nbrup' status of a given neighbor

  Returns the values that represent a histogram of intervals between a change of the 'nbrup' status of a given neighbor. The histogram includes all changes that have been made after the given time t0 and before the given time t1.

  This is a Derived Object estimated from, e.g. the nhdpDiscNeighborNibNeighborSetChanges Base Object defined in this MIB. This object sits in the nhdpDiscNeighborSetPerfTable which is indexed by the nhdpDiscRouterIndex.

The following objects examine the stability of a neighbor. A neighbor is said to be unstable if it 'flaps' frequently between several links. It is said to be stable if the set of Link Tuples that correspond to the given neighbor is stationary.

- Count the changes of the interface(s) over which a given neighbor (as listed in the Neighbor Set) can be reached

  This object counts each time the neighbor changes the interface(s) over which it is reachable. A change in the set of Link Tuples corresponding to the appropriate Neighbor Tuple is registered, i.e. a corresponding Link Tuple is added or removed from the set of all corresponding Link Tuples.

This is a Base Object.

Object name: nhdpDiscNeighborNibNeighborSetReachableLinkChanges

Object type: Counter32

- Acquire history of changes of the interface(s) over which a given neighbor can be reached

  This object returns the history of the exact timestamps of each time the neighbor changes the interface(s) over which it is reachable. That means that there is a change in the set of corresponding Link Tuples of for that Neighbor Tuple, i.e. a corresponding Link Tuple is added or removed from the set of all corresponding Link Tuples.

  This is a Derived Object estimated from, e.g., the nhdpDiscNeighborNibNeighborSetReachableLinkChanges Base Object.

- Histogram of the intervals between a change of the interface(s) over which a given neighbor is reachable

  Returns the values that represent a histogram of intervals between a change of the interface over which a given neighbor is reachable after the given time t0 and before the given time t1.

  This is a Derived Object estimated from the previously discussed Base Object, nhdpDiscNeighborNibNeighborSetChanges counter.

The following objects inspect the stability of a given 2-hop neighbor:

- Count the changes of the union of all N2_neighbor_iface_addr_list of 2-hop Tuples with an N2_2hop_addr equal to one of the given 2-hop neighbor’s addresses

  This object returns the count of the times the 2-hop neighbor changes the neighbor(s) over which it is reachable.

  This is a Base Object.

  Object name: nhdpIib2HopSetPerfChanges

Object type: Counter32

- Acquire history of changes of the N2_neighbor_iface_addr_list of a given 2-hop neighbor

  This object returns the history of the exact timestamps of each time the 2-hop neighbor changes the neighbor(s) over which it is reachable.

  This is a Derived Object estimated from the previously discussed Base Object, nhdpIib2HopSetPerfChanges counter.

- Histogram of the intervals between a change of a 2-hop neighbor's N2_neighbor_iface_addr_list

  Returns the values that represent a histogram of intervals between a change of the neighbor(s) over which the 2-hop neighbor is reachable after the given time t0 and before the given time t1.

  This is a Derived Object estimated from the previously discussed Base Object, nhdpIib2HopSetPerfChanges counter.

The next objects examine the uptime of a given 2-hop neighbor:

- 2-hop Neighbor uptime

  Returns the number of hundredths of a second since the any 2-Hop Tuple with a N2_2hop_addr of the given 2-hop neighbor IP address was registered.

  This is a Base Object.

  Object name: nhdpIib2HopSetPerfUpTime

  Object type: TimeTicks

- Acquire history of change of 'nbrup' status of a given 2-hop neighbor

  This object returns the history of the exact timestamps of each time the 2-hop neighbor becomes 'nbrup' or 'nbrdown'. A 2-hop neighbor becomes 'nbrup' when the first 2-hop Tuple with N2_2hop_addr of the given 2-hop neighbor is created. It becomes 'nbrdown' when the last 2-hop Tuple with N2_2hop_addr of the given 2-hop neighbor has been deleted.

This is a Derived Object estimated from the previously discussed Base Object, nhdpIib2HopSetPerfChanges counter.

- Histogram of the intervals between a change of the ’nbrup’ status of a given 2-hop neighbor

  Returns the values that represent a histogram of intervals between a change of the ’nbrup’ status of a given 2-hop neighbor. The histogram includes all changes that have been made after the given time t0 and before the given time t1.

  This is a Derived Object estimated from the previously discussed Base Object, nhdpIib2HopSetPerfChanges counter.

6. Relationship to Other MIB Modules

This section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. Definitions imported from other MIB modules and other MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

6.1. Relationship to the SNMPv2-MIB

The ’system’ group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The ’system’ group provides identification of the management entity and certain other system-wide data. The NHDP-MIB does not duplicate those objects.

6.2. Relationship to Routing Protocol MIBs Relying on the NHDP-MIB

[RFC6130] allows routing protocols to rely on the neighborhood information that is discovered by means of HELLO message exchange. In order to allow for troubleshooting, fault isolation, and management of such routing protocols through a routing protocol MIB, it may be desired to align the State Group tables of the NHDP-MIB and the routing protocol MIB. This is accomplished through the definition of two TEXTUAL-CONVENTIONS in the NHDP-MIB: the NeighborIfIndex and the NeighborRouterIndex. These object types are used to develop indexes into common NHDP-MIB and routing protocol State Group tables. These objects are locally significant but should be locally common to the NHDP-MIB and the routing protocol MIB implemented on a common networked router. This will allow for improved cross referencing of information across the two MIBs.

6.3. MIB Modules Required for IMPORTS

The following NHDP-MIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF [RFC2580], IF-MIB [RFC2863], INET-ADDRESS-MIB [RFC4001], and FLOAT-TC-MIB [RFC6340].

7. Definitions

This section contains the MIB module defined by the specification.

```
NHDP-MIB DEFINITIONS ::= BEGIN

-- This MIB module defines objects for the management of the
-- NHDP [RFC6130] - The Neighborhood Discovery Protocol,
-- Clausen, T., Dearlove, C. and J. Dean, January 2011.

IMPORTS

    MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
    Counter32, Counter64, Integer32, Unsigned32, mib-2,
                 TimeTicks
                 FROM SNMPv2-SMI  --[RFC2578]

    TEXTUAL-CONVENTION, TruthValue, TimeStamp,
                 RowStatus
                 FROM SNMPv2-TC  --[RFC2579]

    MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
                 FROM SNMPv2-CONF  --[STD58]

    InetAddressType, InetAddress,
    InetAddressPrefixLength
                 FROM INET-ADDRESS-MIB  --[RFC4001]

    InterfaceIndexOrZero
                 FROM IF-MIB  --[RFC2863]

    Float32TC
                 FROM FLOAT-TC-MIB  --[RFC6340]
    ;

nhdpMIB MODULE-IDENTITY
      LAST-UPDATED "201109061000Z" -- September 6, 2011
      ORGANIZATION "IETF MANET working group"
      CONTACT-INFO
      "WG E-Mail: manet@ietf.org

       WG Chairs: ian.chakeres@gmail.com
```

```
                   jmacker@nrl.navy.mil


        Editors:   Ulrich Herberg
                   Ecole Polytechnique
                   LIX
                   91128 Palaiseau Cedex
                   France
                   ulrich@herberg.name
                   http://www.herberg.name/

                   Robert G. Cole
                   US Army CERDEC
                   Space and Terrestrial Communications
                   328 Hopkins Road
                   Bldg 245, Room 16
                   Aberdeen Proving Ground, MD 21005
                   USA
                   +1 410 278-6779
                   robert.g.cole@us.army.mil
                   http://www.cs.jhu.edu/~rgcole/

                   Ian D Chakeres
                   CenGen
                   9250 Bendix Road North
                   Columbia, Maryland  21045
                   USA
                   ian.chakeres@gmail.com
                   http://www.ianchak.com/"

       DESCRIPTION
           "This NHDP-MIB module is applicable to routers
            implementing the Neighborhood Discovery Protocol
            defined in [RFC6130].

            Copyright (C) The IETF Trust (2009). This version
            of this MIB module is part of RFCXXXX; see the RFC
            itself for full legal notices."

       -- revision
       REVISION "201109061000Z" -- September 6, 2011
       DESCRIPTION
            "The thirteenth version of this MIB module,
             published as draft-ietf-manet-nhdp-mib-10.txt.
             Removed references to the REPORT-MIB.
             Added references to RFC 6340 for the
             Float32TC Textual Convention.
            "
       REVISION "201107281000Z" -- July 28, 2011
       DESCRIPTION
            "The twelth version of this MIB module,
             published as draft-ietf-manet-nhdp-mib-09.txt.
            "
       REVISION "201107081000Z" -- July 8, 2011
       DESCRIPTION
            "The eleventh version of this MIB module,
             published as draft-ietf-manet-nhdp-mib-08.txt.
             Clarified the use of the NeighborIfIndex and
             the NeighborRouterIndex.  Also, cleaned up
             the indexing of tables in the StateObjGroup.
            "
       REVISION "201101031000Z" -- January 3, 2011
       DESCRIPTION
            "The tenth version of this MIB module,
             published as draft-ietf-manet-nhdp-mib-07.txt.
             Added Float32TC from FLOAT-TC-MIB using this
             for representing the link quality parameters.
             Added a threshold (number) and window (time
             interval) within the nhdpNotificationsControl
             for the nhdpNbrStateChange, nhdp2HopNbrStateChange
             and nhdpIfRxBadPacket notifications.
            "
       REVISION "201011111000Z" -- November 11, 2010
       DESCRIPTION
            "The ninth version of this MIB module,
            published as draft-ietf-manet-nhdp-mib-06.txt.
            Corrected editorial issues, fixed some small
            bugs in the MIB."
       REVISION "201011081000Z" -- November 08, 2010
       DESCRIPTION
            "The eight version of this MIB module,
            published as draft-ietf-manet-nhdp-mib-05.txt.
            Cleaned up defaults and interdependence's
            between objects."
       REVISION    "201007071000Z"   -- July 07, 2010
       DESCRIPTION
         "The seventh version of this MIB module,
          published as draft-ietf-manet-nhdp-mib-04.txt.
          Cleaned up and condensed the textual material
          in the earlier sections of this draft.  Checked
          consistency with NHDP draft, i.e.,
          draft-ietf-manet-nhdp-12.txt."
       REVISION    "201003081000Z"   -- March 08, 2010
       DESCRIPTION
         "The sixth version of this MIB module,
          published as draft-ietf-manet-nhdp-mib-03.txt.
          Added the local nhdpIfIndex to the
          nhdpIibLinkSetTable."
       REVISION    "200911091000Z"   -- November 09, 2009
       DESCRIPTION
         "The fifth version of this MIB module,
          published as draft-ietf-manet-nhdp-mib-02.txt.
          Cleaned up a few things and updated to newest
          revision of NHDP draft."
       REVISION    "200910211000Z"   -- October 21, 2009
       DESCRIPTION
         "The fourth version of this MIB module,
          published as draft-ietf-manet-nhdp-mib-01.txt.
          Added objects pertaining to the performance
          group."
       REVISION    "200905031500Z"   -- May 3, 2009
       DESCRIPTION
         "The third version of this MIB module,
          published as draft-ietf-manet-nhdp-mib-00.txt.
          No major revisions to this draft.  Mainly rev'd
          as a new working group document.  But also cleaned
          syntax errors, typos and other issues discovered
          with 'smilint'."
       REVISION    "200902151500Z"   -- February 15, 2009
       DESCRIPTION
         "The second version of this MIB module,
          published as draft-cole-manet-nhdp-mib-01.txt.  Major
          update adding objects for configuration and state."
       REVISION    "200804251500Z"   -- April 25, 2008
       DESCRIPTION
         "The original version of this MIB module,
          published as draft-cole-manet-nhdp-mib-00.txt."
       -- RFC-Editor assigns XXXX
       ::= { mib-2 998 }      -- to be assigned by IANA

--
-- Top-Level Components of this MIB
--
nhdpNotifications OBJECT IDENTIFIER ::= { nhdpMIB 0 }
nhdpObjects       OBJECT IDENTIFIER ::= { nhdpMIB 1 }
nhdpConformance   OBJECT IDENTIFIER ::= { nhdpMIB 2 }


--
-- Textual Conventions
--

NeighborIfIndex ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS       current
    DESCRIPTION
        "An arbitrary, locally unique identifier associated with a
        virtual interface of a discovered NHDP neighbor.
        Due to the nature of the NHDP protocol, the local router
        may not know if two distinct addresses belong to the
        same interface of a neighbor or to two different
        interfaces.  As the local router gains more
        knowledge of its neighbors, its local view may change and
        this table will be updated to reflect the local router's current
        understanding associating address sets to neighbor interfaces.
        The local router identifies virtual neighbor interface through
        the receipt of address lists advertised through an
        NHDP HELLO message.

        All objects of type NeighborIfIndex are assigned by the agent
        out of a common number space.

        The value for each discovered virtual neighbor
        interface must remain constant at least from
        one re-initialization of the entity's network management
        agent to the next re-initialization, except that
        if an application is deleted and re-created.  If the
        local router gains information associating two virtual
        interfaces on a neighbor as a common interface,
        then the agent must aggregate the two address sets to
        a single index chosen from the set of aggregated indexes,
        it must update all tables in this
        MIB which are indexed by indexes
        of type NeighborIfIndex.  It can then reuse freed
        index values following the next agent restart.

        The specific value is meaningful only within a given SNMP
        entity."
    SYNTAX       Unsigned32 (1..2147483647)


NeighborRouterIndex ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS       current
    DESCRIPTION
        "An arbitrary, locally unique identifier associated with a
        virtual discovered neighbor (one or two hop).  Due to the
        nature of the NHDP protocol, the local router may identify
        multiple virtual neighbors  which in fact are one and
        the same.  Two hop neighbors with more than one advertised
        address will exhibit this behavior.  As the
        local router's knowledge of its neighbors' topology
        increases, the local router will be able to associate
        multiple virtual neighbor indexes into a single virtual
        neighbor index chosen from the set of aggregated indexes,
        it must update all tables in this MIB indexed by these
        indexes, and it can reuse the freed indexes following the
        next agent re-initialization.

        All objects of type NeighborRouterIndex are assigned by
        the agent out of a common number space.

        The NeighborRouterIndex defines a discovered NHDP peer
        virtual neighbor of the local router.
        The value for each discovered virtual neighbor index must remain
        constant at least from one re-initialization of the entity's
        network management agent to the next re-initialization, except
        that if an application is deleted and re-created.

        The specific value is meaningful only within a given SNMP
        entity. An NeighborRouterIndex value must not be re-used
        until the next agent restart."
    SYNTAX       Unsigned32 (1..2147483647)



--
-- nhdpObjects
--

--    1) Configuration Objects Group
--    2) State Objects Group
--    3) Performance Objects Group



--
-- nhdpConfigurationObjGrp
--

-- Contains the NHDP objects which configure specific options
-- which determine the overall performance and operation of the
-- discovery protocol.


nhdpConfigurationObjGrp OBJECT IDENTIFIER ::= { nhdpObjects 1 }


   nhdpInterfaceTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF NhdpInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "nhdpInterfaceTable describes the
          configuration of the interfaces of this NHDP router.
          The ifIndex is from the interfaces group
          defined in the Interfaces Group MIB.

          The objects in this table are persistent and when
          written the entity SHOULD save the change to
          non-volatile storage."
      REFERENCE
         "[RFC2863] - The Interfaces Group MIB, McCloghrie,
          K., and F. Kastenholtz, June 2000."
   ::= { nhdpConfigurationObjGrp 1 }

   nhdpInterfaceEntry OBJECT-TYPE
      SYNTAX      NhdpInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "nhdpInterfaceEntry describes one NHDP
          local interface configuration as indexed by
          its ifIndex as defined in the Standard MIB II
          Interface Table [RFC2863]."
      INDEX { nhdpIfIndex }
   ::= { nhdpInterfaceTable 1 }

   NhdpInterfaceEntry ::=
      SEQUENCE {
         nhdpIfIndex
            InterfaceIndexOrZero,
         nhdpIfStatus
            TruthValue,
         nhdpHelloInterval
            Unsigned32,
         nhdpHelloMinInterval
            Unsigned32,
         nhdpRefreshInterval
            Unsigned32,
         nhdpLHoldTime
            Unsigned32,
         nhdpHHoldTime
            Unsigned32,
         nhdpHystAcceptQuality
            Float32TC,
         nhdpHystRejectQuality
            Float32TC,
         nhdpInitialQuality
            Float32TC,
         nhdpInitialPending
            TruthValue,
         nhdpHpMaxJitter
            Unsigned32,
         nhdpHtMaxJitter
            Unsigned32,
         nhdpIfRowStatus
            RowStatus
         }

   nhdpIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The ifIndex for this interface."
      ::= { nhdpInterfaceEntry 1 }

   nhdpIfStatus  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhpdIfStatus indicates whether this interface is
          a MANET interface. A value of true(1) indicates
          that the interface is a MANET interface.  A value of
          false(2) indicates that the interface is not a MANET
          interface. This corresponds to the I_manet parameter
          in the Local Interface Set.
         "
      DEFVAL { 2 }
   ::= { nhdpInterfaceEntry 2 }


   --
   -- Interface Parameters - Message Intervals
   --

   nhdpHelloInterval  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhpdHelloInterval corresponds to
          HELLO_INTERVAL of NHDP.
          The following constraint applies to this
          parameter:
             nhpdHelloInterval >= nhdpHelloMinInterval"
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 2000 }
   ::= { nhdpInterfaceEntry 3 }


   nhdpHelloMinInterval  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhpdHelloMinInterval corresponds to
         HELLO_MIN_INTERVAL of NHDP."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 500 }
   ::= { nhdpInterfaceEntry 4 }


   nhdpRefreshInterval  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhpdRefreshInterval corresponds to
         REFRESH_INTERVAL of NHDP.

         The following constraint applies to this
         parameter:
             nhdpRefreshInterval >= nhdpHelloInterval"
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 2000 }
   ::= { nhdpInterfaceEntry 5 }

   --
   -- Interface Parameters - Information Validity times
   --

   nhdpLHoldTime  OBJECT-TYPE
```

```
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpLHoldTime corresponds to
         L_HOLD_TIME of NHDP."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 6000 }
   ::= { nhdpInterfaceEntry 6 }

   nhdpHHoldTime  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpHHoldTime corresponds to
         H_HOLD_TIME of NHDP.

         This object is persistent and when written
         the entity SHOULD save the change to
         non-volatile storage."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 6000 }
   ::= { nhdpInterfaceEntry 7 }

   --
   -- Interface Parameters - Link Quality
   -- (is optional and settings define operation)
   --

   nhdpHystAcceptQuality  OBJECT-TYPE
      SYNTAX      Float32TC
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpHystAcceptQuality corresponds to
         HYST_ACCEPT of NHDP.

          The following constraint applies to this
          parameter:
              0 <= nhdpHystRejectQuality
                <= nhdpHystAcceptQuality <= 1.0"
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      -- DEFVAL { 1.0 }
   ::= { nhdpInterfaceEntry 8 }

   nhdpHystRejectQuality  OBJECT-TYPE
      SYNTAX      Float32TC
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpHystRejectQuality corresponds to
         HYST_REJECT of NHDP.

          The following constraint applies to this
          parameter:
              0 <= nhdpHystRejectQuality
                <= nhdpHystAcceptQuality <= 1.0"
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      -- DEFVAL { 0.0 }
   ::= { nhdpInterfaceEntry 9 }

   nhdpInitialQuality  OBJECT-TYPE
      SYNTAX      Float32TC
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpInitialQuality corresponds to
         INITIAL_QUALITY of NHDP.

          The following constraint applies to this
          parameter:
              0 <= nhdpInitialQuality <= 1.0"
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      -- DEFVAL { 1.0 }
   ::= { nhdpInterfaceEntry 10 }



   nhdpInitialPending  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
```

```
         "nhdpInitialPending corresponds to
         INITIAL_PENDING of NHDP.

         The following constraints apply to this parameter:

         If INITIAL_QUALITY >= HYST_ACCEPT,
         then INITIAL_PENDING := false.

         If INITIAL_QUALITY < HYST_REJECT,
         then INITIAL_PENDING := true."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 2 }   -- i.e. false
   ::= { nhdpInterfaceEntry 11 }

   --
   -- Interface Parameters - Jitter
   --
   nhdpHpMaxJitter  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpHpMaxJitter corresponds to
         HP_MAXJITTER of NHDP.

         For constraints on this object, refer
         to Section 5.4 of [RFC5148]."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 500 }
   ::= { nhdpInterfaceEntry 12 }


   nhdpHtMaxJitter  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpHtMaxJitter corresponds to
         HT_MAXJITTER of NHDP."
      REFERENCE
         "Section 5 on Protocol Parameters and
          Constraints of [RFC6130]."
      DEFVAL { 500 }
   ::= { nhdpInterfaceEntry 13 }

   nhdpIfRowStatus  OBJECT-TYPE
      SYNTAX      RowStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "This object permits management of the table
         by facilitating actions such as row creation,
         construction, and destruction. The value of
         this object has no effect on whether other
         objects in this conceptual row can be
         modified."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpInterfaceEntry 14 }


   --
   -- Router Parameters - Information Validity Time
   --
   nhdpNHoldTime  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "nhdpNHoldTime corresponds to
         N_HOLD_TIME of NHDP.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage."
      REFERENCE
         "[RFC6130].
          Section 5 on Protocol Parameters and
          Constraints."
      DEFVAL { 6000 }
   ::= { nhdpConfigurationObjGrp 2 }


   nhdpIHoldTime  OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "milliseconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
```

```
          "nhdpIHoldTime corresponds to
          I_HOLD_TIME of NHDP.

           This object is persistent and when written
           the entity SHOULD save the change to
           non-volatile storage."
       REFERENCE
          "[RFC6130].
           Section 5 on Protocol Parameters and
           Constraints."
       DEFVAL { 6000 }
    ::= { nhdpConfigurationObjGrp 3 }


     -- An NHDP router's Local Information Base (LIB)

     -- Local Interface Set Table
     -- Entry (foreach local interface): (IfNetAddrs, Is_manet)

     nhdpLibLocalIfSetTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF NhdpLibLocalIfSetEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
           "A router's Local Interface Set records all
            network addresses which are defined as local
            interface network addresses.  The local interface
            is defined by the nhdpIfIndex.

            It consists of Local Interface Address Tuples
            per network interface and their prefix lengths (in
            order to determine the network addresses related to
            the interface) and an indication of whether the
            interface is a MANET interface or not.

            Further guidance on the addition or removal of
            local addresses and network addresses is found
            in Section 9 of [RFC6130]."
        REFERENCE
           "[RFC6130]."
     ::= { nhdpConfigurationObjGrp 4 }

     nhdpLibLocalIfSetEntry  OBJECT-TYPE
        SYNTAX      NhdpLibLocalIfSetEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
           "A router's Local Interface Set consists
            of Configured Interface Address Tuples foreach network
            interface, and an indication of whether the interface
            is a MANET interface or not.

            (IR_local_iface_addr, IR_time)
           "
        REFERENCE
           "[RFC6130]."
        INDEX { nhdpIfIndex }
     ::= { nhdpLibLocalIfSetTable 1 }

     NhdpLibLocalIfSetEntry ::=
        SEQUENCE {
           nhdpLibLocalIfSetIpAddrType
             InetAddressType,
           nhdpLibLocalIfSetIpAddr
             InetAddress,
           nhdpLibLocalIfSetIpAddrPrefixLen
             InetAddressPrefixLength,
           nhdpLibLocalIfSetIsManet
             TruthValue
          }

    nhdpLibLocalIfSetIpAddrType  OBJECT-TYPE
       SYNTAX      InetAddressType
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
          "The type of the nhdpLibLocalIfSetIpAddr
           in the InetAddress MIB [RFC4001]."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpLibLocalIfSetEntry 1 }

    nhdpLibLocalIfSetIpAddr  OBJECT-TYPE
       SYNTAX      InetAddress
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
          "nhdpLibLocalIfSetAddr is an
           address of an interface of
           this router."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpLibLocalIfSetEntry 2 }

    nhdpLibLocalIfSetIpAddrPrefixLen  OBJECT-TYPE
       SYNTAX      InetAddressPrefixLength
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
          "Indicates the number of leading one bits that
           form the mask.  The mask is logically-ANDed
           to the nhdpLibLocalIfSetIpAddr to determine
           the address prefix.  A row match is true
           if the address used as an index falls within
           the network address range defined by the
           address prefix."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpLibLocalIfSetEntry 3 }

      nhdpLibLocalIfSetIsManet  OBJECT-TYPE
         SYNTAX      TruthValue
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
            "Specifies whether this interface is
             a MANET interface or not."
         REFERENCE
            "[RFC6130]."
      ::= { nhdpLibLocalIfSetEntry 4 }



     -- Removed Interface Addr Set Table
     -- Entry (foreach removed network addr): (IfAddrRemoved,
     --                                        ExpirationTime)

     nhdpLibRemovedIfAddrSetTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF NhdpLibRemovedIfAddrSetEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
           "A router's Removed Interface Address Set records
           network addresses which were recently used as local
           interface network addresses.  If a router's interface
           network addresses are immutable then the Removed
           Interface Address Set is always empty and MAY be omitted.
           It consists of Removed Interface Address Tuples, one
           per network address."
        REFERENCE
           "[RFC6130]."
     ::= { nhdpConfigurationObjGrp 5 }

     nhdpLibRemovedIfAddrSetEntry  OBJECT-TYPE
```

```
       SYNTAX      NhdpLibRemovedIfAddrSetEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "A router's Removed Interface Address Set consists
          of Removed Interface Address Tuples, one per network
          address:

          (IR_local_iface_addr, IR_time)

          The association between these addrs and
          the router's Interface is found in the
          Standard MIB II's IP address table
          (RFC1213)."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpLibRemovedIfAddrSetIpAddrType,
               nhdpLibRemovedIfAddrSetIpAddr }
    ::= { nhdpLibRemovedIfAddrSetTable 1 }

    NhdpLibRemovedIfAddrSetEntry ::=
       SEQUENCE {
          nhdpLibRemovedIfAddrSetIpAddrType
            InetAddressType,
          nhdpLibRemovedIfAddrSetIpAddr
            InetAddress,
          nhdpLibRemovedIfAddrSetIpAddrPrefixLen
            InetAddressPrefixLength,
          nhdpLibRemovedIfAddrSetIfIndex
            InterfaceIndexOrZero,
          nhdpLibRemovedIfAddrSetIrTime
            TimeStamp
         }

   nhdpLibRemovedIfAddrSetIpAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The type of the nhdpLibRemovedIfAddrSetIpAddr
          in the InetAddress MIB [RFC4001]."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpLibRemovedIfAddrSetEntry 1 }

   nhdpLibRemovedIfAddrSetIpAddr  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpLibRemovedIfAddrSetAddr is a
          recently used address of an interface of
          this router."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpLibRemovedIfAddrSetEntry 2 }

   nhdpLibRemovedIfAddrSetIpAddrPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "Indicates the number of leading one bits that
          form the mask.  The mask is logically-ANDed
          to the nhdpLibRemovedIfAddrSetIpAddr to determine
          the address prefix.  A row match is true
          if the address used as an index falls within
          the network address range defined by the
          address prefix."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpLibRemovedIfAddrSetEntry 3 }


     nhdpLibRemovedIfAddrSetIfIndex  OBJECT-TYPE
        SYNTAX      InterfaceIndexOrZero
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
           "Specifies the local IfIndex from which this
            IP address was recently removed."
        REFERENCE
           "[RFC6130]."
     ::= { nhdpLibRemovedIfAddrSetEntry 4 }

     nhdpLibRemovedIfAddrSetIrTime  OBJECT-TYPE
        SYNTAX      TimeStamp
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
           "Specifies when this Tuple expires and MUST be removed
            from this table."
        REFERENCE
           "[RFC6130]."
     ::= { nhdpLibRemovedIfAddrSetEntry 5 }
```

```
--
-- nhdpStateObjGrp
--

-- Contains information describing the current state of the NHDP
-- process on this device.

nhdpStateObjGrp    OBJECT IDENTIFIER ::= { nhdpObjects 2 }

   -- Two new constructs have been defined in this MIB for
   -- indexing into the following
   -- tables and indexing into other tables in other MIBs.
   -- This was necessary because the NHDP protocol manages and
   -- indexes based upon dynamic address tuples, i.e.,
   -- address sets, while SMI requires statically
   -- defined indexes for accessing its table rows.
   -- The NeighborIfIndex defines a unique (to the local router)
   -- index referencing a discovered virtual interface on another
   -- neighbor within the MANET. The NeighborRouterIndex defines a
   -- unique (to the local router) index referencing a discovered
   -- virtual neighbor within the MANET.
   --
   -- Due to the nature of the NHDP protocol,
   -- different indexes may be related to common neighbor
   -- interfaces or common neighbor routers, but the information
   -- obtained through NHDP has not allowed the local router
   -- to relate these virtual objects (i.e., interfaces or routers)
   -- at this point in time.  As more topology information
   -- is gathered by the local router, it may associate
   -- virtual interfaces or routers and collapse these
   -- indexes appropriately.

   -- Multiple addresses can be associated with a
   -- given NeighborIfIndex.  Each NeighborIfIndex is
   -- associated with a NeighborRouterIndex.  Throughout
   -- the nhdpStateObjGroup, the
   -- NeighborIfIndex and the NeighborRouterIndex are used
   -- to define the set of IpAddrs related to a virtual
   -- neighbor interface or virtual neighbor under discussion.

   nhdpUpTime OBJECT-TYPE
       SYNTAX TimeTicks
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
             "The number of hundredths of a second since the
             current NHDP process was initialized."
        REFERENCE
```

```
          "[RFC6130]."
    ::= { nhdpStateObjGrp 1 }


   nhdpInterfaceStateTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF NhdpInterfaceStateEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "nhdpInterfaceStateTable lists state information
          related to specific interfaces of this NHDP router.
          The ifIndex is from the interfaces group
          defined in the Interfaces Group MIB.

          The objects in this table are persistent and when
          written the entity SHOULD save the change to
          non-volatile storage."
      REFERENCE
         "RFC 2863 - The Interfaces Group MIB, McCloghrie,
          K., and F. Kastenholtz, June 2000."
   ::= { nhdpStateObjGrp 2 }

   nhdpInterfaceStateEntry OBJECT-TYPE
      SYNTAX      NhdpInterfaceStateEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "nhdpInterfaceStateEntry describes one NHDP
          local interface state as indexed by
          its ifIndex as defined in the Standard MIB II
          Interface Table (RFC2863)."
      INDEX { nhdpIfIndex }
   ::= { nhdpInterfaceStateTable 1 }

   NhdpInterfaceStateEntry ::=
      SEQUENCE {
         nhdpIfStateUpTime
            TimeTicks
         }

   nhdpIfStateUpTime  OBJECT-TYPE
      SYNTAX      TimeTicks
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
             "The number of hundredths of a second since the
             current NHDP process was initialized."
   ::= { nhdpInterfaceStateEntry 1 }
   --
   -- Interface Parameters - Message Intervals
   --

   nhdpDiscIfSetTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF NhdpDiscIfSetEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
           "A router's set of discovered interfaces on
            neighboring routers."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpStateObjGrp 3 }


    nhdpDiscIfSetEntry  OBJECT-TYPE
       SYNTAX      NhdpDiscIfSetEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "The entries include the nhdpDiscRouterIndex of
           the discovered router, the nhdpDiscIfIndex
           of the discovered interface and the
           current set of addresses associated
           with this neighbor interface.  The
           nhdpDiscIfIndex uniquely identifies
           the remote interface address sets
           through this table.  It does not need
           to be unique across the MANET, but must
           be locally unique within this router."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpDiscIfSetIndex }
    ::= { nhdpDiscIfSetTable 1 }

    NhdpDiscIfSetEntry ::=
       SEQUENCE {
          nhdpDiscIfSetIndex
            Integer32,
          nhdpDiscIfIndex
            NeighborIfIndex,
          nhdpDiscRouterIndex
            NeighborRouterIndex,
          nhdpDiscIfSetIpAddrType
            InetAddressType,
          nhdpDiscIfSetIpAddr
            InetAddress,
          nhdpDiscIfSetIpAddrPrefixLen
            InetAddressPrefixLength
         }

   nhdpDiscIfSetIndex  OBJECT-TYPE
      SYNTAX      Integer32 (0..65535)
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The index for this table.  Necessary
          because multiple addresses may be associated
          with a given nhdpDiscIfIndex."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 1 }

   nhdpDiscIfIndex  OBJECT-TYPE
      SYNTAX      NeighborIfIndex
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The NHDP interface index (locally created)
          of a neighbor’s interface.  Used for cross
          indexing into other NHDP tables and other
          MIBs."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 2 }

   nhdpDiscRouterIndex  OBJECT-TYPE
      SYNTAX      NeighborRouterIndex
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The NHDP neighbor index (locally created)
          of a neighboring router.  Used for cross
          indexing into other NHDP tables and other
          MIBs."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 3 }

   nhdpDiscIfSetIpAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The type of the nhdpDiscIfSetIpAddr
```

```
          in the InetAddress MIB [RFC4001]."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 4 }

   nhdpDiscIfSetIpAddr  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The nhdpDiscIfSetIpAddr is a
          recently used address of a neighbor
          of this router."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 5 }

   nhdpDiscIfSetIpAddrPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "Indicates the number of leading one bits that
          form the mask.  The mask is logically-ANDed
          to the nhdpDiscIfSetIpAddr to determine
          the address prefix.  A row match is true
          if the address used as an index falls within
          the network address range defined by the
          address prefix."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetEntry 6 }


   -- Interface Information Base (IIB)

   --
   -- NHDP Interface Information Base (IIB)
   --

   --     IIB Link Set
   --         Entry (foreach discovered link to a
   --                        1-H neighbor): (NeighborIfAddrList,
   --                                        HeardTime,
   --                                        SymTime,
   --                                        Quality,
   --                                        Pending,
   --                                        Lost,
   --                                        ExpireTime)

   nhdpIibLinkSetTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF NhdpIibLinkSetEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
          "A Link Set of an interface records all links
           from other routers which are, or recently
           were, 1-hop neighbors."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpStateObjGrp 4 }

    nhdpIibLinkSetEntry  OBJECT-TYPE
       SYNTAX      NhdpIibLinkSetEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "A Link Set consists of Link Tuples, each
           representing a single link indexed by the
           local and remote interface pair:

           (L_neighbor_iface_addr_list, L_HEARD_time,
            L_SYM_time, L_quality, L_pending,
            L_lost, L_time).

            Note that L_quality is not included in the
            entries below, because updates may be
            required too frequently."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpIfIndex,
               nhdpDiscIfIndex }
    ::= { nhdpIibLinkSetTable 1 }

    NhdpIibLinkSetEntry ::=
       SEQUENCE {
          nhdpIibLinkSetLHeardTime
            TimeStamp,
          nhdpIibLinkSetLSymTime
            TimeStamp,
          nhdpIibLinkSetLPending
            TruthValue,
          nhdpIibLinkSetLLost
            TruthValue,
          nhdpIibLinkSetLTime
            TimeStamp
         }

   nhdpIibLinkSetLHeardTime  OBJECT-TYPE
      SYNTAX      TimeStamp
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIibLinkSetLHeardTime corresponds
         to L_HEARD_time of NHDP."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIibLinkSetEntry 1 }

   nhdpIibLinkSetLSymTime  OBJECT-TYPE
      SYNTAX      TimeStamp
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIibLinkSetLSymTime corresponds
         to L_SYM_time of NHDP."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIibLinkSetEntry 2 }

   nhdpIibLinkSetLPending  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIibLinkSetLPending corresponds
         to L_pending of NHDP"
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIibLinkSetEntry 3 }

   nhdpIibLinkSetLLost  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIibLinkSetLLost corresponds
         to L_lost of NHDP"
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIibLinkSetEntry 4 }

   nhdpIibLinkSetLTime  OBJECT-TYPE
      SYNTAX      TimeStamp
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIibLinkSetLTime specifies
          when this Tuple expires and MUST
          be removed."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIibLinkSetEntry 5 }


  --
  --     IIB 2-Hop Set
  --         Entry (foreach discovered 2-H neighbor
  --                network address): (1HopNeighIfAddrList,
  --                                   2HopNeighNetworkAddr,
  --                                   ExpireTime)
  --
    nhdpIib2HopSetTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF NhdpIib2HopSetEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
           "A 2-Hop Set of an interface records network
           addresses of symmetric 2-hop neighbors, and
           the symmetric links to symmetric 1-hop neighbors
           through which these symmetric 2-hop neighbors
           can be reached.  It consists of 2-Hop Tuples,
           each representing a single network address of
           a symmetric 2-hop neighbor, and a single MANET
           interface of a symmetric 1-hop neighbor.

           (N2_neighbor_iface_addr_list,
            N2_2hop_addr, N2_time)."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpStateObjGrp 5 }

    nhdpIib2HopSetEntry  OBJECT-TYPE
       SYNTAX      NhdpIib2HopSetEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "The entries include the 2-hop neighbor addresses,
           which act as the table index, and associated
           1-hop symmetric link address set, designated
           through nhdpDiscIfIndex, and an expiration time.
           The nhdpIfIndex in the INDEX is
           interface index of the local interface
           through which these 2-hop addresses are
           accessible."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpIfIndex,
               nhdpIib2HopSetIpAddressType,
               nhdpIib2HopSetIpAddress }
    ::= { nhdpIib2HopSetTable 1 }

    NhdpIib2HopSetEntry ::=
       SEQUENCE {
          nhdpIib2HopSetIpAddressType
            InetAddressType,
          nhdpIib2HopSetIpAddress
            InetAddress,
          nhdpIib2HopSetIpAddrPrefixLen
            InetAddressPrefixLength,
          nhdpIib2HopSet1HopIfIndex
            NeighborIfIndex,
          nhdpIib2HopSetN2Time
            TimeStamp
         }

   nhdpIib2HopSetIpAddressType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The type of the nhdpIib2HopSetIpAddress
          in the InetAddress MIB [RFC4001]."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetEntry 1 }

   nhdpIib2HopSetIpAddress  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "nhdpIib2HopSetIpAddr corresponds
         to N2_2hop_addr of NHDP."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetEntry 2 }

   nhdpIib2HopSetIpAddrPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "Indicates the number of leading one bits that
          form the mask.  The mask is logically-ANDed
          to the nhdpIib2HopSetIpAddress to determine
          the address prefix.  A row match is true
          if the address used as an index falls within
          the network address range defined by the
          address prefix."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetEntry 3 }

   nhdpIib2HopSet1HopIfIndex  OBJECT-TYPE
      SYNTAX      NeighborIfIndex
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIib2HopSet1HopIfIndex is
          nhdpDiscIfIndex of the 1-hop
          neighbor which communicated the ipAddress
          of the 2-hop neighbor in this row entry."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetEntry 4 }

   nhdpIib2HopSetN2Time  OBJECT-TYPE
      SYNTAX      TimeStamp
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpIib2HopSetN2Time specifies
          when this column entry expires and
          MUST be removed."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetEntry 5 }


   --
   -- Neighbor Information Base (NIB)
   --
   -- Each router maintains a Neighbor Information Base
   -- that records information about addresses of
   -- current and recently symmetric 1-hop neighbors.
   --     NIB Neighbor Set
   --         Entry (foreach discovered 1-hop neighbor:
   --                N_neighbor_addr_list, N_symmetric)
   --     The NIB Neighbor Set Table is small because
   --     most of the corresponding information is found
   --     in the nhdpDiscoveredIfTable above.
   --
   nhdpNibNeighborSetTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF NhdpNibNeighborSetEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
          "A router's Neighbor Set records all
           network addresses of each 1-hop
           neighbor."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpStateObjGrp 6 }

   nhdpNibNeighborSetEntry  OBJECT-TYPE
      SYNTAX      NhdpNibNeighborSetEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
          "A router's Neighbor Set consists
           of Neighbor Tuples, each representing
           a single 1-hop neighbor:

           (N_neighbor_addr_list, N_symmetric)

           Neighbor tuples are removed from the
           neighbor set only when the
           corresponding link tuples expire from
           the Link Set table."
      REFERENCE
         "[RFC6130]."
      INDEX { nhdpDiscRouterIndex }
   ::= { nhdpNibNeighborSetTable 1 }

    NhdpNibNeighborSetEntry ::=
       SEQUENCE {
         nhdpNibNeighborSetNSymmetric
           TruthValue
        }

   nhdpNibNeighborSetNSymmetric  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpNibNeighborNSymmetric corresponds
         to N_symmetric of NHDP."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpNibNeighborSetEntry 1 }



  --     Lost Neighbor Set
  --         Entry ( foreach recently lost
  --                 1-hop neighbor router):
  --                          (NL_neighbor_addrs, NL_time)
  --
    nhdpNibLostNeighborSetTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF NhdpNibLostNeighborSetEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
           "A router's Lost Neighbor Set records network
           addresses of routers which recently were
           symmetric 1-hop neighbors, but which are now
           advertised as lost."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpStateObjGrp 7 }

    nhdpNibLostNeighborSetEntry  OBJECT-TYPE
       SYNTAX      NhdpNibLostNeighborSetEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "A router's Lost Neighbor Set consists of
          Lost Neighbor Tuples, each representing a
          single such network address:

          (NL_neighbor_addr, NL_time)"
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpDiscRouterIndex }
    ::= { nhdpNibLostNeighborSetTable 1 }

    NhdpNibLostNeighborSetEntry ::=
       SEQUENCE {
          nhdpNibLostNeighborSetNLTime
            TimeStamp
         }
   nhdpNibLostNeighborSetNLTime  OBJECT-TYPE
      SYNTAX      TimeStamp
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "nhdpNibLostNeighborSetNLTime
          specifies when this Tuple expires
          and MUST be removed."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpNibLostNeighborSetEntry 1 }



--
-- nhdpPerformanceObjGrp
--

-- Contains objects which help to characterize the performance of
-- the NHDP process, typically counters.
--
nhdpPerformanceObjGrp OBJECT IDENTIFIER ::= { nhdpObjects 3 }

  --
  -- Objects per local interface
  --

  nhdpInterfacePerfTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF NhdpInterfacePerfEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "This table summarizes performance objects that are
          measured per local NHDP interface."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpPerformanceObjGrp 1 }

   nhdpInterfacePerfEntry OBJECT-TYPE
      SYNTAX      NhdpInterfacePerfEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "A single entry contains performance counters for
          a local NHDP interface."
      INDEX { nhdpIfIndex }
   ::= { nhdpInterfacePerfTable 1 }
   NhdpInterfacePerfEntry ::=
      SEQUENCE {
         nhdpIfHelloMessageXmits
            Counter32,
         nhdpIfHelloMessageRecvd
            Counter32,
         nhdpIfHelloMessageXmitAccumulatedSize
            Counter64,
         nhdpIfHelloMessageRecvdAccumulatedSize
            Counter64,
         nhdpIfHelloMessageTriggeredXmits
            Counter32,
         nhdpIfHelloMessagePeriodicXmits
            Counter32,
         nhdpIfHelloMessageXmitAccumulatedSymmetricNeighborCount
            Counter32,
         nhdpIfHelloMessageXmitAccumulatedHeardNeighborCount
            Counter32,
         nhdpIfHelloMessageXmitAccumulatedLostNeighborCount
            Counter32
         }

   nhdpIfHelloMessageXmits  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented each time a HELLO
         message has been transmitted on that interface."
   ::= { nhdpInterfacePerfEntry 1 }

   nhdpIfHelloMessageRecvd  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented each time a
         HELLO message has been received on that interface."
   ::= { nhdpInterfacePerfEntry 2 }

   nhdpIfHelloMessageXmitAccumulatedSize  OBJECT-TYPE
      SYNTAX      Counter64
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented by the number of octets in
         a HELLO message each time a
         HELLO message has been sent."
   ::= { nhdpInterfacePerfEntry 3 }

   nhdpIfHelloMessageRecvdAccumulatedSize  OBJECT-TYPE
      SYNTAX      Counter64
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented by the number of octets in
         a HELLO message each time a
         HELLO message has been received."
   ::= { nhdpInterfacePerfEntry 4 }

   nhdpIfHelloMessageTriggeredXmits  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented each time a triggered
         HELLO message has been sent."
   ::= { nhdpInterfacePerfEntry 5 }

   nhdpIfHelloMessagePeriodicXmits  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented each time a periodic
         HELLO message has been sent."
   ::= { nhdpInterfacePerfEntry 6 }

   nhdpIfHelloMessageXmitAccumulatedSymmetricNeighborCount  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented by the number of advertised
         symmetric neighbors in a HELLO each time a HELLO
         message has been sent."
   ::= { nhdpInterfacePerfEntry 7 }

   nhdpIfHelloMessageXmitAccumulatedHeardNeighborCount  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented by the number of advertised
         heard neighbors in a HELLO each time a HELLO
         message has been sent."
   ::= { nhdpInterfacePerfEntry 8 }

   nhdpIfHelloMessageXmitAccumulatedLostNeighborCount  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter is incremented by the number of advertised
         lost neighbors in a HELLO each time a HELLO
         message has been sent."
   ::= { nhdpInterfacePerfEntry 9 }



  --
  -- Objects per discovered neighbor interface
  --
  nhdpDiscIfSetPerfTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF NhdpDiscIfSetPerfEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
          "A router's set of performance properties for
          each discovered interface of a neighbor."
       REFERENCE
          "[RFC6130]."
   ::= { nhdpPerformanceObjGrp 2 }


   nhdpDiscIfSetPerfEntry  OBJECT-TYPE
       SYNTAX      NhdpDiscIfSetPerfEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "There is an entry for each discovered
          interface of a neighbor."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpDiscIfIndex }
   ::= { nhdpDiscIfSetPerfTable 1 }

    NhdpDiscIfSetPerfEntry ::=
       SEQUENCE {
          nhdpDiscIfRecvdPackets
            Counter32,
          nhdpDiscIfExpectedPackets
            Counter32
         }
   nhdpDiscIfRecvdPackets  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This counter increments each
         time this router receives a packet from that interface
         of the neighbor."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetPerfEntry 1 }

   nhdpDiscIfExpectedPackets  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This counter increments by the number
          of missed packets from this neighbor based
          on the packet sequence number each time this
          router receives a packet from that interface
          of the neighbor."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscIfSetPerfEntry 2 }



   --
   -- Objects concerning the neighbor set
   --
   nhdpNibNeighborSetChanges  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This counter increments each time the Neighbor Set changes.
         A change occurs whenever a new Neighbor Tuple has been
         added, a Neighbor Tuple has been removed or any entry of
         a Neighbor Tuple has been modified."
   ::= { nhdpPerformanceObjGrp 3 }



   --
   -- Objects per discovered neighbor
   --
   nhdpDiscNeighborSetPerfTable OBJECT-TYPE
```

```
      SYNTAX       SEQUENCE OF NhdpDiscNeighborSetPerfEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
          "A router’s set of discovered neighbors and
           their properties."
       REFERENCE
          "[RFC6130]."
   ::= { nhdpPerformanceObjGrp 4 }

   nhdpDiscNeighborSetPerfEntry  OBJECT-TYPE
       SYNTAX      NhdpDiscNeighborSetPerfEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "The entries include the nhdpDiscRouterIndex of
           the discovered router, as well as performance
           objects related to changes of the Neighbor
           Set."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpDiscRouterIndex }
   ::= { nhdpDiscNeighborSetPerfTable 1 }

    NhdpDiscNeighborSetPerfEntry ::=
       SEQUENCE {
          nhdpDiscNeighborNibNeighborSetChanges
            Counter32,
          nhdpDiscNeighborNibNeighborSetUpTime
            TimeTicks,
          nhdpDiscNeighborNibNeighborSetReachableLinkChanges
            Counter32
         }

   nhdpDiscNeighborNibNeighborSetChanges  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This object returns the number of changes
         to the given Neighbor Tuple."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscNeighborSetPerfEntry 1 }


   nhdpDiscNeighborNibNeighborSetUpTime  OBJECT-TYPE
      SYNTAX      TimeTicks
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This object returns the time in hundredths of a second since
         the neighbor becomes 'nbrup'.  A neighbor is
         said to become 'nbrup' if a new nhdpNibNeighborSetEntry
         is created for a particular nhdpNibNeighborSetRouterIndex.
         It becomes 'nbrdown' if the entry for that neighbor
         has been deleted."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscNeighborSetPerfEntry 2 }

   nhdpDiscNeighborNibNeighborSetReachableLinkChanges  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This object counts each time the neighbor changes
          the interface(s) over which it is reachable.
          A change in the set of Link Tuples corresponding
          to the appropriate Neighbor Tuple is registered,
          i.e. a corresponding Link Tuple is added or removed
          from the set of all corresponding Link Tuples."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpDiscNeighborSetPerfEntry 3 }




   --
   -- Objects per discovered 2-hop neighbor
   --
      nhdpIib2HopSetPerfTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF NhdpIib2HopSetPerfEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
           "This table contains performance objects per
           discovered 2-hop neighbor."
       REFERENCE
          "[RFC6130]."
    ::= { nhdpPerformanceObjGrp 5 }

    nhdpIib2HopSetPerfEntry  OBJECT-TYPE
       SYNTAX      NhdpIib2HopSetPerfEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
          "The entries contain performance objects per
           discovered 2-hop neighbor."
       REFERENCE
          "[RFC6130]."
       INDEX { nhdpDiscRouterIndex }
    ::= { nhdpIib2HopSetPerfTable 1 }

    NhdpIib2HopSetPerfEntry ::=
       SEQUENCE {
          nhdpIib2HopSetPerfChanges
            Counter32,
          nhdpIib2HopSetPerfUpTime
            TimeTicks
         }

   nhdpIib2HopSetPerfChanges  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This object counts the changes of the union of all
         N2_neighbor_iface_addr_list of 2-Hop Tuples with an
         N2_2hop_addr equal to one of the given 2-hop
         neighbor's addresses."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetPerfEntry 1 }


   nhdpIib2HopSetPerfUpTime  OBJECT-TYPE
      SYNTAX      TimeTicks
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "This object returns the time in hundredths of
         a second when the 2-Hop Tuple
         corresponding to the given 2-hop neighbor IP address
         was registered in the nhdpIib2HopSetTable."
      REFERENCE
         "[RFC6130]."
   ::= { nhdpIib2HopSetPerfEntry 2 }
```

```
--
-- nhdpNotifications
--

nhdpNotificationsControl OBJECT IDENTIFIER ::= { nhdpNotifications 1 }
nhdpNotificationsObjects OBJECT IDENTIFIER ::= { nhdpNotifications 2 }
nhdpNotificationsStates  OBJECT IDENTIFIER ::= { nhdpNotifications 3 }


   -- nhdpNotificationsControl

   nhdpSetNotification OBJECT-TYPE
          SYNTAX       OCTET STRING (SIZE(4))
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A 4-octet string serving as a bit map for
             the notification events defined by the NHDP
             notifications. This object is used to enable
             and disable specific NHDP notifications where
             a 1 in the bit field represents enabled. The
             right-most bit (least significant) represents
             notification 0.

             This object is persistent and when written
             the entity SHOULD save the change to
             non-volatile storage.
             "
           ::= { nhdpNotificationsControl 1 }

   nhdpNbrStateChangeThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              nhdpNbrStateChange object.  If the
              number of occurrences exceeds this threshold
              within the previous nhdpNbrStateChangeWindow,
              then the nhdpNbrStateChange notification
              is to be sent.
             "
           ::= { nhdpNotificationsControl 2 }

   nhdpNbrStateChangeWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window for the
              nhdpNbrStateChange object.  If the
              number of occurrences exceeds the
              nhdpNbrStateChangeThreshold
              within the previous nhdpNbrStateChangeWindow,
              then the nhdpNbrStateChange notification
              is to be sent.

              This object represents the time in hundredths
              of a second.
             "
           ::= { nhdpNotificationsControl 3 }

   nhdp2HopNbrStateChangeThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              nhdp2HopNbrStateChange object.  If the
              number of occurrences exceeds this threshold
              within the previous nhdp2HopNbrStateChangeWindow,
              then the nhdp2HopNbrStateChange notification
              is to be sent.
             "
           ::= { nhdpNotificationsControl 4 }

   nhdp2HopNbrStateChangeWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window for the
              nhdp2HopNbrStateChange object.  If the
              number of occurrences exceeds the
              nhdp2HopNbrStateChangeThreshold
              within the previous nhdp2HopNbrStateChangeWindow,
              then the nhdp2HopNbrStateChange notification
              is to be sent.

              This object represents the time in hundredths
              of a second.
             "
           ::= { nhdpNotificationsControl 5 }

   nhdpIfRxBadPacketThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              nhdpIfRxBadPacket object.  If the
              number of occurrences exceeds this threshold
              within the previous nhdpIfRxBadPacketWindow,
              then the nhdpIfRxBadPacket notification
              is to be sent.
             "
           ::= { nhdpNotificationsControl 6 }

   nhdpIfRxBadPacketWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window for the
              nhdpIfRxBadPacket object.  If the
              number of occurrences exceeds the
              nhdpIfRxBadPacketThreshold
              within the previous nhdpIfRxBadPacketWindow,
              then the nhdpIfRxBadPacket notification
              is to be sent.

              This object represents the time in hundredths
              of a second.
             "
           ::= { nhdpNotificationsControl 7 }


   -- nhdpNotificationsObjects

   nhdpNbrStateChange NOTIFICATION-TYPE
          OBJECTS { nhdpIfIndex,  -- The originator of
                                  --     the notification.
                    nhdpNbrState  -- The new state
                  }
          STATUS       current
          DESCRIPTION
             "nhdpNbrStateChange is a notification sent when a
             significant number of neighbors change their status
             (i.e. down, asymmetric, or symmetric) in a short
             time. The network administrator should select
             appropriate values for 'significant number of
             neighbors' and 'short time'."
          ::= { nhdpNotificationsObjects 1 }
```

```
    nhdp2HopNbrStateChange NOTIFICATION-TYPE
          OBJECTS { nhdpIfIndex,      -- The originator
                                      -- of the notification
                    nhdp2HopNbrState  -- The new state
             }
          STATUS       current
          DESCRIPTION
             "nhdp2HopNbrStateChange is a notification sent
             when a significant number of 2-hop neighbors
             change their status (i.e. up or down) in a short
             time. The network administrator should select
             appropriate values for 'significant number of
             neighbors' and 'short time'."
          ::= { nhdpNotificationsObjects 2 }

   nhdpIfRxBadPacket NOTIFICATION-TYPE
          OBJECTS { nhdpDiscRouterIndex, -- The originator of
                                         -- the notification
                    nhdpIfIndex,         -- The interface on which the
                                         -- packet has been received
                    nhdpPacketSrcType,   -- The type of the source IP
                                         -- address of the packet
                    nhdpPacketSrc        -- The source IP address of
                                         -- the packet
             }
          STATUS       current
          DESCRIPTION
             "nhdpIfRxBadPacket is a notification sent when a
             significant number of incoming packets have not
             been successfully parsed in a short time. The
             network administrator should select appropriate
             values for 'significant number of neighbors'
             and 'short time'."
          ::= { nhdpNotificationsObjects 3 }


   nhdpIfStateChange NOTIFICATION-TYPE
          OBJECTS { nhdpIfIndex, -- The local interface
                    nhdpIfState  -- The new state
             }
          STATUS       current
          DESCRIPTION
             "nhdpIfStateChange is a notification sent when
             the status of an interface of this router has
             changed (i.e. an IP address has been added or
             removed to the interface, or the interface has
             changed its status from up to down or vice versa)."
          ::= { nhdpNotificationsObjects 4 }
    -- nhdpNotificationStates

    nhdpNbrState OBJECT-TYPE
       SYNTAX       INTEGER {
                       down (0),
                       asymmetric (1),
                       symmetric(2)
                       }
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
          "NHDP neighbor states."
       DEFVAL { down }
       ::= { nhdpNotificationsStates 1 }

   nhdp2HopNbrState OBJECT-TYPE
       SYNTAX       INTEGER {
                       down (0),
                       up (1)
                       }
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
          "NHDP 2-hop neighbor states."
       DEFVAL { down }
       ::= { nhdpNotificationsStates 2 }

   nhdpIfState OBJECT-TYPE
       SYNTAX       INTEGER {
                       down (0),
                       up (1),
                       addresschange(2) -- If a new address has been
                                        -- added or an address has
                                        -- been removed
                       }
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
          "NHDP interface states."
       DEFVAL { down }
       ::= { nhdpNotificationsStates 3 }

  nhdpPacketSrcType OBJECT-TYPE
            SYNTAX InetAddressType
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
            "The IP address type of the
            address of an inbound packet that
            cannot be identified by a neighbor instance."
            ::= { nhdpNotificationsStates 4 }

   nhdpPacketSrc OBJECT-TYPE
          SYNTAX       InetAddress
          MAX-ACCESS   read-only
          STATUS       current
          DESCRIPTION
             "The IP address of an inbound packet that
             cannot be identified by a neighbor instance. When
             the last value of a notification using this object is
             needed, but no notifications of that type have been sent,
             this value pertaining to this object should
             be returned as 0.0.0.0 or :: respectively."
          ::= { nhdpNotificationsStates 5 }




--
-- nhdpConformance information
--

nhdpCompliances       OBJECT IDENTIFIER ::= { nhdpConformance 1 }
nhdpMIBGroups         OBJECT IDENTIFIER ::= { nhdpConformance 2 }


-- Compliance Statements
nhdpBasicCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The basic implementation requirements for
     managed network entities that implement
     NHDP."
  MODULE -- this module

  MANDATORY-GROUPS { nhdpConfigurationGroup }


  ::= { nhdpCompliances 1 }

nhdpFullCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The full implementation requirements for
     managed network entities that implement
     NHDP."
  MODULE -- this module

  MANDATORY-GROUPS { nhdpConfigurationGroup,
                     nhdpStateGroup,
                     nhdpPerformanceGroup,
                     nhdpNotificationObjectGroup,
                     nhdpNotificationGroup,
                     nhdpPerformanceGroup }

  ::= { nhdpCompliances 2 }


--
-- Units of Conformance
--

  nhdpConfigurationGroup OBJECT-GROUP
     OBJECTS {
             nhdpIfStatus,
             nhdpHelloInterval,
             nhdpHelloMinInterval,
             nhdpRefreshInterval,
             nhdpLHoldTime,
             nhdpHHoldTime,
             nhdpHystAcceptQuality,
             nhdpHystRejectQuality,
             nhdpInitialQuality,
             nhdpInitialPending,
             nhdpHpMaxJitter,
             nhdpHtMaxJitter,
             nhdpNHoldTime,
             nhdpIHoldTime,
             nhdpIfRowStatus,
             nhdpLibLocalIfSetIpAddrType,
             nhdpLibLocalIfSetIpAddr,
             nhdpLibLocalIfSetIpAddrPrefixLen,
             nhdpLibLocalIfSetIsManet,
             nhdpLibRemovedIfAddrSetIpAddrType,
             nhdpLibRemovedIfAddrSetIpAddr,
             nhdpLibRemovedIfAddrSetIpAddrPrefixLen,
             nhdpLibRemovedIfAddrSetIfIndex,
             nhdpLibRemovedIfAddrSetIrTime
            }
      STATUS  current
      DESCRIPTION
         "Set of NHDP configuration objects implemented
          in this module."
   ::= { nhdpMIBGroups 2 }
  nhdpStateGroup OBJECT-GROUP
     OBJECTS {
             nhdpUpTime,
             nhdpIfIndex,
             nhdpIfStateUpTime,
             nhdpDiscRouterIndex,
             nhdpDiscIfIndex,
             nhdpDiscIfSetIpAddrType,
             nhdpDiscIfSetIpAddr,
             nhdpDiscIfSetIpAddrPrefixLen,
             nhdpIibLinkSetLHeardTime,
             nhdpIibLinkSetLSymTime,
             nhdpIibLinkSetLPending,
             nhdpIibLinkSetLLost,
             nhdpIibLinkSetLTime,
             nhdpIib2HopSetIpAddrPrefixLen,
             nhdpIib2HopSet1HopIfIndex,
             nhdpIib2HopSetN2Time,
             nhdpNibNeighborSetNSymmetric,
             nhdpNibLostNeighborSetNLTime
            }
      STATUS  current
      DESCRIPTION
         "Set of NHDP state objects implemented
          in this module."
   ::= { nhdpMIBGroups 3 }

  nhdpPerformanceGroup OBJECT-GROUP
     OBJECTS {
             nhdpIfHelloMessageXmits,
             nhdpIfHelloMessageRecvd,
             nhdpIfHelloMessageXmitAccumulatedSize,
             nhdpIfHelloMessageRecvdAccumulatedSize,
             nhdpIfHelloMessageTriggeredXmits,
             nhdpIfHelloMessagePeriodicXmits,
             nhdpIfHelloMessageXmitAccumulatedSymmetricNeighborCount,
             nhdpIfHelloMessageXmitAccumulatedHeardNeighborCount,
             nhdpIfHelloMessageXmitAccumulatedLostNeighborCount,
             nhdpDiscIfRecvdPackets,
             nhdpDiscIfExpectedPackets,
             nhdpNibNeighborSetChanges,
             nhdpDiscNeighborNibNeighborSetChanges,
             nhdpDiscNeighborNibNeighborSetUpTime,
             nhdpDiscNeighborNibNeighborSetReachableLinkChanges,
             nhdpIib2HopSetPerfChanges,
             nhdpIib2HopSetPerfUpTime
           }
      STATUS  current
      DESCRIPTION
         "Set of NHDP performance objects implemented
          in this module."
   ::= { nhdpMIBGroups 4 }

    nhdpNotificationObjectGroup OBJECT-GROUP
      OBJECTS {
           nhdpSetNotification,
           nhdpNbrStateChangeThreshold,
           nhdpNbrStateChangeWindow,
           nhdp2HopNbrStateChangeThreshold,
           nhdp2HopNbrStateChangeWindow,
           nhdpIfRxBadPacketThreshold,
           nhdpIfRxBadPacketWindow,
           nhdpIfState,
           nhdpNbrState,
           nhdp2HopNbrState,
           nhdpPacketSrcType,
           nhdpPacketSrc
      }
      STATUS current
      DESCRIPTION
      "Set of NHDP notification objects implemented
      in this module."
   ::= { nhdpMIBGroups 5 }



  nhdpNotificationGroup NOTIFICATION-GROUP
     NOTIFICATIONS {
            nhdpNbrStateChange,
            nhdp2HopNbrStateChange,
            nhdpIfRxBadPacket,
            nhdpIfStateChange
            }
      STATUS  current
      DESCRIPTION
         "Set of NHDP notifications implemented
          in this module."
   ::= { nhdpMIBGroups 6 }


END
```

8. Security Considerations

This MIB defines objects for the configuration, monitoring and notification of the Neighborhood Discovery Protocol [RFC6130]. NHDP allows routers to acquire topological information up to two hops away by virtue of exchanging HELLO messages. The information acquired by NHDP may be used by routing protocols. The neighborhood information, exchanged between routers using NHDP, serves these routing protocols as a baseline for calculating paths to all destinations in the MANET, relay set selection for network-wide transmissions etc.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

o nhdpIfStatus - this writable object turns on or off the NHDP process for the specified interface. If disabled, higher level protocol functions, e.g., routing, would fail causing network-wide disruptions.

o nhdpHelloInterval, nhdpHelloMinInterval, and nhdpRefreshInterval - these writable objects control the rate at which HELLO messages are sent on a wireless interface. If set at too high a rate, this could represent a form of DOS attack by overloading interface resources.

o nhdpHystAcceptQuality, nhdpHystRejectQuality, nhdpInitialQuality, nhdpInitialPending - these writable objects affect the perceived quality of the NHDP links and hence the overall stability of the network. If improperly set, these settings could result in network-wide disruptions.

o nhdpInterfaceTable - this table contains writable objects that affect the overall performance and stability of the NHDP process. Failure of the NHDP process would result in network-wide failure. Particularly sensitive objects from this table are discussed in the previous list items. This is the only table in the NHDP-MIB with writable objects.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- nhdpDiscIfSetTable - The contains information on discovered neighbors, specifically their IP address in the nhdpDiscIfSetIpAddr object. This information provides an adversary broad information on the members of the MANET, located within this single table. This information can be use to expedite attacks on the other members of the MANET without having to go through a laborious discovery process on their own. This object is the index into the table, and has a MAX-ACCESS of 'not-accessible'. However, this information can be exposed using SNMP operations.

MANET technology is often deployed to support communications of emergency services or military tactical applications. In these applications, it is imperative to maintain the proper operation of the communications network and to protect sensitive information related to its operation. Therefore, when implementing these capabilities, the full use of SNMPv3 cryptographic mechanisms for authentication and privacy is RECOMMENDED.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

9. IANA Considerations

Editor's Note (to be removed prior to publication): the IANA is requested to assign a value for "XXXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXXX" (here and in the MIB module) with the assigned value and to remove this note. Note well: prior to official assignment by the IANA, a draft document MUST use placeholders (such as "XXXX" above) rather than actual numbers. See RFC4181 Section 4.5 for an example of how this is done in a draft MIB module.

10. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

11. Acknowledgements

The authors wish to thank Thomas Clausen and Justin Dean for their detailed reviews and insightful comments to this document.

12. References

12.1. Normative References

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

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

[RFC6340] Presuhn, R., "Textual Conventions for the Representation of Floating-Point Numbers", RFC 6340, August 2011.

12.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC4750] Joyal, D., Galecki, P., Giacalone, S., Coltun, R., and F. Baker, "OSPF Version 2 Management Information Base", RFC 4750, December 2006.

Appendix A.

```
***************************************************************
* Note to the RFC Editor (to be removed prior to publication) *
*                                                             *
* The reference to RFCXXXX within the DESCRIPTION clauses     *
* of the MIB module point to this draft and are to be         *
* assigned by the RFC Editor.                                 *
*                                                             *
***************************************************************
```

Authors' Addresses

Ulrich Herberg
LIX, Ecole Polytechnique
Palaiseau Cedex,   91128
France

EMail: ulrich@herberg.name
URI: http://www.herberg.name/

Robert G. Cole
US Army CERDEC
6010 Frankford Road, Bldg 6010
Aberdeen Proving Ground, Maryland 21005
USA

Phone: +1 443 395 8744
EMail: robert.g.cole@us.army.mil
URI: http://www.cs.jhu.edu/˜rgcole/

Ian D Chakeres
CenGen
9250 Bendix Road North
Columbia, Maryland 560093
USA

EMail: ian.chakeres@gmail.com
URI: http://www.ianchak.com/

Mobile Ad hoc Networking (MANET)
Internet-Draft
Intended status: Standards Track
Expires: April 16, 2012

T. Clausen
LIX, Ecole Polytechnique
C. Dearlove
BAE Systems ATC
P. Jacquet
Project Hipercom, INRIA
October 14, 2011

The Optimized Link State Routing Protocol version 2
draft-ietf-manet-olsrv2-13

Abstract

This document describes version 2 of the Optimized Link State Routing (OLSRv2) protocol for Mobile Ad hoc NETworks (MANETs).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Optimized Link State Routing protocol version 2 (OLSRv2) is an update to OLSR (version 1) as published in [RFC3626]. Compared to [RFC3626], OLSRv2 retains the same basic mechanisms and algorithms, enhanced by the ability to use a link metric other than hop count in the selection of shortest routes. OLSRv2 also uses a more flexible and efficient signaling framework, and includes some simplification of the messages being exchanged.

OLSRv2 is developed for mobile ad hoc networks (MANETs). It operates as a table driven, proactive protocol, i.e., it exchanges topology information with other routers in the network regularly. OLSRv2 is an optimization of the classical link state routing protocol. Its key concept is that of MultiPoint Relays (MPRs). Each router selects two sets of MPRs, each being a set of its neighbor routers that "cover" all of its symmetrically connected 2-hop neighbor routers. These two sets are of flooding MPRs and routing MPRs, and are used to achieve flooding reduction and topology reduction, respectively.

Flooding reduction is achieved by control traffic being flooded through the network using hop by hop forwarding, but with a router only needing to forward control traffic that is first received directly from one of the routers that have selected it as a flooding MPR (its "flooding MPR selectors"). This mechanism, denoted "MPR flooding", provides an efficient mechanism for information distribution within the MANET by reducing the number of transmissions required.

Topology reduction is achieved by assigning a special responsibility to routers selected as routing MPRs when declaring link state information. A sufficient requirement for OLSRv2 to provide shortest routes to all destinations is that routers declare link state information for their routing MPR selectors, if any. Routers that are not selected as routing MPRs need not send any link state information. Based on this reduced link state information, routing MPRs are used as intermediate routers in multi-hop routes.

Thus the use of MPRs allows reduction of the number and the size of link state messages, and in the amount of link state information maintained in each router. When possible (in particular if using a hop count metric) the same routers may be picked as both flooding MPRs and routing MPRs.

A router selects both routing and flooding MPRs from among its one hop neighbors connected by "symmetric", i.e., bidirectional, links. Therefore, selecting routes through routing MPRs avoids the problems associated with data packet transfer over unidirectional links (e.g., the problem of not getting link layer acknowledgments at each hop, for link layers employing this technique).

OLSRv2 uses and extends the MANET NeighborHood Discovery Protocol (NHDP) defined in [RFC6130] and also uses the MANET generalized packet/message format [RFC5444] and the specifications in [RFC5497] and, optionally, [RFC5148]. These four other protocols and specifications were all originally created as part of OLSRv2, but have been specified separately for wider use.

OLSRv2 makes no assumptions about the underlying link layer. OLSRv2, through its use of [RFC6130], may use link layer information and notifications when available and applicable. In addition OLSRv2 uses link metrics that may be derived from link layer or any other information. OLSRv2 does not specify the physical meaning of link metrics, but specifies a means by which new types of link metrics may be specified in the future, but used by OLSRv2 without modification.

OLSRv2, as OLSR [RFC3626], inherits its concept of forwarding and relaying from HIPERLAN (a MAC layer protocol) which is standardized by ETSI [HIPERLAN], [HIPERLAN2].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

All terms introduced in [RFC5444], including "packet", "Packet Header", "message", "Message Header", "Message Body", "Message Type", "message sequence number", "hop limit", "hop count", "Address Block", "TLV Block", "TLV", "Message TLV", "Address Block TLV", "type" (of TLV), "type extension" (of TLV), "value" (of TLV), "address", "address prefix", and "address object" are to be interpreted as described there.

All terms introduced in [RFC6130], including "interface", "MANET interface", "network address", "link", "symmetric link", "1-hop neighbor", "symmetric 1-hop neighbor", "symmetric 2-hop neighbor", "constant", "interface parameter", "router parameter", "Information Base", and "HELLO message" are to be interpreted as described there.

Additionally, this specification uses the following terminology:

Router:
A MANET router which implements this protocol.

OLSRv2 interface:
A MANET interface running this protocol.

Routable address:
A network address which may be used as the destination of a data packet. A router MUST be able to distinguish a routable address from a non-routable address by direct inspection of the network address, based on global scope address allocations by IANA and/or administrative configuration. Broadcast, multicast and anycast addresses, and addresses which are limited in scope to less than the entire MANET, MUST NOT be considered as routable addresses.

Originator address
An address which is unique (within the MANET) to a router. A router MUST select an originator address; it MAY choose one of its interface addresses as its originator address; it MAY select either a routable or non-routable address. If it selects a routable address then this MUST be one which the router will accept as destination. An originator address MUST NOT have a prefix length, except for when included in an Address Block where it MAY be associated with a prefix of maximum prefix length (e.g., if the originator address is an IPv6 address, it MUST have either no prefix length, or have a prefix length of 128).

Message originator address
The originator address of the router which created a message, as deduced from that message by its recipient. The message originator address will usually be included in the message as its <msg-orig-addr> element as defined in [RFC5444]. However an exceptional case in a HELLO message is also allowed by this specification, when a router only uses a single address. For all messages used in this specification, including HELLO messages defined in [RFC6130], the recipient MUST be able to deduce an originator address.

Willingness:
A numerical value between WILL_NEVER and WILL_ALWAYS (both inclusive), that represents the router's willingness to be selected as an MPR. A router has separate willingness values to be a flooding MPR and a routing MPR.

Willing symmetric 1-hop neighbor
A symmetric 1-hop neighbor of this router that has willingness not equal to WILL_NEVER.

Multipoint relay (MPR):

A router, X, is an MPR for a router, Y, if router Y has indicated its selection of router X as an MPR in a recent HELLO message. Router X may be a flooding MPR for Y, if it is indicated to participate in the flooding process of messages received from router Y, or it may be a routing MPR for Y, if it is indicated to declare link-state information for the link from X to Y. It may also be both at the same time.

MPR selector:

A router, Y, is a flooding/routing MPR selector of router X if router Y has selected router X as a flooding/routing MPR.

MPR flooding:

The optimized MANET-wide information distribution mechanism, employed by this protocol, in which a message is relayed by only a reduced subset of the routers in the network. MPR flooding is the mechanism by which flooding reduction is achieved.

This document employs the same notational conventions as in [RFC5444] and [RFC6130].

3. Applicability Statement

This protocol:

- Is a proactive routing protocol for mobile ad hoc networks (MANETs) [RFC2501].

- Is designed to work in networks with a dynamic topology, and in which messages may be lost, such as due to collisions in wireless networks.

- Supports routers that each have one or more participating OLSRv2 interfaces. The set of a router's interfaces may change over time. Each OLSRv2 interface may have one or more network addresses (which may have prefix lengths), and these may also be dynamically changing.

- Enables hop-by-hop routing, i.e., each router can use its local information provided by this protocol to route packets.

- Continuously maintains routes to all destinations in the network, i.e., routes are instantly available and data traffic is subject to no delays due to route discovery. Consequently, no data traffic buffering is required.

- Supports routers that have non-OLSRv2 interfaces which may be local to a router or that can serve as gateways towards other networks.

- Enables the use of bidirectional additive link metrics to use shortest distance routes (i.e., routes with smallest total of link metrics). Incoming link metric values are to be determined by a process outside this specification.

- Is optimized for large and dense networks; the larger and more dense a network, the more optimization can be achieved by using MPRs, compared to the classic link state algorithm.

- Uses [RFC5444] as described in its "Intended Usage" appendix and by [RFC5498].

- Allows "external" and "internal" extensibility (adding new message types and adding information to existing messages) as enabled by [RFC5444].

- Is designed to work in a completely distributed manner, and does not depend on any central entity.

4. Protocol Overview and Functioning

The objectives of this protocol are for each router to, independently:

- Identify all destinations in the network.

- Identify a sufficient subset of links in the network, in order that shortest paths can be calculated to all available destinations.

- Provide a Routing Set, containing these shortest paths from this router to all destinations (routable addresses and local links).

4.1. Overview

These objectives are achieved, for each router, by:

- Using [RFC6130] to identify symmetric 1-hop neighbors and symmetric 2-hop neighbors.

- Extending [RFC6130] to allow the addition of directional link metrics to advertised links, and to indicate which link metric type is being used by that router. Both incoming and outgoing link metrics may be reported, the latter determined by the advertising router.

- Selecting flooding MPRs and routing MPRs from among its symmetric 1-hop neighbors such that, for each set of MPRs all symmetric 2-hop neighbors are reachable either directly or via at least one symmetric 1-hop neighbor, using a path of minimum total metric where appropriate. An analysis and examples of MPR selection algorithms are given in [MPR]; a suggested algorithm, appropriately adapted for each set of MPRs, is included in this specification. Note that it is not necessary for routers to use the same algorithm in order to interoperate in the same MANET, but these algorithms must each have the appropriate properties.

- Signaling its flooding MPR and routing MPR selections by extending [RFC6130] to report this information in outgoing HELLO messages, by the addition of MPR Address Block TLV(s) associated with the appropriate network addresses.

- Extracting its flooding MPR selectors and routing MPR selectors from received HELLO messages, using the included MPR Address Block TLV(s).

- Reporting its willingness to be a flooding MPR and to be a routing MPR in HELLO messages, by the addition of an MPR_WILLING Message TLV. The router's flooding willingness indicates how willing it is to participate in MPR flooding and the router's routing willingness indicates how willing it is to be an intermediate node for routing, while still being able to be a routing source or destination even if unwilling to perform either function.

- Using the message format specified in [RFC5444], specifically defining a TC (Topology Control) Message Type, used to periodically signal links between routing MPR selectors and itself throughout the MANET. This signaling includes suitable directional neighbor metrics (the best link metric in that direction between those routers).

- Allowing its TC messages, as well as HELLO messages, to be included in packets specified in [RFC5444], using the "manet" IP protocol or UDP port as specified in [RFC5498].

- Diffusing TC messages by using a flooding reduction mechanism, denoted "MPR flooding"; only the flooding MPRs of a router will retransmit messages received from (i.e., originated or last relayed by) that router.

Note that the indicated extensions to [RFC6130] are of forms permitted by that specification.

This specification defines:

- The requirement to use [RFC6130], its parameters, constants, HELLO messages, and Information Bases, each as extended in this specification.

- Two new Information Bases: the Topology Information Base and the Received Message Information Base.

- TC messages, which are used for MANET wide signaling (using MPR flooding) of selected topology (link state) information.

- A requirement for each router to have an originator address to be included in, or deducible from, HELLO messages and TC messages.

- The specification of new Message TLVs and Address Block TLVs that are used in HELLO messages and TC messages, including for reporting link metrics and their usage, willingness to be an MPR, MPR selection, and content sequence number information. Note that the generation of (incoming) link metric values is to be undertaken by a process outside this specification; this specification concerns only the distribution and use of those metrics.

- The generation of TC messages from the appropriate information in the Information Bases.

- The updating of the Topology Information Base according to received TC messages.

- The MPR flooding mechanism, including the inclusion of message originator address and sequence number to manage duplicate messages, using information recorded in the Received Message Information Base.

- The response to other events, such as the expiration of information in the Information Bases.

This protocol inherits the stability of a link state algorithm, and has the advantage of having routes immediately available when needed, due to its proactive nature.

This protocol only interacts with IP through routing table management, and the use of the sending IP address for IP datagrams containing OLSRv2 packets.

4.2. Routers and Interfaces

In order for a router to participate in a MANET using this protocol it MUST have at least one, and possibly more, OLSRv2 interfaces. Each OLSRv2 interface:

- Is configured with one or more network addresses, as specified in [RFC6130]. These addresses MUST each be specific to this router, and MUST include any address that will be used as the sending address of any IP packet sent on this OLSRv2 interface.

- Has a number of interface parameters, adding to those specified in [RFC6130].

- Has an Interface Information Base, extending that specified in [RFC6130].

- Generates and processes HELLO messages according to [RFC6130], extended as specified in Section 15.

In addition to a set of OLSRv2 interfaces as described above, each router:

- May have one or more non-OLSRv2 interfaces and/or local attached networks for which this router can accept packets. All routable addresses for which the router is to accept packets MUST be used as an (OLSRv2 or non-OLSRv2) interface network address or as an address of a local attached network of the router.

- Has a number of router parameters, adding to those specified in [RFC6130].

- Has a Local Information Base, extending that specified in [RFC6130], including selection of an originator address and recording any locally attached networks.

- Has a Neighbor Information Base, extending that specified in [RFC6130] to record MPR selection and advertisement information.

- Has a Topology Information Base, recording information received in TC messages.

- Has a Received Message Information Base, recording information about received messages to ensure that each TC message is only processed once, and forwarded at most once on each OLSRv2 interface, by a router.

- Generates and processes TC messages.

4.3. Information Base Overview

Each router maintains the Information Bases described in the following sections. These are used for describing the protocol in this specification. An implementation of this protocol MAY maintain this information in the indicated form, or in any other organization which offers access to this information. In particular, note that it is not necessary to remove Tuples from Sets at the exact time indicated, only to behave as if the Tuples were removed at that time.

4.3.1. Local Information Base

The Local Information Base is specified in [RFC6130], and contains a router’s local configuration. It is extended in this specification to also record an originator address, and to include a router’s:

- Originator Set, containing addresses that were recently used as this router’s originator address, and is used, together with the router’s current originator address, to enable a router to recognize and discard control traffic which was originated by the router itself.

- Local Attached Network Set, containing network addresses of networks to which this router can act as a gateway, and advertises in its TC messages.

4.3.2. Interface Information Bases

The Interface Information Bases, one for each OLSRv2 interface, are specified in [RFC6130], and are extended to also record, in each Link Set, link metric values (incoming and outgoing) and flooding MPR selector information.

4.3.3. Neighbor Information Base

The Neighbor Information Base is specified in [RFC6130], and is extended to also record, in the Neighbor Tuple for each neighbor:

- Its originator address.

- Neighbor metric values, these being the minimum of the link metric values in the indicated direction for all symmetric 1-hop links with that neighbor.

- Its willingness to be a flooding MPR and to be a routing MPR.

- Whether it has been selected by this router as a flooding MPR or as a routing MPR, and whether it is a routing MPR selector of this router. (Whether it is a flooding MPR selector of this neighbor is recorded in the Interface Information Base.)

- Whether it is to be advertised in TC messages sent by this router.

4.3.4. Topology Information Base

The Topology Information Base in each router contains:

- An Advertising Remote Router Set, recording each other router from which TC messages have been received. This is used in order to determine if a received TC message contains fresh or outdated information; a received TC message is ignored in the latter case.

- A Router Topology Set, recording links between routers in the MANET, as described by received TC messages.

- A Routable Address Topology Set, recording routable addresses in the MANET (available as packet destinations) and from which other router these routable addresses can be directly reached (i.e., in a single IP hop), as reported by received TC messages.

- An Attached Network Set, recording networks to which a remote router has advertised that it may act as a gateway. These networks may be reached in one or more IP hops.

- A Routing Set, recording routes from this router to all available destinations. The IP routing table is to be updated using this Routing Set. (A router MAY choose to use any or all destination network addresses in the Routing Set to update the IP routing table, this selection is outside the scope of this specification.)

The purpose of the Topology Information Base is to record information used, in addition to that in the Local Information Base, the Interface Information Bases and the Neighbor Information Base, to construct the Routing Set (which is also included in the Topology Information Base).

This specification describes the calculation of the Routing Set based on a Topology Graph constructed in two phases. First, a "backbone" graph representing the routers in the MANET, and the connectivity between them, is constructed from the Local Information Base, the Neighbor Information Base and the Router Topology Set. Second, this graph is "decorated" with additional destination network addresses using the Local Information Base, the Routable Address Topology Set and the Attached Network Set.

The Topology Graph does not need to be recorded in the Topology Information Base, it can either be constructed as required when the Routing Set is to be changed, or need not be explicitly constructed (as illustrated in Appendix B). An implementation MAY construct and retain the Topology Graph if preferred.

4.3.5. Received Message Information Base

The Received Message Information Base in each router contains:

- A Received Set for each OLSRv2 interface, describing TC messages received by this router on that OLSRv2 interface.

- A Processed Set, describing TC messages processed by this router.

- A Forwarded Set, describing TC messages forwarded by this router.

The Received Message Information Base serves the MPR flooding mechanism by ensuring that received messages are forwarded at most once by a router, and also ensures that received messages are processed exactly once by a router. The Received Messages Information Base MAY also record information about other message types that use the MPR flooding mechanism.

4.4. Signaling Overview

This protocol generates and processes HELLO messages according to [RFC6130], extended according to Section 15 of this specification to include an originator address, link metrics, and MPR selection information.

This specification defines a single message type, the TC message. TC messages are sent by their originating router proactively, at a regular interval. This interval may be fixed, or may be dynamic, for example it may be backed off due to congestion or network stability. TC messages may also be sent as a response to a change in the router itself, or its advertised 1-hop neighborhood, for example on first being selected as a routing MPR.

Because TC messages are sent periodically, this protocol is tolerant of unreliable transmissions of TC messages. Message losses may occur more frequently in wireless networks due to collisions or other transmission problems. This protocol may use "jitter", randomized adjustments to message transmission times, to reduce the incidence of collisions, as specified in [RFC5148].

This protocol is tolerant of out of sequence delivery of TC messages due to in transit message reordering. Each router maintains an

Advertised Neighbor Sequence Number (ANSN) that is incremented when its recorded neighbor information that is to be included in its TC messages changes. This ANSN is included in the router’s TC messages. The recipient of a TC message can used this included ANSN to identify which of the information it has received is most recent, even if messages have been reordered while in transit. Only the most recent information received is used, older information received later is discarded.

TC messages may be "complete" or "incomplete". A complete TC message advertises all of the originating router’s routing MPR selectors, it may also advertise other symmetric 1-hop neighbors. Complete TC messages are generated periodically (and also, optionally, in response to neighborhood changes). Incomplete TC messages may be used to report additions to advertised information, without repeating unchanged information.

TC messages, and HELLO messages as extended by this specification, include an originator address for the router that created the message. A TC message reports both the originator addresses and routable addresses of its advertised neighbors, distinguishing the two using an Address Block TLV (an address may be both routable and an originator address). TC messages also report the originator’s locally attached networks.

TC messages are MPR flooded throughout the MANET. A router retransmits a TC message received on an OLSRv2 interface if and only if the message did not originate at this router and has not been previously forwarded by this router, this is the first time the message has been received on this OLSRv2 interface, and the message is received from (i.e., originated from or was last relayed by) one of this router’s flooding MPR selectors.

Some TC messages may be MPR flooded over only part of the network, e.g., allowing a router to ensure that nearer routers are kept more up to date than distant routers, such as is used in Fisheye State Routing [FSR] and Fuzzy Sighted Link State routing [FSLS]. This is enabled using [RFC5497].

TC messages include outgoing neighbor metrics that will be used in the selection of routes.

4.5. Link Metrics

OLSRv1 [RFC3626] created minimum hop routes to destinations. However in many, if not most, circumstances, better routes (in terms of quality of service for end users) can be created by use of link metrics.

OLSRv2, as defined in this specification, allows links to have a metric (also known as a cost). Link metrics as defined in OLSRv2 are additive, and the routes that are to be created are minimum length routes, where the length of a route is defined as the sum of the metrics of the links in that route.

Link metrics are defined to be directional; the link metric from one router to another may be different from that on the reverse link. The link metric is assessed at the receiver, as on a (typically) wireless link, that is the better informed as to link information. Both incoming and outgoing link information is used by OLSRv2, the distinctions in the specification must be clearly followed.

This specification also defines both incoming and outgoing neighbor metrics for each symmetric 1-hop neighbor, these being the minimum value of the link metrics in the same direction for all symmetric links with that neighbor. Note that this means that all neighbor metric values are link metric values and that specification of, for example, link metric value encoding also includes neighbor metric values.

This specification does not define the nature of the link metric. However this specification allows, through use of the type extension of a defined Address Block TLV, for link metrics with specific meanings to be defined and either allocated by IANA or privately used. Each HELLO or TC message carrying link (or neighbor) metrics thus indicates which link metric information it is carrying, thus allowing routers to determine if they can interoperate. If link metrics require additional signaling to determine their values, whether in HELLO messages or otherwise, then this is permitted but is outside the scope of this specification.

Users are advised that they should carefully consider how to use link metrics. In particular they should not simply default to use of all links with equal metrics (i.e. hop count) for routing without careful consideration of whether that is advisable or not.

4.6. Routing Set Use

The purpose of the Routing Set is to determine and record routes (local interface network address and next hop interface network address) to all possible routable addresses advertised by this protocol, as well as of all destinations that are local, i.e., within one hop, to the router (whether using routable addresses or not). Only symmetric links are used in such routes.

It is intended that the Routing Set can be used for packet routing, by using its contents to update IP’s routing tables. That update, and whether any Routing Tuples are not used in IP's routing table, is outside the scope of this specification.

The signaling in this specification has been designed so that a "backbone" Topology Graph of routers, each identified by its originator address, with at most one direct connection between any pair of routers, can be constructed (from the Neighbor Set and the Router Topology Set) using a suitable minimum path length algorithm. This Topology Graph can, then, have other network addresses (routable or of symmetric 1-hop neighbors) added to it (using the Interface Information Bases, the Routable Address Topology Set and the Attached Network Set).

5. Protocol Parameters and Constants

The parameters and constants used in this specification are those defined in [RFC6130] plus those defined in this section. The separation in [RFC6130] into interface parameters, router parameters and constants is also used in this specification.

As for the parameters in [RFC6130], parameters defined in this specification MAY be changed dynamically by a router, and need not be the same on different routers, even in the same MANET, or, for interface parameters, on different interfaces of the same router.

5.1. Protocol and Port Numbers

This protocol specifies TC messages, which are included in packets as defined by [RFC5444]. These packets may be sent either using the "manet" protocol number or the "manet" UDP well-known port number, as specified in [RFC5498].

TC messages and HELLO messages [RFC6130] SHOULD, in a given deployment of this protocol, both be using the same of either of IP or UDP, in order that it is possible to combine messages of both protocols into the same [RFC5444] packet for transmission.

5.2. Multicast Address

This protocol specifies TC messages, which are included in packets as defined by [RFC5444]. These packets MAY be transmitted using the link local multicast address "LL-MANET-Routers", as specified in [RFC5498].

5.3. Interface Parameters

5.3.1. Received Message Validity Time

The following parameter manages the validity time of recorded received message information:

RX_HOLD_TIME:
The period after receipt of a message by the appropriate OLSRv2 interface of this router for which that information is recorded, in order that the message is recognized as having been previously received on this OLSRv2 interface.

The following constraints apply to this parameter:

- RX_HOLD_TIME > 0

- RX_HOLD_TIME SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

5.4. Router Parameters

5.4.1. Local History Times

The following router parameter manages the time for which local information is retained:

O_HOLD_TIME:
The time for which a recently used and replaced originator address is used to recognize the router's own messages.

The following constraint apply to this parameter:

- O_HOLD_TIME > 0

5.4.2. Link Metric Parameters

All routes found using this specification use a single link metric type that is specified by the router parameter LINK_METRIC_TYPE, which may take any value from 0 to 255, inclusive.

5.4.3. Message Intervals

The following parameters regulate TC message transmissions by a router. TC messages are usually sent periodically, but MAY also be sent in response to changes in the router's Neighbor Set and/or Local Attached Network Set. In a highly dynamic network, and with a larger value of the parameter TC_INTERVAL and a smaller value of the parameter TC_MIN_INTERVAL, TC messages may be transmitted more often in response to changes than periodically. However because a router has no knowledge of, for example, routers remote to it (i.e., beyond two hops away) joining the network, TC messages MUST NOT be sent purely responsively.

TC_INTERVAL:
The maximum time between the transmission of two successive TC messages by this router. When no TC messages are sent in response to local network changes (by design, or because the local network is not changing) then TC messages SHOULD be sent at a regular interval TC_INTERVAL, possibly modified by jitter as specified in [RFC5148].

TC_MIN_INTERVAL:
The minimum interval between transmission of two successive TC messages by this router. (This minimum interval MAY be modified by jitter, as specified in [RFC5148].)

The following constraints apply to these parameters:

- TC_INTERVAL > 0

- TC_MIN_INTERVAL >= 0

- TC_INTERVAL >= TC_MIN_INTERVAL

- If TLVs with Type = INTERVAL_TIME, as defined in [RFC5497], are included in TC messages, then TC_INTERVAL MUST be representable as described in [RFC5497].

5.4.4. Advertised Information Validity Times

The following parameters manage the validity time of information advertised in TC messages:

T_HOLD_TIME:
Used as the minimum value in the TLV with Type = VALIDITY_TIME included in all TC messages sent by this router. If a single value of parameter TC_HOP_LIMIT (see Section 5.4.7) is used then this will be the only value in that TLV.

A_HOLD_TIME:
The period during which TC messages are sent after they no longer have any advertised information to report, but are sent in order to accelerate outdated information removal by other routers.

The following constraints apply to these parameters:

- T_HOLD_TIME > 0

- A_HOLD_TIME >= 0

- T_HOLD_TIME >= TC_INTERVAL

- If TC messages can be lost, then both T_HOLD_TIME and A_HOLD_TIME SHOULD be significantly greater than TC_INTERVAL; a value >= 3 x TC_INTERVAL is RECOMMENDED.

- T_HOLD_TIME MUST be representable as described in [RFC5497].

5.4.5. Processing and Forwarding Validity Times

The following parameters manage the processing and forwarding validity time of recorded message information:

P_HOLD_TIME:
The period after receipt of a message that is processed by this router for which that information is recorded, in order that the message is not processed again if received again.

F_HOLD_TIME:
The period after receipt of a message that is forwarded by this router for which that information is recorded, in order that the message is not forwarded again if received again.

The following constraints apply to these parameters:

- P_HOLD_TIME > 0

- F_HOLD_TIME > 0

- Both of these parameters SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

5.4.6. Jitter

If jitter, as defined in [RFC5148], is used then the governing jitter parameters are as follows:

TP_MAXJITTER:
Represents the value of MAXJITTER used in [RFC5148] for periodically generated TC messages sent by this router.

TT_MAXJITTER:
Represents the value of MAXJITTER used in [RFC5148] for externally triggered TC messages sent by this router.

F_MAXJITTER:
Represents the default value of MAXJITTER used in [RFC5148] for messages forwarded by this router. However before using F_MAXJITTER a router MAY attempt to deduce a more appropriate value of MAXJITTER, for example based on any TLVs with Type = INTERVAL_TIME or Type = VALIDITY_TIME contained in the message to be forwarded.

For constraints on these parameters see [RFC5148].

5.4.7. Hop Limit

The parameter TC_HOP_LIMIT is the hop limit set in each TC message. TC_HOP_LIMIT MAY be a single fixed value, or MAY be different in TC messages sent by the same router. However each other router, at any hop count distance, SHOULD see a regular pattern of TC messages in order that meaningful values of TLVs with Type = INTERVAL_TIME and Type = VALIDITY_TIME at each hop count distance can be included as defined in [RFC5497]. Thus the pattern of TC_HOP_LIMIT SHOULD be defined to have this property. For example the repeating pattern (255 4 4) satisfies this property (having period TC_INTERVAL at hop counts up to 4, inclusive, and 3 x TC_INTERVAL at hop counts greater than 4), but the repeating pattern (255 255 4 4) does not satisfy this property because at hop counts greater than 4, message intervals are alternately TC_INTERVAL and 3 x TC_INTERVAL.

The following constraints apply to this parameter:

o The maximum value of TC_HOP_LIMIT >= the network diameter in hops, a value of 255 is RECOMMENDED. Note that if using a pattern of different values of TC_HOP_LIMIT as described above, then only the maximum value in the pattern is so constrained.

o All values of TC_HOP_LIMIT >= 2.

5.4.8. Willingness

Each router has two willingness parameters: WILL_FLOODING and WILL_ROUTING, each of which MUST be in the range WILL_NEVER to WILL_ALWAYS, inclusive.

WILL_FLOODING represents the router’s willingness to be selected as a flooding MPR and hence to participate in MPR flooding, in particular of TC messages.

WILL_ROUTING represents the router's willingness to be selected as a routing MPR and hence to be an intermediate router on routes.

In either case, the higher the value, the greater the router's willingness to be a flooding or routing MPR, respectively. If a router has a willingness value of WILL_NEVER (the lowest possible value) it does not perform the corresponding task. A MANET using this protocol with too many routers having either willingness value equal to WILL_NEVER will not function; it MUST be ensured, by administrative or other means, that this does not happen.

If a router has a willingness value equal to WILL_ALWAYS (the highest possible value) then it will always be selected as a flooding or routing MPR, respectively, by all symmetric 1-hop neighbors.

A MANET in which all routers have WILL_FLOODING = WILL_ALWAYS, the flooding operation will effectively disable optimizations, and perform as blind flooding.

A router, which has WILL_ROUTING = WILL_NEVER will not act as an intermediate router in the MANET. Such a router can, act as a source, destination or gateway to another routing domain.

Different routers MAY have different values for WILL_FLOODING and/or WILL_ROUTING. A router that has both WILL_FLOODING = WILL_DEFAULT and WILL_ROUTING = WILL_DEFAULT need not include an MPR_WILLING TLV in its HELLO messages.

The following constraints apply to these parameters:

- WILL_FLOODING >= WILL_NEVER

- WILL_FLOODING <= WILL_ALWAYS

- WILL_ROUTING >= WILL_NEVER

- WILL_ROUTING <= WILL_ALWAYS

5.5. Parameter Change Constraints

If protocol parameters are changed dynamically, then the constraints in this section apply.

RX_HOLD_TIME

- If RX_HOLD_TIME for an OLSRv2 interface changes, then the expiry time for all Received Tuples for that OLSRv2 interface MAY be changed.

O_HOLD_TIME

* If O_HOLD_TIME for a router changes, then the expiry time for all Originator Tuples MAY be changed.

TC_INTERVAL

* If the TC_INTERVAL for a router increases, then the next TC message generated by this router MUST be generated according to the previous, shorter, TC_INTERVAL. Additional subsequent TC messages MAY be generated according to the previous, shorter, TC_INTERVAL.

* If the TC_INTERVAL for a router decreases, then the following TC messages from this router MUST be generated according to the current, shorter, TC_INTERVAL.

P_HOLD_TIME

* If P_HOLD_TIME changes, then the expiry time for all Processed Tuples MAY be changed.

F_HOLD_TIME

* If F_HOLD_TIME changes, then the expiry time for all Forwarded Tuples MAY be changed.

TP_MAXJITTER

* If TP_MAXJITTER changes, then the periodic TC message schedule on this router MAY be changed immediately.

TT_MAXJITTER

* If TT_MAXJITTER changes, then externally triggered TC messages on this router MAY be rescheduled.

F_MAXJITTER

* If F_MAXJITTER changes, then TC messages waiting to be forwarded with a delay based on this parameter MAY be rescheduled.

TC_HOP_LIMIT

* If TC_HOP_LIMIT changes, and the router uses multiple values after the change, then message intervals and validity times included in TC messages MUST be respected. The simplest way to

do this is to start any new repeating pattern of TC_HOP_LIMIT values with its largest value.

LINK_METRIC_TYPE

- If LINK_METRIC_TYPE changes then all link metric information recorded by the router is invalid. The router MUST take the following actions, and all consequent actions described in Section 17 and [RFC6130].

  - For each Link Tuple in any Link Set, either update L_in_metric (the value MAXIMUM_METRIC MAY be used) or remove the Link Tuple from the Link Set.

  - For each Link Tuple that is not removed, set:

    - L_out_metric := UNKNOWN_METRIC;

    - L_SYM_time := expired;

    - L_MPR_selector := false.

  - Remove all Router Topology Tuples, Routable Address Topology Tuples, Attached Network Tuples and Routing Tuples from their respective protocol sets in the Topology Information Base.

5.6. Constants

5.6.1. Link Metric Constants

The constant minimum, maximum and default metric values are defined by:

- MINIMUM_METRIC := 1

- MAXIMUM_METRIC := 16776960

- DEFAULT_METRIC := 256

The symbolic value UNKNOWN_METRIC is defined in Section 6.1.

5.6.2. Willingness Constants

The constant minimum, maximum and default willingness values are defined by:

- o WILL_NEVER := 0

- o WILL_ALWAYS := 15

- o WILL_DEFAULT := 7

6. Link Metric Values

A router records a link metric value for each direction of a link of which it has knowledge. These link metric values are used to create metrics for routes by the addition of link metric values.

6.1. Link Metric Representation

Link metrics are reported in messages using a compressed representation that occupies 12 bits, a 4 bit field and an 8 bit field. The compressed representation represents positive integer values with a minimum value of 1 and a maximum value that is slightly smaller than the maximum 24 bit value. Only those values that have exact representation in the compressed form are used. Route metrics are the summation of no more then 255 link metric values, and can therefore be represented using no more than 32 bits.

Link and route metrics used in the Information Bases defined in this specification refer to the uncompressed values, and arithmetic involving them does likewise, and assumes full precision in the result. (How an implementation records the values is not part of this specification, as long as it behaves as if recording uncompressed values. An implementation can, for example, use 32 bit values for all link and route metrics.)

In some cases a link metric value may be unknown. This is indicated in this specification by the value UNKNOWN_METRIC. An implementation may use any representation of UNKNOWN_METRIC as it is never included in messages or used in any computation. (Possible values are zero, or any value greater than the maximum representable metric value.)

6.2. Link Metric Compressed Form

The 12-bit compressed form of a link metric uses a modified form of a representation with as 8-bit mantissa (denoted b) and a 4-bit exponent (denoted a). Note that if represented as the 12 bit value 256a+b then the ordering of those 12 bit values is identical to the ordering of the represented values.

The value so represented is (257+b)2^a - 256, where ^ denotes exponentiation. This has a minimum value (when a = 0 and b = 0) of MINIMUM_METRIC = 1 and a maximum value (when a = 15 and b = 255) of

MAXIMUM_METRIC = 2^24 - 256.

An algorithm for computing a and b for the smallest representable value not less than a link metric value v such that MINIMUM_METRIC <= v <= MAXIMUM_METRIC is:

1. Find the smallest integer a such that v + 256 <= 2^(a + 9).

2. Set b := (v - 256(2^a - 1)) / (2^a) - 1, rounded up to the nearest integer.

To allow for more efficient messages, a default link metric DEFAULT_METRIC is defined, which can be omitted from messages. Note that this is not the same as the link metric value that should be used when this specification requires a link metric, but no information about a link, beyond that a HELLO message has been received using that link, is available. In this case the link metric used SHOULD be MAXIMUM_METRIC.

7. Local Information Base

The Local Information Base, as defined for each router in [RFC6130], is extended by this protocol by:

- Recording the router's originator address. The originator address MUST be unique to this router. It MUST NOT be used by any other router as an originator address. It MAY be included in any network address in any I_local_iface_addr_list of this router, it MUST NOT be included in any network address in any I_local_iface_addr_list of any other router. It MAY be included in, but MUST NOT be equal to, the AL_net_addr in any Local Attached Network Tuple in this or any other router.

- The addition of an Originator Set, defined in Section 7.1, and a Local Attached Network Set, defined in Section 7.2.

All routable addresses of the router for which it is to accept packets as destination MUST be included in the Local Interface Set or the Local Attached Network Set.

7.1. Originator Set

A router's Originator Set records addresses that were recently used as originator addresses by this router. If a router's originator address is immutable then this set is always empty and MAY be omitted. It consists of Originator Tuples:

(O_orig_addr, O_time)

where:

O_orig_addr is a recently used originator address; note that this does not include a prefix length;

O_time specifies the time at which this Tuple expires and MUST be removed.

7.2. Local Attached Network Set

A router's Local Attached Network Set records its local non-OLSRv2 interfaces via which it can act as gateways to other networks. The Local Attached Network Set is not modified by this protocol. This protocol MAY respond to changes to the Local Attached Network Set, which MUST reflect corresponding changes in the router's status. It consists of Local Attached Network Tuples:

(AL_net_addr, AL_dist, AL_metric)

where:

AL_net_addr is the network address of an attached network which can be reached via this router. This SHOULD be a routable address. It is constrained as described below.

AL_dist is the number of hops to the network with network address AL_net_addr from this router.

AL_metric is the metric of the link to the attached network with address AL_net_addr from this router;

Attached networks local to this router only (i.e., not reachable except via this router) SHOULD be treated as local non-MANET interfaces, and added to the Local Interface Set, as specified in [RFC6130], rather than be added to the Local Attached Network Set.

Because an attached network is not specific to the router, and may be outside the MANET, an attached network MAY also be attached to other routers. Routing to an AL_net_addr will use maximum prefix length matching; consequently an AL_net_addr MAY include, but MUST NOT equal or be included in, any network address which is of any interface of any router (i.e., is included in any I_local_iface_addr_list) or equal any router's originator address.

It is not the responsibility of this protocol to maintain routes from this router to networks recorded in the Local Attached Network Set.

Local Attached Neighbor Tuples are removed from the Local Attached

Network Set only when the routers' local attached network configuration changes, i.e., they are not subject to timer-based expiration or changes due to received messages.

8. Interface Information Base

An Interface Information Base, as defined in [RFC6130], is maintained for each OLSRv2 interface. Its Link Set and 2-Hop Set are modified by this protocol.

8.1. Link Set

The Link Set is modified by adding these additional elements to each Link Tuple:

L_in_metric is the metric of the link from the OLSRv2 interface with addresses L_neighbor_iface_addr_list to this OLSRv2 interface;

L_out_metric is the metric of the link to the OLSRv2 interface with addresses L_neighbor_iface_addr_list from this OLSRv2 interface;

L_mpr_selector is a boolean flag, describing if this neighbor has selected this router as a flooding MPR, i.e., is a flooding MPR selector of this router.

L_in_metric will be specified by a process that is external to this specification. Any Link Tuple with L_status = HEARD or L_status = SYMMETRIC MUST have a specified value of L_in_metric.

A Link Tuple created (but not updated) by [RFC6130] MUST set:

o L_in_metric := UNKNOWN_METRIC;

o L_out_metric := UNKNOWN_METRIC;

o L_mpr_selector := false.

8.2. 2-Hop Set

The 2-Hop Set is modified by adding these additional elements to each 2-Hop Tuple:

N2_in_metric is the neighbor metric from the router with address N2_2hop_iface_addr to the router with OLSRv2 interface addresses N2_neighbor_iface_addr_list;

N2_out_metric is the neighbor metric to the router with address N2_2hop_iface_addr from the router with OLSRv2 interface addresses N2_neighbor_iface_addr_list.

A 2-Hop Tuple created (but not updated) by [RFC6130] MUST set:

- N2_in_metric := UNKNOWN_METRIC;
- N2_out_metric := UNKNOWN_METRIC.

9. Neighbor Information Base

An Neighbor Information Base, as defined in [RFC6130], is maintained for each router. It is modified by this protocol by adding these additional elements to each Neighbor Tuple in the Neighbor Set:

N_orig_addr is the neighbor's originator address, which may be unknown. Note that this originator address does not include a prefix length;

N_in_metric is the neighbor metric of any link from this neighbor to this router, i.e., the minimum of all corresponding L_in_metric with L_status = SYMMETRIC, UNKNOWN_METRIC if there are no such Link Tuples;

N_out_metric is the neighbor metric of any link from this router to this neighbor, i.e., the minimum of all corresponding L_out_metric with L_status = SYMMETRIC, UNKNOWN_METRIC if there are no such Link Tuples;

N_will_flooding is the neighbor's willingness to be selected as a flooding MPR, in the range from WILL_NEVER to WILL_ALWAYS, both inclusive;

N_will_routing is the neighbor's willingness to be selected as a routing MPR, in the range from WILL_NEVER to WILL_ALWAYS, both inclusive;

N_flooding_mpr is a boolean flag, describing if this neighbor is selected as a flooding MPR by this router;

N_routing_mpr is a boolean flag, describing if this neighbor is selected as a routing MPR by this router;

N_mpr_selector is a boolean flag, describing if this neighbor has selected this router as a routing MPR, i.e., is a routing MPR selector of this router.

N_advertised is a boolean flag, describing if this router has elected to advertise a link to this neighbor in its TC messages.

A Neighbor Tuple created (but not updated) by [RFC6130] MUST set:

- N_orig_addr := unknown;

- N_in_metric := UNKNOWN_METRIC;

- N_out_metric := UNKNOWN_METRIC;

- N_will_flooding := WILL_NEVER;

- N_will_routing := WILL_NEVER;

- N_routing_mpr := false;

- N_flooding_mpr := false;

- N_mpr_selector := false;

- N_advertised := false.

The Neighbor Information Base also includes a variable, the Advertised Neighbor Sequence Number (ANSN), whose value is included in TC messages to indicate the freshness of the information transmitted. The ANSN is incremented whenever advertised information (the originator and routable addresses included in Neighbor Tuples with N_advertised = true, and local attached networks recorded in the Local Attached Network Set in the Local Information Base) changes, including addition or removal of such information.

10. Topology Information Base

The Topology Information Base, defined for each router by this specification, stores information received in TC messages, in the Advertising Remote Router Set, the Router Topology Set, the Routable Address Topology Set and the Attached Network Set.

Additionally, a Routing Set is maintained, derived from the information recorded in the Local Information Base, the Interface Information Bases, the Neighbor Information Base and the rest of the Topology Information Base.

10.1. Advertising Remote Router Set

A router's Advertising Remote Router Set records information describing each remote router in the network that transmits TC messages, allowing outdated TC messages to be recognized and discarded. It consists of Advertising Remote Router Tuples:

(AR_orig_addr, AR_seq_number, AR_time)

where:

AR_orig_addr is the originator address of a received TC message, note that this does not include a prefix length;

AR_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address AR_orig_addr (i.e., which contributed to the information contained in this Tuple);

AR_time is the time at which this Tuple expires and MUST be removed.

10.2. Router Topology Set

A router's Topology Set records topology information about the links between routers in the MANET. It consists of Router Topology Tuples:

(TR_from_orig_addr, TR_to_orig_addr, TR_seq_number, TR_metric, TR_time)

where:

TR_from_orig_addr is the originator address of a router which can reach the router with originator address TR_to_orig_addr in one hop, note that this does not include a prefix length;

TR_to_orig_addr is the originator address of a router which can be reached by the router with originator address TR_to_orig_addr in one hop, note that this does not include a prefix length;

TR_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address TR_from_orig_addr (i.e., which contributed to the information contained in this Tuple);

TR_metric is the neighbor metric from the router with originator address TR_from_orig_addr to the router with originator address TR_to_orig_addr;

TR_time specifies the time at which this Tuple expires and MUST be removed.

10.3. Routable Address Topology Set

A router's Routable Address Topology Set records topology information about the routable addresses within the MANET, and via which routers they may be reached. It consists of Routable Address Topology Tuples:

```
(TA_from_orig_addr, TA_dest_addr, TA_seq_number, TA_metric,
 TA_time)
```

where:

TA_from_orig_addr is the originator address of a router which can reach the router with routable address TA_dest_addr in one hop, note that this does not include a prefix length;

TA_dest_addr is a routable address of a router which can be reached by the router with originator address TA_from_orig_addr in one hop;

TA_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address TA_from_orig_addr (i.e., which contributed to the information contained in this Tuple);

TA_metric is the neighbor metric from the router with originator address TA_from_orig_addr to the router with OLSRv2 interface address TA_dest_addr;

TA_time specifies the time at which this Tuple expires and MUST be removed.

10.4. Attached Network Set

A router's Attached Network Set records information about networks (which may be outside the MANET) attached to other routers and their routable addresses. It consists of Attached Network Tuples:

```
(AN_orig_addr, AN_net_addr, AN_seq_number, AN_dist, AN_metric,
 AN_time)
```

where:

AN_orig_addr is the originator address of a router which can act as gateway to the network with network address AN_net_addr, note that this does not include a prefix length;

AN_net_addr is the network address of an attached network, which may be reached via the router with originator address AN_orig_addr;

AN_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address AN_orig_addr (i.e., which contributed to the information contained in this Tuple);

AN_dist is the number of hops to the network with network address AN_net_addr from the router with originator address AN_orig_addr;

AN_metric is the metric of the link from the router with originator address AN_orig_addr to the attached network with address AN_net_addr;

AN_time specifies the time at which this Tuple expires and MUST be removed.

10.5. Routing Set

A router's Routing Set records the first hop along a selected path to each destination for which any such path is known. It consists of Routing Tuples:

```
(R_dest_addr, R_next_iface_addr, R_local_iface_addr, R_dist,
 R_metric)
```

where:

R_dest_addr is the network address of the destination, either the network address of an interface of a destination router, or the network address of an attached network;

R_next_iface_addr is the network address of the "next hop" on the selected path to the destination;

R_local_iface_addr is the network address of the local OLSRv2 interface over which a packet MUST be sent to reach the destination by the selected path.

R_dist is the number of hops on the selected path to the destination;

R_metric is the metric of the route to the destination with address R_dest_addr.

The Routing Set for a router is derived from the contents of other protocol sets of the router (the Link Sets, the Neighbor Set, the Router Topology Set, the Routable Address Topology Set, the Attached Network Set, and OPTIONALLY the 2-Hop Sets). The Routing Set is updated (Routing Tuples added or removed, or the complete Routing Set recalculated) when routing paths are calculated, based on changes to these other protocol sets. Routing Tuples are not subject to timer-based expiration.

11. Received Message Information Base

The Received Message Information Base, defined by this specification, records information required to ensure that a message is processed at most once and is forwarded at most once per OLSRv2 interface of a router, using MPR flooding.

11.1. Received Set

A router has a Received Set per OLSRv2 interface. Each Received Set records the signatures of messages which have been received over that OLSRv2 interface. Each consists of Received Tuples:

(RX_type, RX_orig_addr, RX_seq_number, RX_time)

where:

RX_type is the received Message Type;

RX_orig_addr is the originator address of the received message, note that this does not include a prefix length;

RX_seq_number is the message sequence number of the received message;

RX_time specifies the time at which this Tuple expires and MUST be removed.

11.2. Processed Set

A router has a single Processed Set which records signatures of messages which have been processed by the router. It consists of Processed Tuples:

(P_type, P_orig_addr, P_seq_number, P_time)

where:

P_type is the processed Message Type;

P_orig_addr is the originator address of the processed message, note that this does not include a prefix length;

P_seq_number is the message sequence number of the processed message;

P_time specifies the time at which this Tuple expires and MUST be removed.

11.3. Forwarded Set

A router has a single Forwarded Set which records signatures of messages which have been forwarded by the router. It consists of Forwarded Tuples:

(F_type, F_orig_addr, F_seq_number, F_time)

where:

F_type is the forwarded Message Type;

F_orig_addr is the originator address of the forwarded message, note that this does not include a prefix length;

F_seq_number is the message sequence number of the forwarded message;

F_time specifies the time at which this Tuple expires and MUST be removed.

12. Information Base Properties

As part of this specification, in a number of cases there is a natural correspondence from a Protocol Tuple in one Protocol Set to a single Protocol Tuple in another Protocol Set, in the same or another Information Base. The latter Protocol Tuple is referred to as "corresponding" to the former Protocol Tuple.

Specific examples of corresponding Protocol Tuples include:

- There is a Local Interface Tuple corresponding to each Link Tuple, where the Link Tuple is in the Link Set for an OLSRv2 interface, and the Local Interface Tuple represents that OLSRv2 interface.

- There is a Neighbor Tuple corresponding to each Link Tuple which has L_HEARD_time not expired, such that N_neighbor_addr_list contains L_neighbor_iface_addr_list.

- There is a Link Tuple (in the Link Set in the same Interface Information Base) corresponding to each 2-Hop Tuple such that L_neighbor_iface_addr_list = N2_neighbor_iface_addr_list.

- There is a Neighbor Tuple corresponding to each 2-Hop Tuple, such that N_neighbor_addr_list contains N2_neighbor_iface_addr_list. (This is the Neighbor Tuple corresponding to the Link Tuple that corresponds to the 2-Hop Tuple.)

- There is an Advertising Remote Router Tuple corresponding to each Router Topology Tuple such that AR_orig_addr = TR_from_orig_addr.

- There is an Advertising Remote Router Tuple corresponding to each Routable Address Topology Tuple such that AR_orig_addr = TA_from_orig_addr.

- There is an Advertising Remote Router Tuple corresponding to each Attached Network Tuple such that AR_orig_addr = AN_orig_addr.

- There is an Neighbor Tuple corresponding to each Routing Tuple such that N_neighbor_addr_list contains R_next_iface_addr.

Addresses or network addresses with the following properties are considered as "fully owned" by a router when processing a received message:

- Equaling its originator address, OR;

- Equaling the O_orig_addr in an Originator Tuple, OR;

- Equaling or being a sub-range of the I_local_iface_addr_list in a Local Interface Tuple, OR;

- Equaling or being a sub-range of the IR_local_iface_addr in a Removed Interface Address Tuple, OR;

- Equaling an AL_net_addr in a Local Attached Network Tuple.

Addresses or network addresses with the following properties are considered as "partially owned" (which may include being fully owned) by a router when processing a received message:

- Overlapping (equaling or containing) its originator address, OR;

- Overlapping (equaling or containing) the O_orig_addr in an Originator Tuple, OR;

- o Overlapping the I_local_iface_addr_list in a Local Interface Tuple, OR;

- o Overlapping the IR_local_iface_addr in a Removed Interface Address Tuple, OR;

- o Equaling or having as a sub-range an AL_net_addr in a Local Attached Network Tuple.

13. Packets and Messages

The packet and message format used by this protocol is defined in [RFC5444]. Except as otherwise noted, options defined in [RFC5444] may be freely used, in particular alternative formats defined by packet, message, Address Block and TLV flags.

This section describes the usage of the packets and messages defined in [RFC5444] by this specification, and the TLVs defined by, and used in, this specification.

13.1. Messages

Routers using this protocol exchange information through messages. The message types used by this protocol are the HELLO message and the TC message. The HELLO message is defined by [RFC6130] and extended by this specification, see Section 15. The TC message is defined by this specification, see Section 16.

13.2. Packets

One or more messages sent by a router at the same time SHOULD be combined into a single packet, subject to any constraints on maximum packet size (such as derived from the MTU over a local single hop) that MAY be imposed. These messages may have originated at the sending router, or have originated at another router and are being forwarded by the sending router. Messages with different originating routers MAY be combined for transmission within the same packet. Messages from other protocols defined using [RFC5444], including but not limited to [RFC6130], MAY be combined for transmission within the same packet. This specification does not define or use any contents of the Packet Header.

Forwarded messages MAY be jittered as described in [RFC5148], including the observation that the forwarding jitter of all messages received in a single packet SHOULD be the same. The value of MAXJITTER used in jittering a forwarded message MAY be based on information in that message (in particular any Message TLVs with Type = INTERVAL_TIME or Type = VALIDITY_TIME) or otherwise SHOULD be with a maximum delay of F_MAXJITTER. A router MAY modify the jitter applied to a message in order to more efficiently combine messages in packets, as long as the maximum jitter is not exceeded.

13.3. TLVs

This specification defines 2 Message TLVs and 4 Address Block TLVs.

All references in this specification to TLVs that do not indicate a type extension, assume Type Extension = 0. TLVs in processed messages with a type extension which is neither zero as so assumed, nor a specifically indicated non-zero type extension, are ignored.

13.3.1. Message TLVs

The MPR_WILLING TLV is used in HELLO messages. A message MUST NOT contain more than one MPR_WILLING TLV.

Type	Value Length	Value
MPR_WILLING	1 octet	Bits 0-3 encode the parameter WILL_FLOODING; bits 4-7 encode the parameter WILL_ROUTING.

Table 1: MPR_WILLING TLV definition

The CONT_SEQ_NUM TLV is used in TC messages. message MUST NOT contain more than one CONT_SEQ_NUM TLV.

Type	Value Length	Value
CONT_SEQ_NUM	2 octets	The ANSN contained in the Neighbor Information Base.

Table 2: CONT_SEQ_NUM TLV definition

13.3.2. Address Block TLVs

The LINK_METRIC TLV is used in HELLO messages and TC messages. It MAY use any type extension; only LINK_METRIC TLVs with type extension equal to LINK_METRIC_TYPE will be used by this specification. At most one link metric value of any given kind (link or neighbor) and direction may be associated with any address.

Type	Value Length	Value
LINK_METRIC	2 octets	Bits 0-3 indicates kind(s) and direction(s), Bits 4-7 indicate exponent (a), Bits 8-15 indicate mantissa (b)

Table 3: LINK_METRIC TLV definition

The exponent and mantissa use the representation defined in Section 6. Each bit of the types and directions sub-field, if set ('1') indicates that the link metric is of the indicated kind and direction. Any combination of these bits MAY be used.

Bit	Kind	Direction
0	Link metric	Incoming
1	Link metric	Outgoing
2	Neighbor metric	Incoming
3	Neighbor metric	Outgoing

Table 4: LINK_METRIC TLV types and directions

The MPR TLV is used in HELLO messages, and indicates that an address with which it is associated is of a symmetric 1-hop neighbor that has been selected as an MPR.

Type	Value Length	Value
MPR	1 octet	FLOODING indicates that the corresponding address is of a neighbor selected as a flooding MPR, ROUTING indicates that the corresponding address is of a neighbor selected as a routing MPR, FLOOD_ROUTE indicates both

Table 5: MPR TLV definition

The NBR_ADDR_TYPE TLV is used in TC messages.

Type	Value Length	Value
NBR_ADDR_TYPE	1 octet	ORIGINATOR indicates that the corresponding address (which MUST have maximum prefix length) is an originator address, ROUTABLE indicates that the corresponding network address is a routable address of an interface, ROUTABLE_ORIG indicates that the corresponding address is both

Table 6: NBR_ADDR_TYPE TLV definition

If an address is both an originator address and a routable address, then it may be associated with either one Address Block TLV with Type := NBR_ADDR_TYPE and Value := ROUTABLE_ORIG, or with two Address Block TLVs with Type:= NBR_ADDR_TYPE, one with Value := ORIGINATOR and one with Value := ROUTABLE.

The GATEWAY TLV is used in TC messages. At most one GATEWAY TLV may be associated with any address.

Type	Value Length	Value
GATEWAY	1 octet	Number of hops to attached network.

Table 7: GATEWAY TLV definition

All address objects included in a TC message according to this specification MUST be associated either with at least one TLV with Type := NBR_ADDR_TYPE or with a TLV with Type := GATEWAY, but not both. Any other address objects MAY be included in Address Blocks in a TC message, but are ignored by this specification.

14. Message Processing and Forwarding

This section describes the optimized flooding operation (MPR flooding) used when control messages, as instances of [RFC5444], are intended for MANET wide distribution. This flooding mechanism defines when a received message is, or is not, processed and/or forwarded.

This flooding mechanism is used by this protocol and MAY be used by extensions to this protocol which define, and hence own, other message types, to manage processing and/or forwarding of these messages. This specification contains elements (P_type, RX_type, F_type) required only for such usage.

This flooding mechanism is always used for TC messages (see Section 16). Received HELLO messages (see Section 15 are, unless invalid, always processed, and never forwarded by this flooding mechanism. They thus do not need to be recorded in the Received Message Information Base.

The processing selection and forwarding mechanisms are designed to only need to parse the Message Header in order to determine whether a message is to be processed and/or forwarded, and not to have to parse the Message Body even if the message is forwarded (but not processed). An implementation MAY only parse the Message Body if necessary, or MAY always parse the Message Body and reject the message if it cannot be so parsed, or any other error is identified.

An implementation MUST discard the message silently if it is unable to parse the Message Header or (if attempted) the Message Body, or if a message (other than a HELLO message) does not include a message sequence number.

14.1. Actions when Receiving a Message

On receiving a message of a type specified to be using this mechanism, which includes the TC messages defined in this specification, a router MUST perform the following:

1. If the router recognizes from the originator address of the message that the message is one which the receiving router itself originated (i.e., the message originator address is the originator address of this router, or is an O_orig_addr in an Originator Tuple) then the message MUST be silently discarded.

2. Otherwise:

    1. If the message is of a type which may be processed, then the message is considered for processing according to Section 14.2.

    2. If the message is of a type which may be forwarded, AND:

        + <msg-hop-limit> is present and <msg-hop-limit> > 1, AND;

        + <msg-hop-count> is not present or <msg-hop-count> < 255;

then the message is considered for forwarding according to Section 14.3.

14.2. Message Considered for Processing

If a message (the "current message") is considered for processing, then the following tasks MUST be performed:

1. If the sending address (i.e., the source address of the IP datagram containing the current message) does not match (taking into account any address prefix) a network address in an L_neighbor_iface_addr_list of a Link Tuple, with L_status = SYMMETRIC, in the Link Set for the OLSRv2 interface on which the current message was received (the "receiving interface") then processing the current message is OPTIONAL. If the current message is not processed then the following steps are not carried out.

2. If a Processed Tuple exists with:

   * P_type = the Message Type of the current message, AND;

   * P_orig_addr = the originator address of the current message, AND;

   * P_seq_number = the message sequence number of the current message;

   then the current message MUST NOT be processed.

3. Otherwise:

   1. Create a Processed Tuple with:

      + P_type := the Message Type of the current message;

      + P_orig_addr := the originator address of the current message;

      + P_seq_number := the sequence number of the current message;

      + P_time := current time + P_HOLD_TIME.

   2. Process the current message according to its Message Type. For a TC message this is as defined in Section 16.3.

14.3. Message Considered for Forwarding

If a message (the "current message") is considered for forwarding, then the following tasks MUST be performed:

1. If the sending address (i.e., the source address of the IP datagram containing the current message) does not match (taking into account any address prefix) a network address in an L_neighbor_iface_addr_list of a Link Tuple, with L_status = SYMMETRIC, in the Link Set for the OLSRv2 interface on which the current message was received (the "receiving interface") then the current message MUST be silently discarded.

2. Otherwise:

    1. If a Received Tuple exists in the Received Set for the receiving interface, with:

        + RX_type = the Message Type of the current message, AND;

        + RX_orig_addr = the originator address of the current message, AND;

        + RX_seq_number = the sequence number of the current message;

        then the current message MUST be silently discarded.

    2. Otherwise:

        1. Create a Received Tuple in the Received Set for the receiving interface with:

            - RX_type := the Message Type of the current message;

            - RX_orig_addr := originator address of the current message;

            - RX_seq_number := sequence number of the current message;

            - RX_time := current time + RX_HOLD_TIME.

        2. If a Forwarded Tuple exists with:

            - F_type = the Message Type of the current message, AND;

- F_orig_addr = the originator address of the current message, AND;

- F_seq_number = the sequence number of the current message.

then the current message MUST be silently discarded.

3. Otherwise if the sending address matches (taking account of any address prefix) any network address in an L_neighbor_iface_addr_list of a Link Tuple in the Link Set for the receiving OLSRv2 interface that has L_status = SYMMETRIC and whose corresponding Neighbor Tuple has N_mpr_selector = true, then:

    1. Create a Forwarded Tuple with:

        - o F_type := the Message Type of the current message;

        - o F_orig_addr := originator address of the current message;

        - o F_seq_number := sequence number of the current message;

        - o F_time := current time + F_HOLD_TIME.

    2. The Message Header of the current message is modified by:

        - o if present, decrement <msg-hop-limit> in the Message Header by 1, AND;

        - o if present, increment <msg-hop-count> in the Message Header by 1.

    3. The message is transmitted over all OLSRv2 interfaces, as described in Section 13.

15. HELLO Messages

The HELLO message Message Type is owned by [RFC6130], and thus HELLO messages are generated, transmitted, received and processed by [RFC6130]. This protocol, as permitted by [RFC6130], also uses HELLO messages, including adding to HELLO message generation, and implementing additional processing based on received HELLO messages. HELLO messages are not forwarded by [RFC6130] or by this specification.

15.1. HELLO Message Generation

A HELLO message is generated as defined in [RFC6130], extended by the following elements being added to the HELLO message by this specification before the HELLO message is sent over an OLSRv2 interface:

- A message originator address, recording this router's originator address. This MUST use a <msg-orig-addr> element, unless:

  * The message specifies only a single local interface address (i.e., contains only one address object that is associated with an Address Block TLV with Type = LOCAL_IF, and which has no prefix length, or a maximum prefix length) which will then be interpreted as the message originator address, OR;

  * The message does not include any local interface network addresses (i.e., has no address objects associated with an Address Block TLV with Type = LOCAL_IF), as permitted by the specification in [RFC6130] when the router that generated the HELLO message has only one interface address and will use that as the sending address of the IP datagram in which the HELLO message is contained. In this case that address will be interpreted as the message originator address.

- A Message TLV with Type := MPR_WILLING MUST be included, unless both willingness values that it reports are equal to WILL_DEFAULT (in which case it MAY be included).

- The following cases associate Address Block TLVs with one or more addresses from a Link Tuple or a Neighbor Tuple if these are included in the HELLO message. In each case the TLV MUST be associated with at least copy of one address from the relevant Tuple; the TLV MAY be associated with more such addresses (including a copy of that address object, possibly not itself associated with any other indicated TLVs, in the same or a different Address Block). These additional TLVs MUST NOT be associated with any other addresses in a HELLO message that will be processed by [RFC6130].

  * For each Link Tuple for which L_in_metric != UNKNOWN_METRIC, and for which one or more addresses in its L_neighbor_iface_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS and Value = HEARD or Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an incoming link metric with value L_in_metric, unless this equals DEFAULT_METRIC.

* For each Link Tuple for which L_out_metric != UNKNOWN_METRIC, and for which one or more addresses in its L_neighbor_iface_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an outgoing link metric with value L_out_metric, unless this equals DEFAULT_METRIC.

* For each Neighbor Tuple for which N_symmetric = true, and for which one or more addresses in its N_neighbor_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an incoming neighbor metric with value N_in_metric, unless this equals DEFAULT_METRIC.

* For each Neighbor Tuple for which N_symmetric = true, and for which one or more addresses in its N_neighbor_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an outgoing neighbor metric with value N_out_metric, unless this equals DEFAULT_METRIC.

* For each Neighbor Tuple with N_flooding_mpr = true, and for which one or more network addresses in its N_neighbor_addr_list are included as address objects in the HELLO message with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := MPR and Value := FLOODING or Value := FLOOD_ROUTE.

* For each Neighbor Tuple with N_routing_mpr = true, and for which one or more network addresses in its N_neighbor_addr_list are included as address objects in the HELLO message with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := MPR and Value := ROUTING or Value := FLOOD_ROUTE.

15.2. HELLO Message Transmission

HELLO messages are scheduled and transmitted by [RFC6130]. This protocol MAY require that an additional HELLO message is sent when either of the router’s sets of MPRs changes, in addition to the cases specified in [RFC6130], and subject to the same constraints.

15.3. HELLO Message Processing

When received on an OLSRv2 interface, HELLO messages are made available to this protocol in two ways, both as permitted by [RFC6130]:

- Such received HELLO messages MUST be made available to this protocol on reception, which allows them to be discarded before being processed by [RFC6130], for example if the information added to the HELLO message by this specification is inconsistent.

- Such received HELLO messages MUST be made available to OLSRv2 after [RFC6130] has completed its processing thereof, unless discarded as malformed by [RFC6130], for processing by this specification.

15.3.1. HELLO Message Discarding

In addition to the reasons specified in [RFC6130] for discarding a HELLO message on reception, a HELLO message MUST be discarded before processing by [RFC6130] or this specification if it:

- Has more than one Message TLV with Type = MPR_WILLING.

- Has a message originator address, or a network address corresponding to an address object associated with an Address Block TLV with Type = LOCAL_IF, that is partially owned by this router. (Some of these cases are already excluded by [RFC6130].)

- Includes any address object associated with an Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB that overlaps the message’s originator address.

- Contains any address that will be processed by [RFC6130] that is associated, using the same or different address objects, with two different values of link metric with the same kind and direction using a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE. This also applies to different addresses that are both of the OLSRv2 interface on which the HELLO message was received.

- Contains any address object associated with an Address Block TLV with Type = MPR that is not also associated with an Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC (including using a different copy of that address object, in the same or a different Address Block).

15.3.2. HELLO Message Usage

HELLO messages are first processed as specified in [RFC6130]. That processing includes identifying (or creating) a Link Tuple and a Neighbor Tuple corresponding to the originator of the HELLO message (the "current Link Tuple" and the "current Neighbor Tuple"). After this, the following processing MUST also be performed:

1. If the HELLO message has a well-defined message originator address, i.e., has an <msg-orig-addr> element or has zero or one network addresses associated with a TLV with Type = LOCAL_IF:

    1. Remove any Neighbor Tuple, other than the current Neighbor Tuple, with N_orig_addr = message originator address, taking any consequent action (including removing one or more Link Tuples) as specified in [RFC6130].

    2. The current Link Tuple is then updated according to:

        1. Update L_in_metric and L_out_metric as described in Section 15.3.2.1;

        2. Update L_mpr_selector as described in Section 15.3.2.3.

    3. The current Neighbor Tuple is then updated according to:

        1. N_orig_addr := message originator address;

        2. Update N_in_metric and N_out_metric as described in Section 15.3.2.1;

        3. Update N_will_flooding and N_will_routing as described in Section 15.3.2.2;

        4. Update N_mpr_selector as described in Section 15.3.2.3.

2. If there are any changes to the router's Information Bases, then perform the processing defined in Section 17.

15.3.2.1. Updating Metrics

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS and Value = HEARD or Value = SYMMETRIC, an incoming (to the message originator) link metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (link) and direction (incoming) of metric, or as the value DEFAULT_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS and Value = SYMMETRIC, an outgoing (to the message originator) link metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (link) and direction (outgoing) of metric, or as the value DEFAULT_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, an incoming (to the message originator) neighbor metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (neighbor) and direction (incoming) of metric, or as the value DEFAULT_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, an outgoing (to the message originator) neighbor metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (neighbor) and direction (outgoing) of metric, or as the value DEFAULT_METRIC.

The link metric elements L_in_metric and L_out_metric in a Link Tuple are updated according to the following:

o For any Link Tuple, L_in_metric MAY be set to any representable value, by a process outside this specification, at any time. L_in_metric MUST be so set whenever L_status becomes equal to HEARD or SYMMETRIC (if no other value is available then the value MAXIMUM_METRIC SHOULD be used). This MAY use information based on the receipt of a packet including a HELLO message that causes the creation or updating of that Link Tuple.

o When, as specified in [RFC6130], a Link Tuple is updated (possibly immediately after being created) due to the receipt of a HELLO message, if L_status = SYMMETRIC, then L_out_metric is set equal to the incoming link metric for any included address of the interface on which the HELLO message was received, ignoring any values equal to DEFAULT_METRIC unless there are only such values. (Note that the rules for discarding HELLO messages in Section 15.3.1 make this value unambiguous.)

The neighbor metric elements N_in_metric and N_out_metric in a Neighbor Tuple are updated according to Section 17.3.

The metric elements N2_in_metric and N2_out_metric in any 2-Hop Tuple updated as defined in [RFC6130] are updated to equal the incoming neighbor metric and outgoing neighbor metric, respectively, associated with the corresponding N2_2hop_addr.

15.3.2.2. Updating Willingness

N_will_flooding and N_will_routing in the current Neighbor Tuple are updated as follows:

1. If the HELLO message contains a Message TLV with Type = MPR_WILLING then N_will_flooding := bits 0-3 of the value of that TLV, and N_will_routing := bits 4-7 of the value of that TLV (each in the range 0 to 15).

2. Otherwise, N_will_flooding := WILL_DEFAULT, and N_will_routing := WILL_DEFAULT.

15.3.2.3. Updating MPR Selector Status

L_mpr_selector is updated as follows:

1. If a router finds an address object representing any of its local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = MPR and Value = FLOODING or Value = FLOOD_ROUTE in the HELLO message (indicating that the originating router has selected the receiving router as a flooding MPR) then, for the current Link Tuple:

   * L_mpr_selector := true.

2. Otherwise (i.e., if no such address object and Address Block TLV was found) if a router finds an address object representing any of its local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC in the HELLO message, then for the current Link Tuple:

   * L_mpr_selector := false.

N_mpr_selector is updated as follows:

1. If a router finds an address object representing any of its local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = MPR and Value = ROUTING or Value = FLOOD_ROUTE in the HELLO message (indicating that the originating router has selected the receiving router as a routing MPR) then, for the current Neighbor Tuple:

- N_mpr_selector := true;

- N_advertised := true.

2. Otherwise (i.e., if no such address object and Address Block TLV was found) if a router finds an address object representing any of its local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC in the HELLO message, then for the current Neighbor Tuple:

   * N_mpr_selector := false;

   * The router MAY also set N_advertised := false.

16. TC Messages

This protocol defines, and hence owns, the TC message type (see Section 24). Thus, as specified in [RFC5444], this protocol generates and transmits all TC messages, receives all TC messages and is responsible for determining whether and how each TC message is to be processed (updating the Topology Information Base) and/or forwarded, according to this specification.

16.1. TC Message Generation

A TC message is a message as defined in [RFC5444]. A generated TC message MUST contain the following elements as defined in [RFC5444]:

o A message originator address, recording this router’s originator address. This MUST use a <msg-orig-addr> element.

o <msg-seq-num> element containing the message sequence number.

o A <msg-hop-limit> element, containing TC_HOP_LIMIT. A router MAY use the same or different values of TC_HOP_LIMIT in its TC messages, see Section 5.4.7.

o A <msg-hop-count> element, containing zero, if the message contains a TLV with either Type = VALIDITY_TIME or Type = INTERVAL_TIME (as specified in [RFC5497]) indicating more than one time value according to distance. A TC message MAY contain such a <msg-hop-count> element even if it does not need to.

o A single Message TLV with Type := CONT_SEQ_NUM and Value := ANSN from the Neighbor Information Base. If the TC message is complete then this Message TLV MUST have Type Extension := COMPLETE, otherwise it MUST have Type Extension := INCOMPLETE. (Exception: a TC message MAY omit such a Message TLV if the TC message does not include any address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY.)

- A single Message TLV with Type := VALIDITY_TIME, as specified in [RFC5497]. If all TC messages are sent with the same hop limit then this TLV MUST have a value encoding the period T_HOLD_TIME. If TC messages are sent with different hop limits (more than one value of TC_HOP_LIMIT) then this TLV MUST specify times that vary with the number of hops distance appropriate to the chosen pattern of TC message hop limits, as specified in [RFC5497]; these times SHOULD be appropriate multiples of T_HOLD_TIME. The options included in [RFC5497] for representing zero and infinite times MUST NOT be used.

- If the TC message is complete, all network addresses which are the N_orig_addr of a Neighbor Tuple with N_advertised = true, MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object that is included MUST be associated with an Address Block TLV with Type := NBR_ADDR_TYPE, and Value := ORIGINATOR, or with Value := ROUTABLE_ORIG if that address object is also to be associated with Value = ROUTABLE.

- If the TC message is complete, all routable addresses which are in the N_neighbor_addr_list of a Neighbor Tuple with N_advertised = true MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object MUST be associated with an Address Block TLV with Type = NBR_ADDR_TYPE, and Value = ROUTABLE, or with Value = ROUTABLE_ORIG if also to be associated with Value = ORIGINATOR. At least one copy of each such address object MUST be associated with an Address Block TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE indicating an outgoing neighbor metric with value equal to the corresponding N_out_metric, unless that value is DEFAULT_METRIC.

- If the TC message is complete, all network addresses which are the AL_net_addr of a Local Attached Network Tuple MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object MUST be associated with an Address Block TLV with Type := GATEWAY, and Value := AN_dist. At least one copy of each such address object MUST be associated with an Address Block TLV with Type =

LINK_METRIC and Type Extension = LINK_METRIC_TYPE indicating an outgoing neighbor metric equal to the corresponding AL_metric, unless that value is DEFAULT_METRIC.

A TC message MAY contain:

- A single Message TLV with Type := INTERVAL_TIME, as specified in [RFC5497]. If all TC messages are sent with the same hop limit then this TLV MUST have a value encoding the period TC_INTERVAL. If TC messages are sent with different hop limits, then this TLV MUST specify times that vary with the number of hops distance appropriate to the chosen pattern of TC message hop limits, as specified in [RFC5497]; these times SHOULD be appropriate multiples of TC_INTERVAL. The options included in [RFC5497] for representing zero and infinite times MUST NOT be used.

16.2. TC Message Transmission

A router with one or more OLSRv2 interfaces, and with any Neighbor Tuples with N_advertised = true, or with a non-empty Local Attached Network Set MUST generate TC messages. A router which does not have such information to advertise SHOULD also generate "empty" TC messages for a period A_HOLD_TIME after it last generated a non-empty TC message.

Complete TC messages are generated and transmitted periodically on all OLSRv2 interfaces, with a default interval between two consecutive TC message transmissions by the same router of TC_INTERVAL.

TC messages MAY be generated in response to a change in the information which they are to advertise, indicated by a change in the ANSN in the Neighbor Information Base. In this case a router MAY send a complete TC message, and if so MAY re-start its TC message schedule. Alternatively a router MAY send an incomplete TC message with at least the newly advertised network addresses (i.e., not previously, but now, an N_orig_addr or in an N_neighbor_addr_list in a Neighbor Tuple with N_advertised = true, or an AL_net_addr) in its Address Blocks, with associated Address Block TLV(s). Note that a router cannot report removal of advertised content using an incomplete TC message.

When sending a TC message in response to a change of advertised network addresses, a router MUST respect a minimum interval of TC_MIN_INTERVAL between generated TC messages. Sending an incomplete TC message MUST NOT cause the interval between complete TC messages to be increased, and thus a router MUST NOT send an incomplete TC message if within TC_MIN_INTERVAL of the next scheduled complete TC message.

The generation of TC messages, whether scheduled or triggered by a change of contents, MAY be jittered as described in [RFC5148]. The values of MAXJITTER used SHOULD be:

- TP_MAXJITTER for periodic TC message generation;
- TT_MAXJITTER for responsive TC message generation.

16.3. TC Message Processing

On receiving a TC message, the receiving router MUST then follow the processing and forwarding procedure, defined in Section 14.

If the message is considered for processing (Section 14.2), then a router MUST first check if the message is invalid for processing by this router, as defined in Section 16.3.1. A router MAY make a similar check before considering a message for forwarding, it MUST make those aspects of the check that apply to elements in the Message Header.

If the TC message is not invalid, then the TC message type specific processing, described in Section 16.3.2 MUST be applied. This will update its appropriate Interface Information Base and its Router Information Base. Following this, if there are any changes in these Information Bases, then the processing in Section 17 MUST be performed.

16.3.1. Invalid Message

A received TC message is invalid for processing by this router if the message:

- Has an address length specified in the Message Header that is not equal to the length of the addresses used by this router.
- Does not include a message originator address and a message sequence number.
- Does not include a hop count, and contains a multi-value TLV with Type = VALIDITY_TIME or Type = INTERVAL_TIME, as defined in [RFC5497].
- Does not have exactly one Message TLV with Type = VALIDITY_TIME.
- Has more than one Message TLV with Type = INTERVAL_TIME.

- Does not have a Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE or Type Extension = INCOMPLETE, and contains at least one address object associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY.

- Has more than one Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE or Type Extension = INCOMPLETE.

- Has a message originator address that is partially owned by this router.

- Includes any address object with a prefix length which is not maximal (equal to the address length, in bits), associated with an Address Block TLV with Type = NBR_ADDR_TYPE and Value = ORIGINATOR or Value = ROUTABLE_ORIG.

- Includes any address object that represents a non-routable address, associated with an Address Block TLV with Type = NBR_ADDR_TYPE and Value = ROUTABLE or Value = ROUTABLE_ORIG.

- Includes any address object associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY that also represents the message's originator address.

- Includes any address object (including different copies of an address object, in the same or different Address Blocks) that is associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY, that is also associated with more than one outgoing neighbor metric using a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE.

- Associates any address object (including different copies of an address object, in the same or different Address Blocks) with more than one single hop count value using one or more Address Block TLV(s) with Type = GATEWAY.

- Associates any address object (including different copies of an address object, in the same or different Address Blocks) with Address Block TLVs with Type = NBR_ADDR_TYPE and Type = GATEWAY.

A router MAY recognize additional reasons for identifying that a message is invalid. An invalid message MUST be silently discarded, without updating the router's Information Bases.

16.3.2. TC Message Processing Definitions

When, according to Section 14.2, a TC message is to be "processed according to its type", this means that:

- If the TC message contains a Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE, then processing according to Section 16.3.3 and then according to Section 16.3.4 is carried out.

- If the TC message contains a Message TLV with Type = CONT_SEQ_NUM and Type Extension = INCOMPLETE, then only processing according to Section 16.3.3 is carried out.

For the purposes of this section:

- "validity time" is calculated from a VALIDITY_TIME Message TLV in the TC message according to the specification in [RFC5497]. All information in the TC message has the same validity time.

- "received ANSN" is defined as being the value of a Message TLV with Type = CONT_SEQ_NUM.

- "associated metric value" is defined for any address in the TC message as being either the outgoing neighbor metric value indicated by a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that is associated with any address object in the TC message that is equal to that address, or as DEFAULT_METRIC otherwise. (Note that the rules in Section 16.3.1 make this definition unambiguous.)

- Comparisons of sequence numbers are carried out as specified in Section 21.

16.3.3. Initial TC Message Processing

The TC message is processed as follows:

1. The Advertising Remote Router Set is updated according to Section 16.3.3.1. If the TC message is indicated as discarded in that processing then the following steps are not carried out.

2. The Router Topology Set is updated according to Section 16.3.3.2.

3. The Routable Address Topology Set is updated according to Section 16.3.3.3.

4. The Attached Network Set is updated according to Section 16.3.3.4.

16.3.3.1. Populating the Advertising Remote Router Set

The router MUST update its Advertising Remote Router Set as follows:

1. If there is an Advertising Remote Router Tuple with:

   * AR_orig_addr = message originator address, AND;

   * AR_seq_number > received ANSN,

   then the TC message MUST be discarded.

2. Otherwise:

   1. If there is no Advertising Remote Router Tuple such that:

      + AR_orig_addr = message originator address;

      then create an Advertising Remote Router Tuple with:

      + AR_orig_addr := message originator address.

   2. This Advertising Remote Router Tuple (existing or new) is then modified as follows:

      + AR_seq_number := received ANSN;

      + AR_time := current time + validity time.

16.3.3.2. Populating the Router Topology Set

The router MUST update its Router Topology Set as follows:

1. For each address (henceforth advertised address) corresponding to one or more address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE and Value = ORIGINATOR or Value = ROUTABLE_ORIG, and that is not partially owned by this router, perform the following processing:

   1. If there is no Router Topology Tuple such that:

      + TR_from_orig_addr = message originator address, AND;

      + TR_to_orig_addr = advertised address,

      then create a new Router Topology Tuple with:

    - TR_from_orig_addr := message originator address;

    - TR_to_orig_addr := advertised address.

2. This Router Topology Tuple (existing or new) is then modified as follows:

    - TR_seq_number := received ANSN;

    - TR_metric := associated link metric;

    - TR_time := current time + validity time.

16.3.3.3. Populating the Routable Address Topology Set

The router MUST update its Routable Address Topology Set as follows:

1. For each network address (henceforth advertised address) corresponding to one or more address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE and Value = ROUTABLE or Value = ROUTABLE_ORIG, and that is not partially owned by this router, perform the following processing:

    1. If there is no Routable Address Topology Tuple such that:

        - TA_from_orig_addr = message originator address, AND;

        - TA_dest_addr = advertised address,

        then create a new Routable Address Topology Tuple with:

        - TA_from_orig_addr := message originator address;

        - TA_dest_addr := advertised address.

    2. This Routable Address Topology Tuple (existing or new) is then modified as follows:

        - TA_seq_number := received ANSN;

        - TA_metric := associated link metric;

        - TA_time := current time + validity time.

16.3.3.4. Populating the Attached Network Set

The router MUST update its Attached Network Set as follows:

1. For each network address (henceforth attached address) corresponding to one or more address objects with an associated Address Block TLV with Type = GATEWAY, and that is not fully owned by this router, perform the following processing:

    1. If there is no Attached Network Tuple such that:

        + AN_net_addr = attached address, AND;

        + AN_orig_addr = message originator address,

        then create a new Attached Network Tuple with:

        + AN_net_addr := attached address;

        + AN_orig_addr := message originator address.

    2. This Attached Network Tuple (existing or new) is then modified as follows:

        + AN_seq_number := received ANSN;

        + AN_dist := the Value of the associated GATEWAY TLV;

        + AN_metric := associated link metric;

        + AN_time := current time + validity time.

16.3.4. Completing TC Message Processing

The TC message is processed as follows:

1. The Router Topology Set is updated according to Section 16.3.4.1.

2. The Routable Address Topology Set is updated according to Section 16.3.4.2.

3. The Attached Network Set is updated according to Section 16.3.4.3.

16.3.4.1. Purging the Router Topology Set

The Router Topology Set MUST be updated as follows:

1. Any Router Topology Tuples with:

   * TR_from_orig_addr = message originator address, AND;

   * TR_seq_number < received ANSN,

   MUST be removed.

16.3.4.2. Purging the Routable Address Topology Set

The Routable Address Topology Set MUST be updated as follows:

1. Any Routable Address Topology Tuples with:

   * TA_from_orig_addr = message originator address, AND;

   * TA_seq_number < received ANSN,

   MUST be removed.

16.3.4.3. Purging the Attached Network Set

The Attached Network Set MUST be updated as follows:

1. Any Attached Network Tuples with:

   * AN_orig_addr = message originator address, AND;

   * AN_seq_number < received ANSN,

   MUST be removed.

17. Information Base Changes

The changes described in the following sections MUST be carried out when any Information Base changes as indicated.

17.1. Originator Address Changes

If the router changes originator address, then:

1. If there is no Originator Tuple with:

- O_orig_addr = old originator address

then create an Originator Tuple with:

- O_orig_addr := old originator address

The Originator Tuple (existing or new) with:

- O_orig_addr = new originator address

is then modified as follows:

- O_time := current time + O_HOLD_TIME

17.2. Link State Changes

The consistency of a Link Tuple MUST be maintained according to the following rules, in addition to those in [RFC6130]:

- o If L_status = HEARD or L_status = SYMMETRIC, then L_in_metric MUST be set (by a process outside this specification).

- o If L_status != SYMMETRIC, then set L_mpr_selector := false.

- o If L_out_metric = UNKNOWN_METRIC, then L_status MUST NOT equal SYMMETRIC; set L_SYM_time := expired if this would otherwise be the case.

17.3. Neighbor State Changes

The consistency of a Neighbor Tuple MUST be maintained according to the following rules, in addition to those in [RFC6130]:

1. If N_symmetric = true, then N_in_metric MUST equal the minimum value of all L_in_metric of corresponding Link Tuples with L_status = SYMMETRIC.

2. If N_symmetric = true, then N_out_metric MUST equal the minimum value of all L_out_metric of corresponding Link Tuples with L_status = SYMMETRIC.

3. If N_symmetric = false, then N_flooding_mpr, N_routing_mpr, N_mpr_selector and N_advertised MUST all be equal to false.

4. If N_mpr_selector = true, then N_advertised MUST be equal to true.

5. If N_symmetric = true and N_mpr_selector = false, then a router MAY select N_advertised = true or N_advertised = false. The more neighbors that are advertised, the larger TC messages become, but the more redundancy is available for routing. A router SHOULD consider the nature of its network in making such a decision, and SHOULD avoid unnecessary changes in advertising status, which may result both in additional TC messages having to be sent by its neighbors, and in unnecessary changes to routing, which will have similar effects to other forms of topology changes in the MANET.

17.4. Advertised Neighbor Changes

The router MUST increment the ANSN in the Neighbor Information Base whenever:

1. Any Neighbor Tuple changes its N_advertised value, or any Neighbor Tuple with N_advertised = true is removed.

2. Any Neighbor Tuple with N_advertised = true changes its N_orig_addr, or has any routable address is added to or removed from N_neighbor_addr_list.

3. Any Neighbor Tuple with N_advertised = true has N_out_metric changed.

4. There is any change to the Local Attached Network Set.

17.5. Advertising Remote Router Tuple Expires

The Router Topology Set, the Routable Address Topology Set and the Attached Network Set MUST be changed when an Advertising Remote Router Tuple expires (AR_time is reached). The following changes are required before the Advertising Remote Router Tuple is removed:

1. All Router Topology Tuples with:

   * TR_from_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

2. All Routable Address Topology Tuples with:

   * TA_from_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

3. All Attached Network Tuples with:

   * AN_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

17.6. Neighborhood Changes and MPR Updates

The sets of symmetric 1-hop neighbors selected as flooding MPRs and routing MPRs MUST satisfy the conditions defined in Section 18. To ensure this:

1. The set of flooding MPRs of a router MUST be recalculated if:

   * a Link Tuple is added with L_status = SYMMETRIC, OR;

   * a Link Tuple with L_status = SYMMETRIC is removed, OR;

   * a Link Tuple with L_status = SYMMETRIC changes to having L_status = HEARD or L_status = LOST, OR;

   * a Link Tuple with L_status = HEARD or L_status = LOST changes to having L_status = SYMMETRIC, OR;

   * the flooding MPR selection process uses metrics (see Section 18.4 and the L_out_metric of any Link Tuple with L_status = SYMMETRIC changes, OR;

   * a 2-Hop Tuple is added or removed, OR;

   * the N_will_flooding of a Neighbor Tuple with N_symmetric = true changes from WILL_NEVER to any other value, OR;

   * the N_will_flooding of a Neighbor Tuple with N_flooding_mpr = true changes to WILL_NEVER from any other value, OR;

   * the N_will_flooding of a Neighbor Tuple with N_symmetric = true and N_flooding_mpr = false changes to WILL_ALWAYS from any other value, OR;

   * the flooding MPR selection process uses metrics (see Section 18.4 and the N2_out_metric of any 2-Hop Tuple changes.

2. Otherwise, the set of flooding MPRs of a router MAY be recalculated if the N_will_flooding of a Neighbor Tuple with N_symmetric = true changes in any other way; it SHOULD be recalculated if N_flooding_mpr = false and this is an increase in

N_will_flooding or if N_flooding_mpr = true and this is a decrease in N_will_flooding.

3. The set of routing MPRs of a router MUST be recalculated if:

    * a Link Tuple is added with L_status = SYMMETRIC, OR;

    * a Link Tuple with L_status = SYMMETRIC is removed, OR;

    * a Link Tuple with L_status = SYMMETRIC changes to having L_status = HEARD or L_status = LOST, OR;

    * a Link Tuple with L_status = HEARD or L_status = LOST changes to having L_status = SYMMETRIC, OR;

    * a 2-Hop Tuple is added or removed, OR;

    * the N_will_routing of a Neighbor Tuple with N_symmetric = true changes from WILL_NEVER to any other value, OR;

    * the N_will_routing of a Neighbor Tuple with N_routing_mpr = true changes to WILL_NEVER from any other value, OR;

    * the N_will_routing of a Neighbor Tuple with N_symmetric = true and N_routing_mpr = false changes to WILL_ALWAYS from any other value, OR;

    * the N_in_metric of any Neighbor Tuple with N_symmetric changes, OR;

    * the N2_in_metric of any 2-Hop Tuple changes.

4. Otherwise, the set of routing MPRs of a router MAY be recalculated if the N_will_routing of a Neighbor Tuple with N_symmetric = true changes in any other way; it SHOULD be recalculated if N_routing_mpr = false and this is an increase in N_will_routing or if N_routing_mpr = true and this is a decrease in N_will_routing.

If either set of MPRs of a router is recalculated, this MUST be as described in Section 18.

17.7. Routing Set Updates

The Routing Set MUST be updated, as described in Section 19, when changes in the Local Information Base, the Neighborhood Information Base or the Topology Information Base indicate a change (including of any potentially used outgoing neighbor metric values) of the known symmetric links and/or attached networks in the MANET, hence changing the Topology Graph. It is sufficient to consider only changes which affect at least one of:

- The Local Interface Set, if the change removes any network address in an I_local_iface_addr_list. In this case, unless the OLSRv2 interface is removed, it may not be necessary to do more than replace such network addresses, if used, by an alternative network address from the same I_local_iface_addr_list.

- The Local Attached Set, if the change removes any AL_net_addr which is also an AN_net_addr. In this case it may not be necessary to do more than add Routing Tuples with R_dest_addr equal to that AN_net_addr.

- The Link Set of any OLSRv2 interface, considering only Link Tuples which have, or just had, L_status = SYMMETRIC (including removal of such Link Tuples).

- The Neighbor Set of the router, considering only Neighbor Tuples that have, or just had, N_symmetric = true, and do not have N_orig_addr = unknown.

- The 2-Hop Set of any OLSRv2 interface, if used in the creation of the Routing Set.

- The Router Topology Set of the router.

- The Routable Address Topology Set of the router.

- The Attached Network Set of the router.

18. Selecting MPRs

Each router MUST select, from among its willing symmetric 1-hop neighbors, two subsets of these routers, as flooding and routing MPRs. This selection is recorded in the router's Neighbor Set, and reported in the router's HELLO messages. Routers MAY select their MPRs by any process that satisfies the conditions which follow, which may, but need not, use the organization of the data described. Routers can freely interoperate whether they use the same or different MPR selection algorithms.

Only flooding MPRs forward control messages flooded through the MANET, thus effecting a flooding reduction, an optimization of the flooding mechanism, known as MPR flooding. Routing MPRs are used to effect a topology reduction in the MANET. (If no such reduction is required then a router can select all of its relevant neighbors as routing MPRs.) Consequently, while it is not essential that these two sets of MPRs are minimal, keeping the numbers of MPRs small ensures that the overhead of this protocol is kept to a minimum.

18.1. Overview

MPRs are selected according to the following steps, defined in the following sections:

- A form of data structure known as a Neighbor Graph is defined.
- The properties of an MPR Set derived from a Neighbor Graph are defined. Any algorithm that creates an MPR Set that satisfies these properties is a valid MPR selection algorithm. An example algorithm that creates such an MPR Set is given in Appendix A.
- How to create a Neighbor Graph for each interface based on the corresponding Interface Information Base is defined, and how to combine the resulting MPR Sets to determine the router's flooding MPRs and record those in the router's Neighbor Set.
- How to create a single Neighbor Graph based on all Interface Information Bases and the Neighbor Information Base is defined, and how to record the resulting MPR Set as the router's routing MPRs in the router's Neighbor Set.
- A specification as to when MPRs MUST be calculated is given.

When a router selects its MPRs it MAY consider any other characteristics of its neighbors that it is aware of. In particular it SHOULD consider the willingness of the neighbor, as recorded by the corresponding N_will_flooding or N_will_routing value, as appropriate, preferring neighbors with higher values. (Note that willingness values equal to WILL_NEVER and WILL_ALWAYS are always considered, as described.) However a router MAY consider other characteristics to have a greater significance.

Each router MAY select its flooding and routing MPRs independently from each other, or coordinate its selections. A router MAY make its MPR selections independently of the MPR selection by other routers, or it MAY, for example, give preference to routers that either are, or are not, already selected as MPRs by other routers.

18.2. Neighbor Graph

A Neighbor Graph is a structure defined here as consisting of sets N1 and N2 and some associated metric and willingness values. Elements of set N1 represent willing symmetric 1-hop neighbors, and elements of set N2 represent addresses of a symmetric 2-hop neighbor.

A Neighbor Graph has the following properties:

- It contains two disjoint sets N1 and N2.

- For each element x in N1 there is an associated willingness value W(x) such that WILL_NEVER < W(x) <= WILL_ALWAYS.

- For each element x in N1 there is an associated metric d1(x) > 0.

- For some elements y in N2 there is an associated metric d1(y) > 0. (Other elements y in N2 have undefined d1(y), this may be considered to be infinite.)

- For each element x in N1 there is a subset N2(x) of elements of N2; this subset may be empty. For each x in N1 and each y in N2(x) there is an associated metric d2(x,y) > 0. (For other x in N1 and y in N2, d2(x,y) is undefined, and may be considered infinite.)

- N2 is equal to the union of all the N2(x) for all x in N1, i.e. for each y in N2 there is at least one x in N1 such that y is in N2(x).

It is convenient to also define:

- For each y in N2 the set N1(y) that contains x in N1 if and only if y is in N2(x). From the final property above, N1(y) is not empty.

- For each x in N1 and y in N2, if d2(x,y) is defined then d(x,y) := d1(x)+d2(x,y), otherwise d(x,y) is not defined. (Thus d(x,y) is defined if y is in N2(x), or equivalently if x is in N1(y).)

- For any subset S of N1, and for each y in N2, the metric d(y,S) is the minimum value of d1(y), if defined, and of all d(x,y) for x in N1(y) and in S. If there are no such metrics to take the minimum value of, then d(y,S) is undefined (may be considered to be infinite). From the final property above, d(y,N1) is defined for all y.

18.3. MPR Properties

Given a Neighbor Graph as defined in Section 18.2. an MPR Set for that Neighbor Graph is a subset M of the set N1 that satisfies the following properties:

- o If x in N1 has W(x) = WILL_ALWAYS then x is in M.

- o For any y in N2 that does not have a defined d1(y), there is at least one element in M that is also in N1(y). This is equivalent to the requirement that d(y,M) is defined.

- o For any y in N2, d(y,M) = d(y,N1).

These two properties correspond first to that the MPR Set consists of a set of symmetric 1-hop neighbors that cover all the symmetric 2-hop neighbors, and second that they do so retaining a minimum distance route (1-hop, if present, or 2-hop) to each symmetric 2-hop neighbor.

Note that if M is an MPR Set, then so is any subset of N1 that contains M, and also that N1 is always an MPR Set. An MPR Set may be empty, but cannot be empty if N2 contains any elements y that do not have a defined d1(y).

18.4. Flooding MPRs

Whenever flooding MPRs are to be calculated, an implementation MUST determine and record a set of flooding MPRs that is equivalent to one calculated as described in this section.

The calculation of flooding MPRs need not use link metrics, or equivalently may use link metrics with a fixed value, here taken to be 1. Routers MAY make individual decisions as to whether to use link metrics for the calculation of flooding MPRs. A router MUST use the same approach to the choice of whether to use link metrics for all links, i.e. in the cases indicated by a or b, the same choice MUST be made in each case.

For each OLSRv2 interface (the "current interface") define a Neighbor Graph as defined in Section 18.2 according to the following:

- o Define a reachable Link Tuple to be a Link Tuple in the Link Set for the current interface with L_status = SYMMETRIC.

- o Define an allowed Link Tuple to be a reachable Link Tuple whose corresponding Neighbor Tuple has N_will_flooding > WILL_NEVER.

- o Define an allowed 2-Hop Tuple to be a 2-Hop Tuple in the 2-Hop Set for the current interface for which there is an allowed Link Tuple with L_neighbor_iface_addr_list = N2_neighbor_iface_addr_list.

- o Define an element of N1 for each allowed Link Tuple. This then defines the corresponding Link Tuple for each element of N1 and the corresponding Neighbor Tuple for each element of N1, being the

Neighbor Tuple corresponding to that Link Tuple.

- For each element x in N1, define W(x) := N_will_flooding of the corresponding Neighbor Tuple.

- For each element x in N1, define d1(x) as either:

  A. L_out_metric of the corresponding Link Tuple, OR;

  B. 1.

- Define an element of N2 for each network address that is the N2_2hop_addr of one or more allowed 2-Hop Tuples. This then defines the corresponding address for each element of N2.

- For each element y in N2, if the corresponding address is in the N_neighbor_addr_list of a Neighbor Tuple that corresponds to one or more reachable Link Tuples, then define d1(y) as either:

  A. the minimum value of the L_out_metric of those Link Tuples, OR;

  B. 1.

  Otherwise d1(y) is not defined. In the latter case, where d1(y) : =1, all such y in N2 may instead be removed from N2.

- For each element x in N1, define N2(x) as the set of elements y in N2 whose corresponding address is the N2_2hop_addr of an allowed 2-Hop Tuple that has N2_neighbor_iface_addr_list = L_neighbor_iface_addr_list of the Link Tuple corresponding to x. For all such x and y, define d2(x,y) as either:

  A. N2_out_metric of that 2-Hop Tuple;

  B. 1.

It is up to the implementer to decide how to label each element of N1 or N2. For example an element of N1 may be labeled with one or more addresses from the corresponding L_neighbor_iface_addr_list, or with a pointer or reference to the corresponding Link Tuple.

Using these Neighbor Graphs, flooding MPRs are selected and recorded by:

- For each OLSRv2 interface, determine an MPR Set as specified in Section 18.3.

- A Neighbor Tuple represents a flooding MPR and has N_flooding_mpr := true (otherwise N_flooding_mpr := false) if and only if that Neighbor Tuple corresponds to an element in an MPR Set created for any interface as described above. That is, the overall set of flooding MPRs is the union of the sets of flooding MPRs for all OLSRv2 interfaces.

A router MAY select its flooding MPRs for each OLSRv2 interface independently, or it MAY coordinate its MPR selections across its OLSRv2 interfaces, as long as the required condition is satisfied for each OLSRv2 interface. One such coordinated approach is to process the OLSRv2 interfaces sequentially, and for each OLSRv2 interface start with flooding MPRs selected (and not removable) if the neighbor has been already selected as an MPR for an OLSRv2 interface that has already been processed. The algorithm specified in Appendix A can be used in this way.

18.5. Routing MPRs

Whenever routing MPRs are to be calculated, an implementation MUST determine and record a set of routing MPRs that is equivalent to one calculated as described in this section.

Define a single Neighbor Graph as defined in Section 18.2 according to the following:

- Define a reachable Neighbor Tuple to be a Neighbor Tuple with N_symmetric = true.

- Define an allowed Neighbor Tuple to be a reachable Neighbor Tuple with N_will_routing > WILL_NEVER.

- Define an allowed 2-Hop Tuple to be a 2-Hop Tuple in the 2-Hop Set for any OLSRv2 interface for which there is an allowed Neighbor Tuple with N_neighbor_addr_list containing N2_neighbor_iface_addr_list.

- Define an element of N1 for each allowed Neighbor Tuple. This then defines the corresponding Neighbor Tuple for each element of N1.

- For each element x in N1, define W(x) := N_will_routing of the corresponding Neighbor Tuple.

- For each element x in N1, define d1(x) := N_in_metric of the corresponding Neighbor Tuple.

- o Define an element of N2 for each network address that is the N2_2hop_addr of one or more allowed 2-Hop Tuples. This then defines the corresponding address for each element of N2.

- o For each element y in N2, if the corresponding address is in the N_neighbor_addr_list of a reachable Neighbor Tuple, then define d1(y) to be the N_in_metric of that Neighbor Tuple, otherwise d1(y) is not defined.

- o For each element x in N1, define N2(x) as the set of elements y in N2 whose corresponding address is the N2_2hop_addr of an allowed 2-Hop Tuple that has N2_neighbor_iface_addr_list contained in N_neighbor_addr_list of the Neighbor Tuple corresponding to x. For all such x and y, define d2(x,y) := N2_out_metric of that 2-Hop Tuple.

It is up to the implementer to decide how to label each element of N1 or N2. For example an element of N1 may be labeled with one or more addresses from the corresponding N_neighbor_addr_list, or with a pointer or reference to the corresponding Neighbor Tuple.

Using these Neighbor Graphs, routing MPRs are selected and recorded by:

- o Determine an MPR Set as specified in Section 18.3

- o A Neighbor Tuple represents a routing MPR and has N_routing_mpr := true (otherwise N_routing_mpr := false) if and only if that Neighbor Tuple corresponds to an element in the MPR Set created as described above.

18.6. Calculating MPRs

A router MUST recalculate each of its sets of MPRs whenever the currently selected set of MPRs does not still satisfy the required conditions. It MAY recalculate its MPRs if the current set of MPRs is still valid, but could be more efficient. Sufficient conditions to recalculate a router's sets of MPRs are given in Section 17.6.

19. Routing Set Calculation

The Routing Set of a router is populated with Routing Tuples that represent paths from that router to all destinations in the network. These paths are calculated based on the Network Topology Graph, which is constructed from information in the Information Bases, obtained via HELLO and TC message exchange.

Changes to the Routing Set do not require any messages to be transmitted. The state of the Routing Set SHOULD, however, be reflected in IP's routing table by adding and removing entries from IP's routing table as appropriate. Only appropriate Routing Tuples (in particular only those that represent local links or paths to routable addresses) need be reflected in IP's routing table.

19.1. Network Topology Graph

The Network Topology Graph is formed from information from the router's Local Interface Set, Link Sets, Neighbor Set, Router Topology Set, Routable Address Topology Set and Attached Network Set. The Network Topology Graph MAY also use information from the router's 2-Hop Sets. The Network Topology Graph forms the router's topological view of the network in form of a directed graph. Each edge in that graph has a metric value. The Network Topology Graph has a "backbone" (within which minimum total metric routes will be constructed) containing the following edges:

- Edges X -> Y for all possible Y, and one X per Y, such that:

  - Y is the N_orig_addr of a Neighbor Tuple, AND;

  - N_orig_addr is not unknown;

  - X is in the I_local_iface_addr_list of a Local Interface Tuple, AND;

  - There is a Link Tuple with L_status = SYMMETRIC such that this Neighbor Tuple and this Local Interface Tuple correspond to it. A network address from L_neighbor_iface_addr_list will be denoted R in this case.

  It SHOULD be preferred, where possible, to select R = S and X from the Local Interface Tuple corresponding to the Link Tuple from which R was selected. The metric such an edge is the corresponding N_out_metric.

- All edges W -> U such that:

  - W is the TR_from_orig_addr of a Router Topology Tuple, AND;

  - U is the TR_to_orig_addr of the same Router Topology Tuple.

  The metric of such an edge is the corresponding TR_metric.

The Network Topology Graph is further "decorated" with the following edges. If a network address S, V, Z or T equals a network address Y or W, then the edge terminating in the network address S, V, Z or T

MUST NOT be used in any path.

- Edges X -> S for all possible S, and one X per S, such that:

  * S is in the N_neighbor_addr_list of a Neighbor Tuple, AND;

  * X is in the I_local_iface_addr_list of a Local Interface Tuple, AND;

  * There is a Link Tuple with L_status = SYMMETRIC such that this Neighbor Tuple and this Local Interface Tuple correspond to it. A network address from L_neighbor_iface_addr_list will be denoted R in this case.

  It SHOULD be preferred, where possible, to select R = S and X from the Local Interface Tuple corresponding to the Link Tuple from which R was selected. The metric of such an edge is the corresponding N_out_metric.

- All edges W -> V such that:

  * W is the TA_from_orig_addr of a Routable Address Topology Tuple, AND;

  * V is the TA_dest_addr of the same Routable Address Topology Tuple.

  The metric for such an edge is the corresponding TA_metric.

- All edges W -> T such that:

  * W is the AN_orig_addr of an Attached Network Tuple, AND;

  * T is the AN_net_addr of the same Attached Network Tuple.

  The metric for such an edge is the corresponding AN_metric.

- OPTIONALLY, all edges Y -> Z such that:

  * Z is a routable address and is the N2_2hop_addr of a 2-Hop Tuple, AND;

  * Y is the N_orig_addr of the corresponding Neighbor Tuple, AND;

  * This Neighbor Tuple has N_will_routing not equal to WILL_NEVER.

  A path terminating with such an edge SHOULD NOT be used in preference to any other path. The metric for such an edge is the corresponding N2_out_metric.

Any part of the Topology Graph which is not connected to a local network address X is not used. Only one selection X SHOULD be made from each I_local_iface_addr_list, and only one selection R SHOULD be made from any L_neighbor_iface_addr_list. All edges have a hop count of 1, except edges W -> T that have a hop count of the corresponding value of AN_dist.

19.2. Populating the Routing Set

The Routing Set MUST contain the shortest paths for all destinations from all local OLSRv2 interfaces using the Network Topology Graph. This calculation MAY use any algorithm, including any means of choosing between paths of equal total metric. (In the case of two paths of equal total metric but differing hop counts, the path with the lower hop count SHOULD be used.)

Using the notation of Section 19.1, initially "backbone" paths using only edges X -> Y and W -> U need be constructed (using a minimum distance algorithm). Then paths using only a final edge of the other types may be added. These MUST NOT replace backbone paths with the same destination (and paths terminating in an edge Y -> Z SHOULD NOT replace paths with any other form of terminating edge).

Each path will correspond to a Routing Tuple. These will be of two types. The first type will represent single edge paths, of type X -> S or X -> Y, by:

- R_local_iface_addr := X;

- R_next_iface_addr := R;

- R_dest_addr := S or Y;

- R_dist := 1;

- R_metric := edge metric.

where R is as defined in Section 19.1 for these types of edges.

The second type will represent a multiple edge path, which will always have first edge of type X -> Y, and will have final edge of type W -> U, W -> V, W -> T or Y -> Z. The Routing Tuple will be:

- R_local_iface_addr := X;

- R_next_iface_addr := Y;

- R_dest_addr := U, V, T or Z;

- R_dist := the total hop count of all edges in the path;

- R_metric := the total metric of all edges in the path.

Finally, Routing Tuples of the second type whose R_dest_addr is not routable MAY be discarded.

An example algorithm for calculating the Routing Set of a router is given in Appendix B.

20. Proposed Values for Parameters

This protocol uses all parameters defined in [RFC6130] and additional parameters and defined in this specification. All but one (RX_HOLD_TIME) of these additional parameters are router parameters as defined in [RFC6130]. The proposed values of the additional parameters defined in the following sections are appropriate to the case where all parameters (including those defined in [RFC6130]) have a single value. Proposed values for parameters defined in [RFC6130] are given in that specification.

20.1. Local History Time Parameters

- O_HOLD_TIME := 30 seconds

20.2. Message Interval Parameters

- TC_INTERVAL := 5 seconds

- TC_MIN_INTERVAL := TC_INTERVAL/4

20.3. Advertised Information Validity Time Parameters

- T_HOLD_TIME := 3 x TC_INTERVAL

- A_HOLD_TIME := T_HOLD_TIME

20.4. Received Message Validity Time Parameters

- RX_HOLD_TIME := 30 seconds

- P_HOLD_TIME := 30 seconds

- F_HOLD_TIME := 30 seconds

20.5. Jitter Time Parameters

- TP_MAXJITTER := HP_MAXJITTER
- TT_MAXJITTER := HT_MAXJITTER
- F_MAXJITTER := TT_MAXJITTER

20.6. Hop Limit Parameter

- TC_HOP_LIMIT := 255

20.7. Willingness Parameter

- WILLINGNESS := WILL_DEFAULT

21. Sequence Numbers

Sequence numbers are used in this specification for the purpose of discarding "old" information, i.e., messages received out of order. However with a limited number of bits for representing sequence numbers, wrap-around (that the sequence number is incremented from the maximum possible value to zero) will occur. To prevent this from interfering with the operation of this protocol, the following MUST be observed when determining the ordering of sequence numbers.

The term MAXVALUE designates in the following one more than the largest possible value for a sequence number. For a 16 bit sequence number (as are those defined in this specification) MAXVALUE is 65536.

The sequence number S1 is said to be "greater than" the sequence number S2 if:

- S1 > S2 AND S1 - S2 < MAXVALUE/2 OR
- S2 > S1 AND S2 - S1 > MAXVALUE/2

When sequence numbers S1 and S2 differ by MAXVALUE/2 their ordering cannot be determined. In this case, which should not occur, either ordering may be assumed.

Thus when comparing two messages, it is possible - even in the presence of wrap-around - to determine which message contains the most recent information.

22. Extensions

An extension to this protocol will need to interact with this specification, and possibly also with [RFC6130]. This protocol is designed to permit such interactions, in particular:

- Through accessing, and possibly extending, the information in the Information Bases. All updates to the elements specified in this specification are subject to the constraints specified in [RFC6130] and Appendix D.

- Through accessing an outgoing message prior to it being transmitted over any OLSRv2 interface, and to add information to it as specified in [RFC5444]. This MAY include Message TLVs and/or network addresses with associated Address Block TLVs. (Network addresses without new associated TLVs SHOULD NOT be added to messages.) This may, for example, be to allow a security protocol, as suggested in Section 23, to add a TLV containing a cryptographic signature to the message.

- Through accessing an incoming message, and potentially discarding it prior to processing by this protocol. This may, for example, allow a security protocol as suggested in Section 23 to perform verification of message signatures and prevent processing and/or forwarding of unverifiable messages by this protocol.

- Through accessing an incoming message after it has been completely processed by this protocol. This may, in particular, allow a protocol which has added information, by way of inclusion of appropriate TLVs, or of network addresses associated with new TLVs, access to such information after appropriate updates have been recorded in the Information Bases in this protocol.

- Through requesting that a message be generated at a specific time. In that case, message generation MUST still respect the constraints in [RFC6130] and Section 5.4.3.

23. Security Considerations

Currently, this protocol does not specify any special security measures. As a proactive routing protocol, this protocol is a potential target for various attacks. Various possible vulnerabilities are discussed in this section.

23.1. Confidentiality

This protocol periodically MPR floods topological information to all routers in the network. Hence, if used in an unprotected wireless network, the network topology is revealed to anyone who listens to the control messages.

In situations where the confidentiality of the network topology is of importance, regular cryptographic techniques, such as exchange of OLSRv2 control traffic messages encrypted by PGP [RFC4880] or encrypted by some shared secret key, can be applied to ensure that control traffic can be read and interpreted by only those authorized to do so.

23.2. Integrity

Each router is injecting topological information into the network through transmitting HELLO messages and, for some routers, TC messages. If some routers for some reason, malicious or malfunction, inject invalid control traffic, network integrity may be compromised. Therefore, message authentication is recommended.

Different such situations may occur, for instance:

1. a router generates TC messages, advertising links to non-neighbor routers;

2. a router generates TC messages, pretending to be another router;

3. a router generates HELLO messages, advertising non-neighbor routers;

4. a router generates HELLO messages, pretending to be another router;

5. a router forwards altered control messages;

6. a router does not forward control messages;

7. a router does not select multipoint relays correctly;

8. a router forwards broadcast control messages unaltered, but does not forward unicast data traffic;

9. a router "replays" previously recorded control traffic from another router.

Authentication of the originator router for control messages (for situations 2, 4 and 5) and on the individual links announced in the control messages (for situations 1 and 3) may be used as a countermeasure. However to prevent routers from repeating old (and correctly authenticated) information (situation 9) temporal information is required, allowing a router to positively identify such delayed messages.

In general, digital signatures and other required security information may be transmitted as a separate Message Type, or signatures and security information may be transmitted within the HELLO and TC messages, using the TLV mechanism. Either option permits that "secured" and "unsecured" routers can coexist in the same network, if desired,

Specifically, the authenticity of entire control packets can be established through employing IPsec authentication headers, whereas authenticity of individual links (situations 1 and 3) require additional security information to be distributed.

An important consideration is that all control messages are transmitted either to all routers in the neighborhood (HELLO messages) or broadcast to all routers in the network (TC messages).

For example, a control message in this protocol is always a point-to-multipoint transmission. It is therefore important that the authentication mechanism employed permits that any receiving router can validate the authenticity of a message. As an analogy, given a block of text, signed by a PGP private key, then anyone with the corresponding public key can verify the authenticity of the text.

23.3. Interaction with External Routing Domains

This protocol does, through the use of TC messages, provide a basic mechanism for injecting external routing information to this protocol's domain. Routing information can be extracted from the protocol's Information Bases, in particular the Routing Set, of this protocol and, potentially, injected into an external domain, if the routing protocol governing that domain permits this.

When operating routers connecting a MANET using this protocol to an external routing domain, care MUST be taken not to allow potentially insecure and untrustworthy information to be injected from this domain to external routing domains. Care MUST also be taken to validate the correctness of information prior to it being injected as to avoid polluting routing tables with invalid information.

A recommended way of extending connectivity from an existing routing domain to a MANET routed using this protocol is to assign an IP prefix (under the authority of the routers/gateways connecting the MANET with the exiting routing domain) exclusively to that MANET area, and to statically configure the gateways to advertise routes for that IP sequence to routers in the existing routing domain.

24. IANA Considerations

This specification defines one Message Type, which must be allocated from the "Message Types" repository of [RFC5444], two Message TLV Types, which must be allocated from the "Message TLV Types" repository of [RFC5444], and four Address Block TLV Types, which must be allocated from the "Address Block TLV Types" repository of [RFC5444].

24.1. Expert Review: Evaluation Guidelines

For the registries where an Expert Review is required, the designated expert SHOULD take the same general recommendations into consideration as are specified by [RFC5444].

24.2. Message Types

This specification defines one Message Type, to be allocated from the 0-223 range of the "Message Types" namespace defined in [RFC5444], as specified in Table 8.

Type	Description
TBD1	TC : Topology Control (MANET-wide signaling)

Table 8: Message Type assignment

24.3. Message-Type-Specific TLV Type Registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for TC messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 9.

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 9: TC Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for TC messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 10.

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 10: TC Message-Type-specific Address Block TLV Types

24.4. Message TLV Types

This specification defines two Message TLV Types, which must be allocated from the "Message TLV Types" namespace defined in [RFC5444]. IANA is requested to make allocations in the 0-127 range for these types. This will create two new Type Extension registries with assignments as specified in Table 11 and Table 12. Specifications of these TLVs are in Section 13.3.1. Each of these TLVs MUST NOT be included more than once in a Message TLV Block.

Name	Type	Type Extension	Description	Allocation Policy
MPR_WILLING	TBD2	0	Bits 0-3 specify the originating router’s willingness to act as a flooding MPR; bits 4-7 specify the originating router’s willingness to act as a routing MPR	
MPR_WILLING	TBD2	1-255	Unassigned	Expert Review

Table 11: Message TLV Type assignment: MPR_WILLING

Name	Type	Type Extension	Description	Allocation Policy
CONT_SEQ_NUM	TBD3	0	COMPLETE : Specifies a content sequence number for this complete message	
CONT_SEQ_NUM	TBD3	1	INCOMPLETE : Specifies a content sequence number for this incomplete message	
CONT_SEQ_NUM	TBD3	2-255	Unassigned	Expert Review

Table 12: Message TLV Type assignment: CONT_SEQ_NUM

Type extensions indicated as Expert Review SHOULD be allocated as described in [RFC5444], based on Expert Review as defined in [RFC5226].

24.5. Address Block TLV Types

This specification defines four Address Block TLV Types, which must be allocated from the "Address Block TLV Types" namespace defined in [RFC5444]. IANA are requested to make allocations in the 8-127 range for these types. This will create four new Type Extension registries with assignments as specified in Table 13, Table 14, Table 15 and Table 16. Specifications of these TLVs are in Section 13.3.2.

Name	Type	Type Extension	Description	Allocation Policy
LINK_METRIC	TBD4	0	Link metric meaning assigned by administrative action	
LINK_METRIC	TBD4	1-223	Unassigned	Expert Review
LINK_METRIC	TBD4	224-255	Unassigned	Experimental Use

Table 13: Address Block TLV Type assignment: LINK_METRIC

All LINK_METRIC TLVs, whatever their type extension, MUST use their value field to encode the kind and value (in the interval MINIMUM_METRIC, to MAXIMUM_METRIC, inclusive) of a link metric as specified in Section 6 and Section 13.3.2. An assignment of a LINK_METRIC TLV type extension MUST specify the physical meaning, and mapping of that physical meaning to the representable values in the indicated interval, of the link metric.

Name	Type	Type Extension	Description	Allocation Policy
MPR	TBD5	0	Specifies that a given network address is of a router selected as a flooding MPR (FLOODING = 1), that a given network address is of a router selected as a routing MPR (ROUTING = 2), or both (FLOOD_ROUTE = 3)	
MPR	TBD5	1-255	Unassigned	Expert Review

Table 14: Address Block TLV Type assignment: MPR

Name	Type	Type Extension	Description	Allocation Policy
NBR_ADDR_TYPE	TBD6	0	Specifies that a given network address is of a neighbor reached via the originating router, if it is an originator address (ORIGINATOR = 1), is a routable address (ROUTABLE = 2), or if it is both (ROUTABLE_ORIG = 3)	
NBR_ADDR_TYPE	TBD6	1-255	Unassigned	Expert Review

Table 15: Address Block TLV Type assignment: NBR_ADDR_TYPE

Name	Type	Type extension	Description	Allocation Policy
GATEWAY	TBD7	0	Specifies that a given network address is reached via a gateway on the originating router, with value equal to the number of hops	
GATEWAY	TBD7	1-255		Expert Review

Table 16: Address Block TLV Type assignment: GATEWAY

Type extensions indicated as Expert Review SHOULD be allocated as described in [RFC5444], based on Expert Review as defined in [RFC5226].

24.6. NBR_ADDR_TYPE and MPR Values

Note: This section does not require any IANA action, as the required information is included in the descriptions of the MPR and NBR_ADDR_TYPE Address Block TLVs allocated in Section 24.5. This information is recorded here for clarity, and for use elsewhere in this specification.

The Values which the MPR Address Block TLV can use are the following:

- FLOODING := 1;

- ROUTING := 2;

- FLOOD_ROUTE := 3.

The Values which the NBR_ADDR_TYPE Address Block TLV can use are the following:

- ORIGINATOR := 1;

- ROUTABLE := 2;

- ROUTABLE_ORIG := 3.

25. Contributors

This specification is the result of the joint efforts of the following contributors -- listed alphabetically.

- Cedric Adjih, INRIA, France, <Cedric.Adjih@inria.fr>

- Emmanuel Baccelli, INRIA , France, <Emmanuel.Baccelli@inria.fr>

- Thomas Heide Clausen, LIX, France, <T.Clausen@computer.org>

- Justin Dean, NRL, USA, <jdean@itd.nrl.navy.mil>

- Christopher Dearlove, BAE Systems, UK, <chris.dearlove@baesystems.com>

- Satoh Hiroki, Hitachi SDL, Japan, <hiroki.satoh.yj@hitachi.com>

- Philippe Jacquet, INRIA, France, <Philippe.Jacquet@inria.fr>

- Monden Kazuya, Hitachi SDL, Japan, <kazuya.monden.vw@hitachi.com>

- Kenichi Mase, Niigata University, Japan, <mase@ie.niigata-u.ac.jp>

- Ryuji Wakikawa, Toyota, Japan, <ryuji@sfc.wide.ad.jp>

26. Acknowledgments

The authors would like to acknowledge the team behind OLSRv1, specified in RFC3626, including Anis Laouiti (INT, Paris), Pascale Minet (INRIA, France), Laurent Viennot (INRIA, France), and Amir Qayyum (M.A. Jinnah University, Islamabad) for their contributions.

The authors would like to gratefully acknowledge the following people for intense technical discussions, early reviews and comments on the specification and its components (listed alphabetically): Khaldoun Al Agha (LRI), Teco Boot (Infinity Networks), Song-Yean Cho (LIX), Alan Cullen (BAE Systems), Ulrich Herberg (Fujitsu), Louise Lamont (CRC), Li Li (CRC), Joe Macker (NRL), Richard Ogier (SRI), Charles E. Perkins (WiChorus), Henning Rogge (FGAN), and the entire IETF MANET working group.

27. References

27.1. Normative References

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

[RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)", RFC 5148, February 2008.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, BCP 26, May 2008.

[RFC5444] Clausen, T., Dean, J., Dearlove, C., and C. Adjih, "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, February 2009.

[RFC5497] Clausen, T. and C. Dearlove, "Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)", RFC 5497, March 2009.

[RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, March 2009.

[RFC6130] Clausen, T., Dean, J., and C. Dearlove, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

27.2. Informative References

[RFC2501] Macker, J. and S. Corson, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, January 1999.

[RFC3626] Clausen, T. and P. Jacquet, "The Optimized Link State Routing Protocol", RFC 3626, October 2003.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., and R. Thayer, "OpenPGP message format", RFC 4880, November 2007.

[HIPERLAN] ETSI, "ETSI STC-RES10 Committee. Radio equipment and systems: HIPERLAN type 1, functional specifications ETS 300-652", June 1996.

[HIPERLAN2] Jacquet, P., Minet, P., Muhlethaler, P., and N. Rivierre, "Increasing reliability in cable free radio LANs: Low level forwarding in HIPERLAN.", 1996.

[MPR] Qayyum, A., Viennot, L., and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks.", 2001.

[FSR] Pei, G., Gerla, M., and T. Chen, "Fisheye state routing in mobile ad hoc networks", 2000.

[FSLS] Santivanez, C., Ramanathan, R., and I. Stavrakakis, "Making link-state routing scale for ad hoc networks", 2000.

Appendix A. Example Algorithm for Calculating MPRs

The following specifies an algorithm which MAY be used to select an MPR Set given a Neighbor Graph, as defined in Section 18.2 and Section 18.3.

This algorithm selects an MPR Set M that is a subset of the set N1 that is part of the Neighbor Graph. This algorithm assumes that a subset I of N1 is pre-selected as MPRs, i.e., that M will contain I.

A.1. Additional Notation

The following additional notation, in addition to that in Section 18.2 will be used by this algorithm:

N:
A subset of N2, consisting of those elements y in N2 such that either d1(y) is not defined, or there is at least one x in N1 such that d(x,y) is defined and d(x,y) < d1(y).

D(x):
For an element x in N1, the number of elements y in N for which d(x,y) is defined and has minimal value among the d(z,y) for all z in N1.

R(x,M):
For an element x in N1, the number of elements y in N for which d(x,y) is defined, has minimal value among the d(z,y) for all z in N1, and no such minimal values have z in M. (Note that, denoting the empty set by 0, D(x) = R(x,0).)

A.2. MPR Selection Algorithm

To create the MPR Set M, starting with M := I:

1. Add all elements x in N1 that have W(x) = WILL_ALWAYS.

2. For each element y in N for which there is only one element x in N1 such that d2(x,y) is defined, add that element x to M.

3. While there exists any element x in N1 with R(x,M) > 0:

    1. Select an element x in N1 with R(x,M) > 0 in the following order of priority:

        + greatest W(x), THEN;

        + greatest R(x,M), THEN;

        + greatest D(x), THEN;

        + any choice, which MAY be based on other criteria (for example a router MAY choose to prefer a neighbor as an MPR if that neighbor has already selected the router as an MPR of the same type, MAY prefer a neighbor based on information freshness, or MAY prefer a neighbor based on length of time previously selected as an MPR) or MAY be random.

4. OPTIONALLY, consider each element x in M, but not in I, in turn and if x can be removed from M while still leaving it satisfying the definition of an MPR Set, then remove that element x from M. Elements MAY be considered in any order, e.g. in order of

increasing W(x).

Appendix B. Example Algorithm for Calculating the Routing Set

The following procedure is given as an example for calculating the Routing Set using a variation of Dijkstra's algorithm. First all Routing Tuples are removed, and then, using the selections and definitions in Appendix B.1, the procedures in the following sections (each considered a "stage" of the processing) are applied in turn.

B.1. Local Interfaces and Neighbors

The following selections and definitions are made:

1. For each Local Interface Tuple, select a network address from its I_local_iface_addr_list, this is defined as the selected address for this Local Interface Tuple.

2. For each Link Tuple, the selected address of its corresponding Local Interface Tuple is defined as the selected local address for this Local Interface Tuple.

3. For each Neighbor Tuple with N_symmetric = true, select a Link Tuple with L_status = SYMMETRIC for which this is the corresponding Neighbor Tuple and has L_out_metric = N_out_metric. This is defined as the selected Link Tuple for this Neighbor Tuple.

4. For each network address (N_orig_addr or in N_neighbor_addr_list, the "neighbor address") from a Neighbor Tuple with N_symmetric = true, select a Link Tuple (the "selected Link Tuple") from those for which this is the corresponding Neighbor Tuple, have L_status = SYMMETRIC, and have L_out_metric = N_out_metric, by:

    1. If there is such a Link Tuple whose L_neighbor_iface_addr_list contains the neighbor address, select that Link Tuple.

    2. Otherwise select the selected Link Tuple for this Neighbor Tuple.

    Then for this neighbor address:

    3. The selected local address is defined as the selected local address for the selected Link Tuple.

    4. The selected link address is defined as an address from the L_neighbor_iface_addr_list of the selected Link Tuple, if

possible equal to this neighbor address.

5. Routing Tuple preference is decided by preference for minimum R_dist, then for minimum R_dist, and then for preference for corresponding Neighbor Tuples in this order:

   * For greater N_will_routing.

   * For N_mpr_selector = true over N_mpr_selector = false.

   Note that preferred Routing Tuples SHOULD be used. Routing Tuples with minimum R_metric MUST be used, this is specified outside the definition of preference. An implementation MAY modify this definition of preference (including for minimum R_dist) without otherwise affecting this algorithm.

B.2. Add Neighbor Routers

The following procedure is executed once.

1. For each Neighbor Tuple with N_symmetric = true, add a Routing Tuple with:

   * R_dest_addr := N_orig_addr;

   * R_next_iface_addr := selected link address for N_orig_addr;

   * R_local_iface_addr := selected local address for N_orig_addr;

   * R_metric := N_out_metric;

   * R_dist := 1.

B.3. Add Remote Routers

The following procedure is executed once.

1. Add a label that may be "used" or "unused" to each Routing Tuple, with all initial values equal to unused. (Note that this label is only required during this algorithm.)

2. If there are no unused Routing Tuples then this stage is complete, otherwise repeat the following until that is the case.

   1. Find the unused Routing Tuple with minimum R_metric (if more than one, pick any) and denote it the "current Routing Tuple".

2. Mark the current Routing Tuple as used.

3. For each Router Topology Tuple, with:

    + TR_from_orig_addr = R_dest_addr of the current Routing Tuple.

    2. Define:

        - new_metric := R_metric of the current Routing Tuple + TR_metric;

        - new_dist := R_dist of the current Routing Tuple + 1.

    3. If there is no Routing Tuple with R_dest_addr = TR_to_orig_addr, then create an unused Routing Tuple with:

        - R_dest_addr := TR_to_orig_addr;

        - R_next_iface_addr := R_next_iface_addr of the current Routing Tuple;

        - R_local_iface_addr := R_local_iface_addr of the current Routing Tuple;

        - R_metric := new_metric;

        - R_dist := new_dist.

    4. Otherwise, if there is an unused Routing Tuple with R_dest_addr = TR_to_orig_addr, and either new_metric < R_metric or (new_metric = R_metric and the updated Routing Tuple would be preferred) then update this Routing Tuple to have:

        - R_next_iface_addr := R_next_iface_addr of the current Routing Tuple;

        - R_local_iface_addr := R_local_iface_addr of the current Routing Tuple;

        - R_metric := new_metric;

        - R_dist := new_dist.

B.4. Add Neighbor Addresses

The following procedure is executed once.

1. For each Neighbor Tuple with N_symmetric = true:

    1. For each network address (the "neighbor address") in N_neighbor_addr_list, if the neighbor address is not equal to the R_dest_addr of any Routing Tuple, then add a new Routing Tuple, with:

        + R_dest_addr := neighbor address;

        + R_next_iface_addr := selected link address for the neighbor address;

        + R_local_iface_addr := selected local address for the neighbor address;

        + R_metric := N_out_metric;

        + R_dist := 1.

B.5. Add Remote Routable Addresses

The following procedure is executed once.

1. For each Routable Address Topology Tuple, if:

    * TA_dest_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;

    * TA_from_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple"),

    then add a new Routing Tuple, with:

    * R_dest_addr := TA_dest_addr;

    * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;

    * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;

    * R_metric := R_metric of the previous Routing Tuple + TA_metric.

- R_dist := R_dist of the previous Routing Tuple + 1.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

B.6. Add Attached Networks

The following procedure is executed once.

1. For each Attached Network Tuple, if:

   - AN_net_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;

   - AN_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple),

   then add a new Routing Tuple, with:

   - R_dest_addr := AN_net_addr;

   - R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;

   - R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;

   - R_metric := R_metric of the previous Routing Tuple + AN_metric;

   - R_dist := R_dist of the previous Routing Tuple + AN_dist.

   There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

B.7. Add 2-Hop Neighbors

The following procedure is executed once.

1. For each 2-Hop Tuple, if:

- N2_2hop_addr is a routable address, AND;
- N2_2hop_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage, AND;
- the Routing Tuple with R_dest_addr = N_orig_addr of the corresponding Neighbor Tuple (the "previous Routing Tuple") has R_dist = 1,

then add a new Routing Tuple, with:

- R_dest_addr := N2_2hop_addr;
- R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;
- R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;
- R_metric := R_metric of the previous Routing Tuple + N_out_metric of the corresponding Neighbor Tuple;
- R_dist := 2.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

Appendix C. TC Message Example

TC messages are instances of [RFC5444] messages. This specification requires that TC messages contains <msg-hop-limit> and <msg-orig-addr> fields. It supports TC messages with any combination of remaining message header options and address encodings, enabled by [RFC5444] that convey the required information. As a consequence, there is no single way to represent how all TC messages look. This appendix illustrates a TC message, the exact values and content included are explained in the following text.

The TC message’s four bit Message Flags (MF) field has value 15 indicating that the message header contains originator address, hop limit, hop count, and message sequence number fields. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 71 octets.

The message has a Message TLV Block with content length 13 octets containing three TLVs. The first two TLVs are validity and interval times for the message. The third TLV is the content sequence number TLV used to carry the 2 octet ANSN, and (with default type extension zero, i.e., COMPLETE) indicating that the TC message is complete. Each TLV uses a TLV with Flags octet (MTLVF) value 16, indicating that it has a Value, but no type extension or start and stop indexes. The first two TLVs have a Value Length of 1 octet, the last has a Value Length of 2 octets.

The message has two Address Blocks. (This is not necessary, the information could be conveyed using a single Address Block, the use of two Address Blocks, which is also allowed, is illustrative only.) The first Address Block contains 3 addresses, with Flags octet (ATLVF) value 128, hence with a Head section (with length 2 octets), but no Tail section, and hence with Mid sections with length two octets. The following TLV Block (content length 13 octets) contains two TLVs. The first TLV is a NBR_ADDR_TYPE TLV with Flags octet (ATLVF) value 16, indicating a single Value but no indexes. Thus all three addresses are associated with the Value (with Value Length 1 octet) ROUTABLE_ORIG, i.e., they are originator addresses of advertised neighbors that are also routable addresses. The second TLV is a LINK_STATUS TLV with Flags octet (ATLVF) value 20, indicating a Value for each address, i.e. as the total Value Length is 6 octets, each address is associated with a Value with length two octets. These Value fields are each shown as having four bits indicating that they are outgoing neighbor metric values, and as having twelve bits that represent the metric value (the first four bits being the exponent, the remaining twelve bits the mantissa).

The second Address Block contains 1 address, with Flags octet (ATLVF) 176, indicating that there is a Head section (with length 2 octets), that the Tail section (with length 2 octets) consists of zero valued octets (not included), and that there is a single prefix length, which is 16. The network address is thus Head.0.0/16. The following TLV Block (content length 8 octets) includes two TLVs. The first has a Flags octet (ATLVF) of 16, again indicating that no indexes are needed, but that a Value (with Value Length 1 octet) is present, indicating the address distance as a number of hops. The second TLV is another LINK_METRIC TLV, as in the first Address TLV Block except with a Flags octet (ATLVF) value 16, indicating that a single Value is present.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |      TC       | MF=15 | MAL=3 |      Message Length = 71      |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Originator Address                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |   Hop Limit   |   Hop Count   |    Message Sequence Number    |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Message TLV Block Length = 13 | VALIDITY_TIME |  MTLVF = 16   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Value Len = 1 | Value (Time)  | INTERVAL_TIME |  MTLVF = 16   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Value Len = 1 | Value (Time)  | CONT_SEQ_NUM  |  MTLVF = 16   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Value Len = 2 |         Value (ANSN)          | Num Addrs = 3 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |   ABF = 128   | Head Len = 2  |             Head              |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |              Mid              |              Mid              |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |              Mid              | Address TLV Block Length = 13 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | NBR_ADDR_TYPE |  ATLVF = 16   | Value Len = 1 | ROUTABLE_ORIG |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |  LINK_METRIC  |  ATLVF = 20   | Value Len = 6 |0|0|0|1|Metric |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Metric (cont) |0|0|0|1|        Metric         |0|0|0|1|Metric |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Metric (cont) | Num Addrs = 1 |   ABF = 176   | Head Len = 2  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |             Head              | Tail Len = 2  | Pref Len = 16 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Address TLV Block Length = 9  |    GATEWAY    |  ATLVF = 16   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Value Len = 1 | Value (Hops)  |  LINK_METRIC  |  ATLVF = 16   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Value Len = 2 |0|0|0|1|        Metric         |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Appendix D. Constraints

Any process which updates the Local Information Base, the Neighborhood Information Base or the Topology Information Base MUST ensure that all constraints specified in this appendix are maintained, as well as those specified in [RFC6130].

In each Originator Tuple:

- O_orig_addr MUST NOT equal any other O_orig_addr.

- O_orig_addr MUST NOT equal this router's originator address.

In each Local Attached Network Tuple:

- AL_net_addr MUST NOT equal any other AL_net_addr.

- AL_net_addr MUST NOT equal or be a sub-range of any network address in the I_local_iface_addr_list of any Local Interface Tuple.

- AL_net_addr MUST NOT equal this router's originator address, or equal the O_orig_addr in any Originator Tuple.

- AL_dist MUST NOT be less than zero.

In each Link Tuple:

- L_neighbor_iface_addr_list MUST NOT contain any network address that AL_net_addr of any Local Attached Network Tuple equals or is a sub-range of.

- If L_status = HEARD or L_status = SYMMETRIC then L_in_metric != UNKNOWN_METRIC.

- If L_status = SYMMETRIC then L_in_metric != UNKNOWN_METRIC.

- if L_in_metric != UNKNOWN_METRIC then L_in_metric MUST be representable in the defined compressed form.

- if L_out_metric != UNKNOWN_METRIC then L_out_metric MUST be representable in the defined compressed form.

- If L_mpr_selector = true, then L_status = SYMMETRIC.

In each Neighbor Tuple:

- N_orig_addr MUST NOT be changed to unknown.

- N_orig_addr MUST NOT equal this router's originator address, or equal O_orig_addr in any Originator Tuple.

- N_orig_addr MUST NOT equal the AL_net_addr in any Local Attached Network Tuple.

- If N_orig_addr != unknown, then N_orig_addr MUST NOT equal the N_orig_addr in any other Neighbor Tuple.

- N_neighbor_addr_list MUST NOT contain any network address which includes this router's originator address, the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

- If N_orig_addr = unknown, then N_will_flooding = WILL_NEVER, N_will_routing = WILL_NEVER, N_flooding_mpr, N_routing_mpr = false, N_mpr_selector = false, and N_advertised = false.

- N_in_metric MUST equal the minimum value of the L_in_metric values of all corresponding Link Tuples, if any, otherwise N_in_metric = UNKNOWN_METRIC.

- N_out_metric MUST equal the minimum value of the L_out_metric values of all corresponding Link Tuples, if any, otherwise N_out_metric = UNKNOWN_METRIC.

- N_will_flooding and N_will_routing MUST be in the range from WILL_NEVER to WILL_ALWAYS, inclusive.

- If N_flooding_mpr = true, then N_symmetric MUST be true and N_will_flooding MUST NOT equal WILL_NEVER.

- If N_routing_mpr = true, then N_symmetric MUST be true and N_will_routing MUST NOT equal WILL_NEVER.

- If N_symmetric = true and N_flooding_mpr = false, then N_will_flooding MUST NOT equal WILL_ALWAYS.

- If N_symmetric = true and N_routing_mpr = false, then N_will_routing MUST NOT equal WILL_ALWAYS.

- If N_mpr_selector = true, then N_advertised MUST be true.

- If N_advertised = true, then N_symmetric MUST be true.

In each Lost Neighbor Tuple:

- NL_neighbor_addr MUST NOT include this router's originator address, the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

In each 2-Hop Tuple:

- N2_2hop_addr MUST NOT equal this router's originator address, equal the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple

- N2_in_metric and N2_out_metric MUST be representable in the defined compressed form.

In each Advertising Remote Router Tuple:

- AR_orig_addr MUST NOT be in any network address in the I_local_iface_addr_list in any Local Interface Tuple or be in the IR_local_iface_addr in any Removed Interface Address Tuple.

- AR_orig_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.

- AR_orig_addr MUST NOT be in the AL_net_addr in any Local Attached Network Tuple.

- AR_orig_addr MUST NOT equal the AR_orig_addr in any other Advertising Remote Router Tuple.

In each Router Topology Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = TR_from_orig_addr.

- TR_to_orig_addr MUST NOT be in any network address in the I_local_iface_addr_list in any Local Interface Tuple or be in the IR_local_iface_addr in any Removed Interface Address Tuple.

- TR_to_orig_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.

- TR_to_orig_addr MUST NOT be in the AL_net_addr in any Local Attached Network Tuple.

- The ordered pair (TR_from_orig_addr, TR_to_orig_addr) MUST NOT equal the corresponding pair for any other Router Topology Tuple.

- TR_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = TR_from_orig_addr.

- TR_metric MUST be representable in the defined compressed form.

In each Routable Address Topology Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = TA_from_orig_addr.

- TA_dest_addr MUST be routable.

- TA_dest_addr MUST NOT overlap any network address in the I_local_iface_addr_list in any Local Interface Tuple or overlap the IR_local_iface_addr in any Removed Interface Address Tuple.

- TA_dest_addr MUST NOT include this router's originator address or include the O_orig_addr in any Originator Tuple.

- TA_dest_addr MUST NOT equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

- The ordered pair (TA_from_orig_addr, TA_dest_addr) MUST NOT equal the corresponding pair for any other Attached Network Tuple.

- TA_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = TA_from_orig_addr.

- TA_metric MUST be representable in the defined compressed form.

In each Attached Network Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = AN_orig_addr.

- AN_net_addr MUST NOT equal or be a sub-range of any network address in the I_local_iface_addr_list in any Local Interface Tuple or be a sub-range of the IR_local_iface_addr in any Removed Interface Address Tuple.

- AN_net_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.

- AN_net_addr MUST NOT equal the AL_net_addr in any Local Attached Network Tuple.

- The ordered pair (AN_orig_addr, AN_net_addr) MUST NOT equal the corresponding pair for any other Attached Network Tuple.

- AN_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = AN_orig_addr.

- AN_dist MUST NOT be less than zero.

- AN_metric MUST be representable in the defined compressed form.

Appendix E. Flow and Congestion Control

Due to its proactive nature, this protocol has a natural control over the flow of its control traffic. Routers transmit control messages at predetermined rates specified and bounded by message intervals.

This protocol employs [RFC6130] for local signaling, embedding MPR selection advertisement through a simple Address Block TLV, and router willingness advertisement (if any) as a single Message TLV. Local signaling, therefore, shares the characteristics and constraints of [RFC6130].

Furthermore, the use of MPRs can greatly reduce the signaling overhead from link state information dissemination in two ways, attaining both flooding reduction and topology reduction. First, using MPR flooding, the cost of distributing link state information throughout the network is reduced, as compared to when using classic flooding, since only MPRs need to forward link state declaration messages. Second, the amount of link state information for a router to declare is reduced to need only contain that router's MPR selectors. This reduces the size of a link state declaration as compared to declaring full link state information. In particular some routers may not need to declare any such information. In dense networks, the reduction of control traffic can be of several orders of magnitude compared to routing protocols using classical flooding [MPR]. This feature naturally provides more bandwidth for useful data traffic and pushes further the frontier of congestion.

Since the control traffic is continuous and periodic, it keeps the quality of the links used in routing more stable. However, using some options, some control messages (HELLO messages or TC messages) may be intentionally sent in advance of their deadline in order to increase the responsiveness of the protocol to topology changes. This may cause a small, temporary, and local increase of control traffic, however this is at all times bounded by the use of minimum message intervals.

Authors' Addresses

Thomas Heide Clausen
LIX, Ecole Polytechnique

Phone: +33 6 6058 9349
EMail: T.Clausen@computer.org
URI: http://www.ThomasClausen.org/

Christopher Dearlove
BAE Systems ATC

Phone: +44 1245 242194
EMail: chris.dearlove@baesystems.com
URI: http://www.baesystems.com/

Philippe Jacquet
Project Hipercom, INRIA

Phone: +33 1 3963 5263
EMail: philippe.jacquet@inria.fr

Mobile Ad hoc Networking (MANET)   U. Herberg
Internet-Draft   Fujitsu Laboratories of America
Intended status: Standards Track   T. Clausen
Expires: March 9, 2012   LIX, Ecole Polytechnique
                                                       September 6, 2011

MANET Cryptographical Signature TLV Definition
draft-ietf-manet-packetbb-sec-06

Abstract

This document describes general and flexible TLVs (type-length-value structure) for representing cryptographic signatures as well as timestamps, using the generalized MANET packet/message format [RFC5444]. It defines two Packet TLVs, two Message TLVs, and two Address Block TLVs, for affixing cryptographic signatures and timestamps to a packet, message and address, respectively.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies:

- two TLVs for carrying cryptographic signatures and timestamps in packets, messages, and address blocks as defined by [RFC5444],

- a generic framework for calculating cryptographic signatures, accounting (for Message TLVs) for mutable message header fields (<msg-hop-limit> and <msg-hop-count>), where these fields are present in messages.

This document requests from IANA:

- allocations for these Packet, Message, and Address Block TLVs from the 0-223 Packet TLV range, the 0-127 Message TLV range and the 0-127 Address Block TLV range from [RFC5444],

- creation of two IANA registries for recording code points for hash function and signature calculation, respectively.

Finally, this document defines, in Section 12:

- one common method for generating signatures as a cryptographic function, calculated over the hash value of the content to be signed.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document uses the terminology and notation defined in [RFC5444]. In particular, the following TLV fields from [RFC5444] are used in this specification:

<msg-hop-limit> - hop limit of a message, as specified in Section 5.2 of [RFC5444].

<msg-hop-count> - hop count of a message, as specified in Section 5.2 of [RFC5444].

<length> - length of a TLV in octets, as specified in Section 5.4.1 of [RFC5444].

3. Applicability Statement

MANET routing protocols using the format defined in [RFC5444] are accorded the ability to carry additional information in control messages and packets, through inclusion of TLVs. Information so included MAY be used by a MANET routing protocol, or by an extension of a MANET routing protocol, according to its specification.

This document specifies how to include a cryptographic signature for a packet, a message, and addresses in address blocks within a message, by way of such TLVs. This document also specifies how to treat "mutable" fields, specifically the <msg-hop-count> and <msg-hop-limit> fields, if present in the message header when calculating signatures, such that the resulting signature can be correctly verified by any recipient, and how to include this signature.

This document describes a generic framework for creating signatures, and how to include these signatures in TLVs. In Section 12, an example method for calculating such signatures is given, using a cryptographic function over the hash value of the content to be signed.

4. Security Architecture

Basic MANET routing protocol specifications are often "oblivious to security", however have a clause allowing a control message to be rejected as "badly formed" prior to it being processed or forwarded. MANET routing protocols such as [RFC6130] and [OLSRv2] recognize external reasons (such as failure to verify a signature) for rejecting a message as "badly formed", and therefore "invalid for processing". This architecture is a result of the observation that with respect to security in MANETs, "one size rarely fits all" and that MANET routing protocol deployment domains have varying security requirements ranging from "unbreakable" to "virtually none". The virtue of this approach is that MANET routing protocol specifications (and implementations) can remain "generic", with extensions providing proper deployment-domain specific security mechanisms.

The MANET routing protocol "security architecture", in which this specification situates itself, can therefore be summarized as follows:

- o Security-oblivious MANET routing protocol specifications, with a clause allowing an extension to reject a message (prior to processing/forwarding) as "badly formed".

- o MANET routing protocol security extensions, rejecting messages as "badly formed", as appropriate for a given deployment-domain specific security requirement.

- o Code-points and an exchange format for information, necessary for specification of such MANET routing protocol security extensions.

This document addresses the last of these issues, by specifying a common exchange format for cryptographic signatures, making reservations from within the Packet TLV, Message TLV, and Address Block TLV registries of [RFC5444], to be used (and shared) among MANET routing protocol security extensions.

For the specific decomposition of a signature into a cryptographic function over a hash value, specified in Section 12, this document establishes two IANA registries for code-points for hash functions and cryptographic functions adhering to [RFC5444].

With respect to [RFC5444], this document:

- o is intended to be used in the non-normative, but intended, mode of use described in Appendix B of [RFC5444].

- o is a specific example of the Security Considerations section of [RFC5444] (the authentication part).

5. Overview and Functioning

This document specifies a syntactical representation of security related information for use with [RFC5444] addresses, messages, and packets, as well as establishes IANA registrations and registries.

Moreover, this document provides guidelines how MANET routing protocols and MANET routing protocol extensions, using this specification, should treat Signature and Timestamp TLVs, and mutable fields in messages. This specification does not represent a stand-alone protocol; MANET routing protocols and MANET routing protocol extensions, using this specification, MUST provide instructions as to how to handle packets, messages and addresses with security information, associated as specified in this document.

This document requests assignment of TLV types from the registries defined for Packet, Message and Address Block TLVs in [RFC5444].

When a TLV type is assigned from one of these registries, a registry for "Type Extensions" for that TLV type is created by IANA. This document utilizes these "Type Extension" registries so created, in order to specify internal structure (and accompanying processing) of the <value> field of a TLV.

For example, and as defined in this document, a SIGNATURE TLV with Type Extension = 0 specifies that the <value> field has no pre-defined internal structure, but is simply a sequence of octets. A SIGNATURE TLV with Type Extension = 1 specifies that the <value> field has a pre-defined internal structure, and defines its interpretation (specifically, the <value> field consists of a cryptographic operation over a hash value, with fields indicating which hash function and cryptographic operation has been used, specified in Section 12).

Other documents may request assignments for other Type Extensions, and must if so specify their internal structure (if any) and interpretation.

6. General Signature TLV Structure

The value of the Signature TLV is:

```
          <value> := <signature-value>
```

where:

<signature-value> is a field, of <length> octets, which contains the information, to be interpreted by the signature verification process, as specified by the Type Extension.

Note that this does not stipulate how to calculate the <signature-value>, nor the internal structure hereof, if any; such MUST be specified by way of the Type Extension for the SIGNATURE TLV type, see Section 13. This document specifies two such type-extensions, for signatures without pre-defined structures, and for signatures constructed by way of a cryptographic operation over a hash-value.

7. General Timestamp TLV Structure

The value of the Timestamp TLV is:

```
       <value> := <time-value>
```

where:

<time-value> is an unsigned integer field, of length <length>, which contains the timestamp.

Note that this does not stipulate how to calculate the <time-value>, nor the internal structure hereof, if any; such MUST be specified by way of the Type Extension for the TIMESTAMP TLV type, see Section 13.

A timestamp is essentially "freshness information". As such, its setting and interpretation is to be determined by the MANET routing protocol, or MANET routing protocol extension, that uses the timestamp, and may, e.g., correspond to a UNIX-timestamp, GPS timestamp or a simple sequence number.

8. Packet TLVs

Two Packet TLVs are defined, for including the cryptographic signature of a packet, and for including the timestamp indicating the time at which the cryptographic signature was calculated.

8.1. Packet SIGNATURE TLV

A Packet SIGNATURE TLV is an example of a Signature TLV as described in Section 6.

The following considerations apply:

- As packets defined in [RFC5444] are never forwarded by routers, no special considerations are required regarding mutable fields (e.g. <msg-hop-count> and <msg-hop-limit>), if present, when calculating the signature.

- Any Packet SIGNATURE TLVs already present in the Packet TLV block MUST be removed before calculating the signature, and the Packet TLV block size MUST be recalculated accordingly. The TLVs can be restored after having calculated the signature value.

The rationale for removing any Packet SIGNATURE TLV already present prior to calculating the signature is that several signatures may be added to the same packet, e.g., using different signature functions.

8.2. Packet TIMESTAMP TLV

A Packet TIMESTAMP TLV is an example of a Timestamp TLV as described in Section 7. If a packet contains a TIMESTAMP TLV and a SIGNATURE TLV, the TIMESTAMP TLV SHOULD be added to the packet before any SIGNATURE TLV, in order that it be included in the calculation of the signature.

9. Message TLVs

Two Message TLVs are defined, for including the cryptographic signature of a message, and for including the timestamp indicating the time at which the cryptographic signature was calculated.

9.1. Message SIGNATURE TLV

A Message SIGNATURE TLV is an example of a Signature TLV as described in Section 6. When determining the <signature-value> for a message, the following considerations must be applied:

- The fields <msg-hop-limit> and <msg-hop-count>, if present, MUST both be assumed to have the value 0 (zero) when calculating the signature.

- Any Message SIGNATURE TLVs already present in the Message TLV block MUST be removed before calculating the signature, and the message size as well as the Message TLV block size MUST be recalculated accordingly. Removed SIGNATURE TLVs SHOULD be restored after having calculated the signature value.

The rationale for removing any Message SIGNATURE TLV already present prior to calculating the signature is that several signatures may be added to the same message, e.g., using different signature functions.

9.2. Message TIMESTAMP TLV

A Message TIMESTAMP TLV is an example of a Timestamp TLV as described in Section 7. If a message contains a TIMESTAMP TLV and a SIGNATURE TLV, the TIMESTAMP TLV SHOULD be added to the message before the SIGNATURE TLV, in order that it be included in the calculation of the signature.

10. Address Block TLVs

Two Address Block TLVs are defined, for associating a cryptographic signature to an address, and for including the timestamp indicating the time at which the cryptographic signature was calculated.

10.1. Address Block SIGNATURE TLV

An Address Block SIGNATURE TLV is an example of a Signature TLV as described in Section 6. The signature is calculated over the address, concatenated with any other values, for example, any other TLV value that is associated with that address. A MANET routing protocol or MANET routing protocol extension using Address Block SIGNATURE TLVs MUST specify how to include any such concatenated attribute of the address in the verification process of the signature.

10.2. Address Block TIMESTAMP TLV

An Address Block TIMESTAMP TLV is an example of a Timestamp TLV as described in Section 7. If both a TIMESTAMP TLV and a SIGNATURE TLV are associated with an address, the timestamp value should be considered when calculating the value of the signature.

11. Signature: Basic

The basic signature proposed, represented by way of a SIGNATURE TLV with Type Extension = 0, is a simple bit-field containing the cryptographic signature. This assumes that the mechanism stipulating how signatures are calculated and verified is established outside of this specification, e.g., by way of administrative configuration or external out-of-band signaling. Thus, the <signature-value> for when using Type Extension = 0 is:

```
          <signature-value> := <signature-data>
```

where:

<signature-data> is an unsigned integer field, of length <length>, which contains the cryptographic signature.

12. Signature: Cryptographic Function over a Hash Value

One common way of calculating a signature is applying a cryptographic function on a hash value of the content. This decomposition is specified in the following, using a Type Extension = 1 in the Signature TLVs.

12.1. General Signature TLV Structure

The following data structure allows representation of a cryptographic signature, including specification of the appropriate hash function and cryptographic function used for calculating the signature:

```
            <signature-value> := <hash-function>
                                 <cryptographic-function>
                                 <key-index>
                                 <signature-data>
```

where:

<hash-function> is an 8-bit unsigned integer field specifying the hash function.

<cryptographic-function> is an 8-bit unsigned integer field specifying the cryptographic function.

<key-index> is an 8-bit unsigned integer field specifying the key index of the key which was used to sign the message, which allows unique identification of different keys with the same originator. It is the responsibility of each key originator to make sure that actively used keys that it issues have distinct key indices and that all key indices have a value not equal to 0x00. The value 0x00 is reserved for a pre-installed, shared key.

<signature-data> is an unsigned integer field, whose length is <length> - 3, and which contains the cryptographic signature.

The version of this TLV, specified in this section, assumes that calculating the signature can be decomposed into:

```
signature-value = cryptographic-function(hash-function(content))
```

The hash function and the cryptographic function correspond to the entries in two IANA registries, set up by this specification in Section 13.

12.1.1. Rationale

The rationale for separating the hash function and the cryptographic function into two octets instead of having all combinations in a single octet - possibly as TLV type extension - is twofold: First, if further hash functions or cryptographic functions are added in the future, the number space might not remain continuous. More importantly, the number space of possible combinations would be rapidly exhausted. As new or improved cryptographic mechanism are continuously being developed and introduced, this format should be able to accommodate such for the foreseeable future.

The rationale for not including a field that lists parameters of the cryptographic signature in the TLV is, that before being able to validate a cryptographic signature, routers have to exchange or acquire keys (e.g. public keys). Any additional parameters can be provided together with the keys in that bootstrap process. It is therefore not necessary, and would even entail an extra overhead, to transmit the parameters within every message. One implicitly available parameter is the length of the signature, which is <length> - 3, and which depends on the choice of the cryptographic function.

12.2. Considerations for Calculating the Signature

In the following, considerations are listed, which MUST be applied when calculating the signature for Packet, Message and Address SIGNATURE TLVs, respectively.

12.2.1. Packet SIGNATURE TLV

When determining the <signature-value> for a Packet, the signature is calculated over the three fields <hash-function>, <cryptographic-function> and <key-index> (in that order), concatenated with the entire Packet, including the packet header, all Packet TLVs (other than Packet SIGNATURE TLVs) and all included Messages and their message headers, in accordance with Section 8.1.

12.2.2. Message SIGNATURE TLV

When determining the <signature-value> for a message, the signature is calculated over the three fields <hash-function>, <cryptographic-function>, and <key-index> (in that order), concatenated with the entire message. The considerations in Section 9.1 MUST be applied.

12.2.3. Address Block SIGNATURE TLV

When determining the <signature-value> for an address, the signature is calculated over the three fields <hash-function>, <cryptographic-function>, and <key-index> (in that order), concatenated with the address, concatenated with any other values, for example, any other TLV value that is associated with that address. A MANET routing protocol or MANET routing protocol extension using Address Block SIGNATURE TLVs MUST specify how to include any such concatenated attribute of the address in the verification process of the signature. The considerations in Section 10.2 MUST be applied.

12.3. Example of a Signed Message

The sample message depicted in Figure 1 is derived from appendix D of [RFC5444]. The message contains a SIGNATURE Message TLV, with the value representing a 16 octet long signature of the whole message. The type extension of the Message TLV is 1, for the specific decomposition of a signature into a cryptographic function over a hash value, as specified in Section 12.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | PV=0  | PF=8  |    Packet Sequence Number     | Message Type  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | MF=15 | MAL=3 |       Message Length = 40     | Msg. Orig Addr|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |       Message Originator Address (cont)       |   Hop Limit   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |   Hop Count   |    Message Sequence Number    | Msg. TLV Block|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Length = 30   |   SIGNATURE   |  MTLVF = 144  |  MTLVExt = 1  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |Value Len = 19 |   Hash Func   |  Crypto Func  |   Key Index   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                        Signature Value                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                        Signature Value (cont)                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                        Signature Value (cont)                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                        Signature Value (cont)                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Example message with signature

13. IANA Considerations

This specification defines:

- two Packet TLV types, which MUST be allocated from the 0-223 range of the "Assigned Packet TLV Types" repository of [RFC5444] as specified in Table 1,

- two Message TLV types, which MUST be allocated from the 0-127 range of the "Assigned Message TLV Types" repository of [RFC5444] as specified in Table 2,

- two Address Block TLV types, which MUST be allocated from the 0-127 range of the "Assigned Address Block TLV Types" repository of [RFC5444] as specified in Table 3.

This specification requests:

- o creation of type extension registries for these TLV types with initial values as in Table 1 to Table 3.

IANA is requested to assign the same numerical value to the Packet TLV, Message TLV and Address Block TLV types with the same name.

The following terms are used with the meanings defined in [BCP26]: "Namespace", "Assigned Value", "Registration", "Unassigned", "Reserved", "Hierarchical Allocation", and "Designated Expert".

The following policies are used with the meanings defined in [BCP26]: "Private Use", "Expert Review", and "Standards Action".

13.1. Expert Review: Evaluation Guidelines

For the registries for TLV type extensions where an Expert Review is required, the designated expert SHOULD take the same general recommendations into consideration as are specified by [RFC5444].

For the Timestamp TLV, the same type extensions for all Packet, Message and Address TLVs SHOULD be numbered identically.

13.2. Packet TLV Type Registrations

IANA is requested to make allocations from the "Packet TLV Types" namespace of [RFC5444] for the Packet TLVs specified in Table 1.

Name	Type	Type Extension	Description
SIGNATURE	TBD1	0	Signature of a packet
		1	Signature, decomposed into cryptographic function over a hash value, as specified in Section 12 in this document.
		2-223	Expert Review
		224-255	Experimental Use
TIMESTAMP	TBD2	0	Unsigned timestamp of arbitrary length, given by the TLV length field. The MANET routing protocol has to define how to interpret this timestamp
		1-223	Expert Review
		224-255	Experimental Use

Table 1: Packet TLV types

13.3. Message TLV Type Registrations

IANA is requested to make allocations from the "Message TLV Types" namespace of [RFC5444] for the Message TLVs specified in Table 2.

Name	Type	Type Extension	Description
SIGNATURE	TBD3	0	Signature of a message
		1	Signature, decomposed into cryptographic function over a hash value, as specified in Section 12 in this document.
		2-223	Expert Review
		224-255	Experimental Use
TIMESTAMP	TBD4	0	Unsigned timestamp of arbitrary length, given by the TLV length field.
		1-223	Expert Review
		224-255	Experimental Use

Table 2: Message TLV types

13.4. Address Block TLV Type Registrations

IANA is requested to make allocations from the "Address Block TLV Types" namespace of [RFC5444] for the Packet TLVs specified in Table 3.

Name	Type	Type Extension	Description
SIGNATURE	TBD5	0	Signature of an object (e.g. an address)
		1	Signature, decomposed into cryptographic function over a hash value, as specified in Section 12 in this document.
		2-223	Expert Review
		224-255	Experimental Use
TIMESTAMP	TBD6	0	Unsigned timestamp of arbitrary length, given by the TLV length field.
		1-223	Expert Review
		224-255	Experimental Use

Table 3: Address Block TLV types

13.5. Hash Function

IANA is requested to create a new registry for hash functions that can be used when creating a signature, as specified in Section 12 of this document. The initial assignments and allocation policies are specified in Table 4.

Hash function value	Algorithm	Description
0	none	The "identity function": the hash value of an object is the object itself
1-223		Expert Review
224-255		Experimental Use

Table 4: Hash-Function registry

13.6. Cryptographic Algorithm

IANA is requested to create a new registry for the cryptographic function, as specified in Section 12 of this document. Initial assignments and allocation policies are specified in Table 5.

Cryptographic function value	Algorithm	Description
0	none	The "identity function": the value of an encrypted hash is the hash itself
1-223		Expert Review
224-255		Experimental Use

Table 5: Cryptographic function registry

14. Security Considerations

This document does not specify a protocol. It provides a syntactical component for cryptographic signatures of messages and packets as defined in [RFC5444]. It can be used to address security issues of a MANET routing protocol or MANET routing protocol extension. As such, it has the same security considerations as [RFC5444].

In addition, a MANET routing protocol or MANET routing protocol extension that uses this specification MUST specify the usage as well as the security that is attained by the cryptographic signatures of a message or a packet.

As an example, a MANET routing protocol that uses this component to reject "badly formed" messages if a control message does not contain a valid signature, SHOULD indicate the security assumption that if the signature is valid, the message is considered valid. It also SHOULD indicate the security issues that are counteracted by this measure (e.g. link or identity spoofing) as well as the issues that are not counteracted (e.g. compromised keys).

15. Acknowledgements

The authors would like to thank Bo Berry (Cisco), Alan Cullen (BAE), Justin Dean (NRL), Christopher Dearlove (BAE), Paul Lambert (Marvell), Jerome Milan (Ecole Polytechnique) and Henning Rogge (FGAN) for their constructive comments on the document.

16. References

16.1. Normative References

[BCP26] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, BCP 26, May 2008.

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

[RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444, February 2009.

16.2. Informative References

[OLSRv2] Clausen, T., Dearlove, C., and P. Jacquet, "The Optimized Link State Routing Protocol version 2", work in progress draft-ietf-manet-olsrv2-12.txt, July 2011.

[RFC6130] Clausen, T., Dean, J., and C. Dearlove, "MANET Neighborhood Discovery Protocol (NHDP)", RFC 6130, March 2011.

Authors' Addresses

Ulrich Herberg
Fujitsu Laboratories of America
1240 E. Arques Ave. M/S 345
Sunnyvale, CA, 94085
USA

Email: ulrich@herberg.name
URI: http://www.herberg.name/

Thomas Heide Clausen
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33 6 6058 9349
Email: T.Clausen@computer.org
URI: http://www.thomasclausen.org/

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 21, 2011

R. Cole
US Army CERDEC
J. Macker
Naval Research Laboratory
A. Morton
AT&T Laboratories

February 17, 2011

Definition of Managed Objects for Performance Reporting
draft-ietf-manet-report-mib-01

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring autonomous report generation on any device that supports MIBs containing counter and gauge objects for performance monitoring. This allows a management station to instruct a device to build off-line reports to be collected asynchronously by the management station. Further, this REPORT-MIB can be configured in a proxy configuration where the report generation is performed on a device in close network proximity to the device containing the referenced counter objects. Hence, this capability allows network operators to reduce the SNMP polling traffic burden on Mobile Ad-Hoc and Disruption Tolerant Networks which is typical of SNMP performance management applications. This capability also improves the accuracy of the performance reports by minimizing the delay variation between the reporting agent (this MIB) and the data monitor (the MIB containing the monitored counter objects).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring autonomous, off-line report generation for performance monitoring on any device supporting MIBs containing variables that resolve to type Integer32 (i.e., Integer32, Counter, Gauge, or TimeTicks). This REPORT-MIB allows for the report generation to occur on the same device as containing the referenced counter object or on a device in close network proximity to the device with the referenced counter object. This should be useful to devices or networks where efficient use of bandwidth is of concern or where intermittent connectivity is common. Hence, the REPORT-MIB is useful for devices managed over some Mobile Ad-Hoc Networks (MANETs) or Disruption Tolerant Networks (DTNs).

The REPORT-MIB offers three types of off-line reporting. One type offering reports which present statistical analysis of the objects being tracked; found within the reportStatsGroup. The second type offering a means to collect sampled data related to defined MIB objects. This second type of reporting is contained in the reportSampledGroup. The third offering reports which present (collect) raw data values and their time of change from the objects being tracked; found within the reportHistoryGroup.

For statistical reporting, the REPORT-MIB borrows from the RMON [RFC1757] ReportsControl and Reports Tables. Here the reportStatsCapabilitiesGroup defines the capabilities of the device with respect performance monitoring and statistical analysis. Some analysis is hard-coded into the definition of the reportStatsDataGroup while the device can also advertise extended statistical reporting via the reportMetricExtDefTable. The reportsControlTable specifies the report metrics, the Object ID to monitor and other aspects of the statistical report development and storage.

For the collection of sampled data, the REPORT-MIB draws directly from the usrHistoryGroup from RMON 2 [RFC2021]. Here the reportSampledControlTable allows the user to define aspects of the report for sampled data, including the number of MIB objects to be sampled and the nature of the sampling frequency and overall report duration. This group uses the notion of buckets, which contained sampled data from a set of identified MIB objects sampled at the same time point. The report consists of the buckets, each containing sets of sampled data from the selected MIB objects but at the specific sampling times. The reportSampledObjectTable allows the user to identify the multiple MIB objects to be sampled. The reportSampledDataTable contains the storage of the reported sampled data contained within buckets, one bucket for each time sampling instance.

For the collection of raw data, the REPORT-MIB contains a reportHistoryGroup comprised of the reportHistoryControlTable for control of historical data reports and the reportHistoryDataTable for the storage of the historical reports.

Various compliance groups are defined which allow for development of raw data collection reports, collection of sampled data reports or only statistical data reports, or all combinations.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Conventions

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].

4. Overview

The REPORT-MIB references performance objects in other MIBs (and in other devices) and generates offline performance reports on those referenced objects. The REPORT-MIB can be coincident with the other MIB or can reside on another device in close network proximity to the device containing the referenced performance related object.

4.1. REPORT-MIB Management Model

This section describes the management model for the REPORT-MIB process. First, the model for the reportStatsGroup is presented. Then the models for the reportSampledGroup and the reportHistoryGroup are presented.

Figure 1 illustrates a potential use of the REPORT-MIB for the generation of off-line, remotely generated reports. The management station on the left hand side of the illustration instructs the remote device to create reports through manipulation of the ReportCntrl Objects in the REPORT-MIB resident on the remote device. The reports instruct the device to monitor the status of specified counters (on other MIBs and potentially on other devices in close network proximity) periodically and to generate a set of metrics describing the temporal behavior of those counter values. The reports are stored locally until the management station decides to pull them off the device. The figure shows a case where the REPORT-MIB generates a notification that Report_2 has completed, prompting the management station to pull Report_2 from the device.

```
   Mgmt       Device
   Station
                          REPORT-MIB         MIB_1
                          +---------+        +-----+
                          |         |        |     |
    Build_Rep_1           |+-------+|        +--+  |
       +----------------->||cntrl_1||<------>|PC|  |
                          |+-------+|<-+     +--+--+
                          |         | |
                          |         | |     MIB_2
                          |         | |     +-----+
                          |           |     |     |
                          |         | |     +--+  |
                          |         | +---->|PC|  |
    Build_Rep_2           |+-------+|    +-->+--+--+
       +----------------->||cntrl_2||<---+
                          |+-------+|
                          |         |
                          |         |
    Build_Rep_3           |+-------+|
       +----------------->||cntrl_3||<---+   MIB_n
                          |+-------+|    |   +-----+
                          |         |    |   |     |
                          |+-----+  |    |   +--+  |
                          ||Rep_1|  |    +-->|PC|  |
                          |+-----+  |        +--+--+
                          |         |
                          |+-----+  |
       <------------------||Rep_2|  |
    Notf_Rep_2            |+-----+  |
                          |         |
                          |+-----+  |
                          ||Rep_3|  |
                          |+-----+  |
    Get_Rep_2             |         |
       +----------------->|         |
                          |         |
       <-----------------+|         |
    Send_Rep_2            +---------+
```

Figure 1: REPORT-MIB front-end report generation process.

The REPORT-MIB's reportStatsGroup defines specifically a set of metrics which are computed within all reports. It also allows for the specification of metric extensions which are local to the specific implementation of the REPORT-MIB. These are identified in the reportStatsCapabilitiesGroup metricExtension Table.

Each metric has an associated Object ID of type counter associated with it. The control table specifies a report interval and a bin interval. The report interval is an integral multiple of the bin interval. For each bin interval, the device identifies the change in the counter value over the bin interval (called x_i) and then computes the associated metric, e.g., sum, sum of the square, etc, over the set {x_i}. It maintains the sum of these computations within the metric objects in the 'reportStatsDataTable'. Once the report interval is complete, the management station has enough information to compute a set of interesting and useful statistics.

The computational model of the reportStatsGroup of the REPORT-MIB is illustrated in the figure below. The important controls are a) the contrlInterval, b) the cntrlBinInterval, c) the specific counterObjectId, and d) the metric. In the figure x_i represents the ith value of the counter change, i.e., x_i = counterValue(t_i+1) - counterValue(t_i). The metrics reported are then computed from the set (x_i). Three examples are identified in the figure, e.g., StatSumX, StatSumSq and StatMaxX. Other existing and potential metrics are discussed below.

```
  |
  |<-------------------- cntrlInterval ---------------------------->|
  |                                                                  |

  |     |     |     |     |                              |     |     |
  | x_0 | x_1 | x_2 | x_3 |            ...               |x_n-2|x_n-1|
  +-----+-----+-----+-----+---                        ---+-----+-----+

  ^     ^     ^     ^     ^                              ^     ^     ^
  t_0   t_1   t_2   t_3   t_4                            t_n-2 t_n-1 t_n


  where t_i - t_i-1 = cntrlBinInterval
        n = cntrlInterval / cntrlBinInterval

        StatSumX = Sum(x_i)  from i=0, ..., n-1
        StatSumSq = Sum((x_i)^2) from i=0, ..., n-1
        StatMaxX = Max(x_i) for i=0, ..., n-1
```

Figure 2: REPORT-MIB statistical analysis computation process.

This capability then allows for the computation of various significant statistics related to the behavior of the referenced object.

- Maximum and Minimum - the maximum and the minimum change in the referenced object during a single cntrlBinInterval during the cntrlInterval.

- Arithmetic Mean - the mean change in the referenced object over all control bin intervals during the cntrlInterval. This is derived from the StatSumX quantity.

- Variance - the variance in the change of the referenced object over all control bin intervals within the cntrlInterval. This is derived from the StatSumSq and the StatSumX quantities.

These are accessible from the statistical datum provided by this MIB module. Other statistics are derivable including, e.g., the slope of a least-squares fit to the rate of change of the referenced object. These are described below.

The REPORT-MIB also provides for the collection of sampled data instead of statistical data. It does this by importing (copying) the usrHistory group from RMON2 [RFC2021] which allows for the generation of reports collecting the sampled object values binned for the purpose of aggregation and efficiency of collection. These are defined within the reportSampledGroup. The model used for this type of report generation is based upon three tables. The reportSampledControlTable defines aspects of the report generation related to duration of the reporting interval, the bin (or bucket) sizes for the report, and the number of object values collected for each bucket. The reportUsrHistoryObjectTable identifies the specific MIB objects whose values are binned within the report. And the reportSampledDataTable contains the binned data values collected for the report.

The REPORT-MIB also provides for the collection of historical data instead of statistical or sampled data. It does this by defining the reportHistoryControlTable for the control of the historical reports and the reportHistoryDataTable for the storage of the historical reports.

4.2. Terms

The following definitions apply throughout this document:

- Capabilities - Objects related to the capabilities of the device and MIB implemented on the device. Some objects are explicitly defined within the REPORT-MIB. Other capabilities can be exposed through the REPORT-MIB, but which are not explicitly defined within this document. These later capabilities include objects, e.g., for new metrics.

- Control - Objects defined within this document which set the parameters for specific reports to be generated offline on the the remote managed device.

- Data - Objects which hold the report data, either statistical, sampled or raw history data.

5. Structure of the MIB Module

This section presents the structure of the REPORT-MIB module. The objects are arranged into the following groups:

- reportMIBNotifications - defines the notifications associated with the REPORT-MIB.

- reportMIBObjects - defines the objects forming the basis for the REPORT MIB. These objects are divided up by function into the following groups:

-

  * Statistics Group - This group contains the objects which support the generation of reports of a statistical nature.

  * Sampled Group - This group contains the objects which support the generation (collection) of reports exposing sampled data values.

  * History Group - This group contains the objects which support the generation (collection) of historical reports exposing raw data values.

- reportMIBConformance - Defines a variety of conformance of implementations of this REPORT-MIB.

5.1. Textual Conventions

The textual conventions used in the REPORT-MIB are as follows. The RowStatus textual convention is imported from RFC 2579 [RFC2579].

5.2. The Statistics Group

The REPORT-MIB Statistics Group contains objects which allows for the generation of statistical analysis reports. For example, this group can be exercised to generate the mean and variance of the referenced counter object. The Statistics Group is composed of:

- reportStatsCapabilitiesGroup - lists the statistics collections capabilities of this device. Certain statistics are mandatory, i.e., hard coded into the MIB definitions. While, the capabilities group allows the developer to add additional statistical analysis capabilities.

- reportStatsControlGroup - allows the management application to define the parameters of the reports.

- reportStatsDataGroup - presents the data from the specified reports.

As an example of how the metrics are to be computed within the REPORT-MIB, consider the standard metric object 'reportStatsDataStatSumX'. For each bin interval defined by the object reportCntrlReportsBinInterval, the change in the value of the counter pointed to by the Object ID reportCntrlReportsPriObjID is calculated. Then this (delta) value is added to the current value of the value contained in the object 'reportStatsDataStatSumX'. Then, if interested in computing the average change in this object (sampled each bin interval) for the duration of the report, the management station simply divides reportStatsDataStatSumX by reportStatsDataStatN. Although this is a trivial example because the value of reportAggrReportStatSumX is simple the difference in the counter reportCntrolReportsPriObjID at the start and the end of the total report interval, the other metrics defined are not as trivial.

The objects 'reportStatsDataOverflowStatSumX' and 'reportStatsDataHCSumX' are borrowed from RMON [RFC2021] and exist to handle integer overflow situations where, e.g., 'reportStatsDataStatSumX' overruns its maximum value numerous times.

Computation of the least-square fit of the data collected for a report can be accomplished. (NOTE: describe this capability here.)

5.3. The Sampled Group

The Sampled Group contains tables which allows for the development of reports based upon sampling the referenced counter objects at specified intervals. The development of this group within the REPORT-MIB follows exactly the User History group from the RMON 2 MIB [RFC2021]. The Sampled Group is composed of:

- reportSampledControlTable - allows for the setting of the parameters of the report.

- reportSampledObjectTable - sets the referenced objects to be sampled during the test. With this capability, the management application can reference multiple objects, all of which are sampled during the test and reported out through the reportSampledData Table.

- o reportSampledDataTable - contains the reports.

5.4. The History Group

The History Group contains tables which capture information on change events for the referenced objects. Depending upon the referenced objects, this could force the generation of large amounts of data. Care should be exercised when considering the use of this capability.

- o reportHistoryControlTable - defines the parameters for the test.
- o reportHistoryDataTable - presents the reports associated with the constructed tests.

5.5. The Notifications Group

The Notifications Sub-tree contains the list of notifications supported within the REPORT-MIB and their intended purpose or utility. (Note: This group is currently empty.)

6. Relationship to Other MIB Modules

[TODO]: The text of this section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. Definitions imported from other MIB modules and other MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

6.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The REPORT-MIB does not duplicate those objects.

6.2. Relationship to the RMON2-MIB

The REPORT-MIB is closely related in many aspects to the RMON2-MIB [RFC2021]. Specifically, the reportSampledGroup is a direct copy of the RMON2 User History Group, with the names changed to comply with the naming conventions within the REPORT-MIB. Further, the design and use of the control tables within the REPORT-MIB draw exactly from the definition of these table structures in the earlier RMON MIBs.

6.3. Relationship to the TPM-MIB

The REPORT-MIB pulled the reportStatsGroup directory from the TPM-MIB [RFC4150]. The table structures and the choice of statistics draws directly from the earlier TPM-MIB developed within the RMON Working Group.

6.4. MIB modules required for IMPORTS

[TODO]: Citations are not permitted within a MIB module, but any module mentioned in an IMPORTS clause or document mentioned in a REFERENCE clause is a Normative reference, and must be cited someplace within the narrative sections. If there are imported items in the MIB module, such as Textual Conventions, that are not already cited, they can be cited in text here. Since relationships to other MIB modules should be described in the narrative text, this section is typically used to cite modules from which Textual Conventions are imported.

The REPORT-MIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF [RFC2580], and IF-MIB [RFC2863]

7. Definitions

```
REPORT-MIB DEFINITIONS ::= BEGIN

IMPORTS

   ZeroBasedCounter32
      FROM RMON2-MIB                           -- [RFC2021]

   MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
   Counter32, Gauge32, Unsigned32, Integer32, mib-2
      FROM SNMPv2-SMI                          -- [RFC2578]

   TEXTUAL-CONVENTION, RowStatus,
   TimeStamp, StorageType
      FROM SNMPv2-TC                           -- [RFC2579]

   MODULE-COMPLIANCE, OBJECT-GROUP,
   NOTIFICATION-GROUP
      FROM SNMPv2-CONF                         -- [RFC2580]

   OwnerString
```

```
      FROM RMON-MIB                                -- [RFC2819]

   ZeroBasedCounter64
      FROM HCNUM-TC                                -- [RFC2856]

   SnmpAdminString
      FROM SNMP-FRAMEWORK-MIB                      -- [RFC3411]

   InetAddress, InetAddressType
      FROM INET-ADDRESS-MIB                        -- [RFC4001]

   SspmClockSource, SspmClockMaxSkew,
   SspmMicroSeconds
      FROM SSPM-MIB                               -- [RFC4149]
   ;

reportMIB MODULE-IDENTITY
   LAST-UPDATED "201102171300Z"  -- February 17, 2011
   ORGANIZATION "IETF MANET Working Group"
   CONTACT-INFO
      "WG E-Mail: manet@ietf.org

       WG Chairs: ian.chakeres@gmail.com
                  jmacker@nrl.navy.mil


       Editors:   Robert G. Cole
                  US Army CERDEC
                  328 Hopkins Road
                  Aberdeen Proving Ground, MD 21005
                  USA
                  +1 410 278-6779
                  robert.g.cole@us.army.mil

                  Joseph Macker
                  Naval Research Laboratory
                  Washington, D.C. 20375
                  USA
                  macker@itd.nrl.navy.mil

                  Al Morton
                  AT&T Laboratories
                  Middletown, N.J. 07724
                  USA
                  amorton@att.com"
   DESCRIPTION
      "This MIB module contains managed object definitions for
       the autonmous reporting of performance object counters.
```

```
          Copyright (C) The IETF Trust (2009). This version
          of this MIB module is part of RFC xxxx; see the RFC
          itself for full legal notices."

        -- Revision History
        REVISION    "201102171300Z"   -- February 17, 2011
        DESCRIPTION
           "The fifth draft of this MIB module published as
            draft-ietf-manet-report-mib-01.txt.  This document
            has been promoted to a MANET Working Group
            draft.

            Revisions to this draft include
            a) Proposed changes to the statsReport table to
               simplify communications between device and
               mgmt application,
            b) Added Notifications,
            c) Changed the reporting structure of the
               Sampled and the History reporting
               to align with the structure of the
               Statistics reports for the purpose of
               allowing for efficient notification and
               collection of data reports.
            d) Ran through smilint to clean up all errors
               and most warning.  A few still remain.
            "
        REVISION    "201007051300Z"   -- July 05, 2010
        DESCRIPTION
           "The fourth draft of this MIB module published as
            draft-ietf-manet-report-mib-00.txt.  This document
            has been promoted to a MANET Working Group
            draft.

            Significant revisions to this draft include
            a) added support for proxy configurations through
            the addition of address objects associated with
            the referenced counter objects associated with the
            performance reports."
        REVISION    "201003021300Z"   -- March 02, 2010
        DESCRIPTION
           "The third draft of this MIB module published as
            draft-cole-manet-report-mib-02.txt.  Significant
            revisions to this draft include a) changed naming
            of usrHistoryGroup to sampledGroup and  b) added
            a historyGroup."
        REVISION    "200910251300Z"   -- October 25, 2009
        DESCRIPTION
           "The second draft of this MIB module published as
```

```
         draft-cole-manet-report-mib-01.txt.  Significant
         revisions to this draft include a) the inclusion of
         raw data collection borrow blatently from the
         usrHistory Group within RMON2, b) the deletion of
         the CurrentHistoryTable from version -00,
         c) modifications to the overall structure of the
         MIB, and d) the definition of various Compliance
         options for implementations related to this MIB."
     REVISION    "200904281300Z"   -- April 28, 2009
     DESCRIPTION
        "Initial draft of this MIB module published as
         draft-cole-manet-report-mib-00.txt."
     -- RFC-Editor assigns XXXX
     ::= { mib-2 998 }   -- to be assigned by IANA



-- TEXTUAL CONVENTIONs

   ReportMetricDefID ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "d"
       STATUS       current
       DESCRIPTION
           "An index that identifies through reference to a specific
           statistical metrics.
           "
       SYNTAX       Unsigned32 (1..2147483647)



--
-- Top-Level Object Identifier Assignments
--

reportMIBNotifications OBJECT IDENTIFIER ::= { reportMIB 0 }
reportMIBObjects       OBJECT IDENTIFIER ::= { reportMIB 1 }
reportMIBConformance   OBJECT IDENTIFIER ::= { reportMIB 2 }

-- The reportMIBObjects Assignments:
--      reportStatsGroup          - 1
--      reportSampledGroup        - 2
--      reportHistoryGroup        - 3




reportStatsGroup       OBJECT IDENTIFIER ::= { reportMIBObjects 1 }
```

```
-- Then, the reportStatsGroup assignments are :
--      reportStatsCapabilitiesGroup    - 1
--      reportStatsControlGroup         - 2
--      reportStatsDataGroup            - 3


-- reportStatsCapabilitiesGroup
--    This group contains the REPORT objects that identify specific
--    capabilities within this device related to REPORT functions.


reportCapabilitiesGroup  OBJECT IDENTIFIER ::= { reportStatsGroup 1 }

reportClockResolution  OBJECT-TYPE
    SYNTAX      SspmMicroSeconds
    MAX-ACCESS  read-only
    STATUS      current
    -- UNITS       Microseconds
    DESCRIPTION
        "A read-only variable indicating the resolution
         of the measurements possible by this device."
    ::= { reportCapabilitiesGroup 1 }

reportClockMaxSkew  OBJECT-TYPE
    SYNTAX      SspmClockMaxSkew
    MAX-ACCESS  read-only
    STATUS      current
    -- UNITS       Seconds
    DESCRIPTION
        "A read-only variable indicating the maximum
         offset error due to skew of the local clock
         over the time interval 86400 seconds, in seconds."
    ::= { reportCapabilitiesGroup 2 }

reportClockSource  OBJECT-TYPE
    SYNTAX      SspmClockSource
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A read-only variable indicating the source of the clock.
         This is provided to allow a user to determine how accurate
         the timing mechanism is compared with other devices."
    ::= { reportCapabilitiesGroup 3 }

reportMetricDirLastChange  OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value of sysUpTime at the time the
        reportTransMetricDirTable was last modified, through
        modifications of the reportTransMetricDirConfig object."
    ::= { reportCapabilitiesGroup 4 }


-- REPORT Metric Extensions Definition Table

reportMetricExtDefTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF ReportMetricExtDefEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The reportMetricExtDefTable describes the metrics
        available to the REPORT-MIB.  The reportMetricExtDefTable
        can define metrics by referencing existing IETF,
        ITU, and other standards organizations’ documents,
        including enterprise-specific documents.
        Examples of appropriate references include the
        ITU-T Recommendation Y.1540 [Y.1540] on IP
        packet transfer performance metrics and the
        IETF documents from the IPPM WG; e.g., RFC2681
        on the round trip delay metric [RFC2681] or
        RFC3393 on the delay variation metric [RFC3393].
        Other examples include RFC2679 [RFC2679], RFC2680
        [RFC2680], and RFC3432 [RFC3432].  Although no
        specific metric is mandatory, implementations
        should, at a minimum, support a round-trip delay
        and a round-trip loss metric.

        This table contains one row per metric supported by this
        agent, and it should be populated during system
        initialization."
    ::= { reportCapabilitiesGroup 5 }

reportMetricExtDefEntry  OBJECT-TYPE
    SYNTAX      ReportMetricExtDefEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a particular metric."
    INDEX   { reportMetricExtDefID }
    ::= { reportMetricExtDefTable 1 }

ReportMetricExtDefEntry ::= SEQUENCE {
      reportMetricExtDefID            ReportMetricDefID,
      reportMetricExtDefType          INTEGER,
      reportMetricExtDefName           SnmpAdminString,
      reportMetricExtDefOperation      SnmpAdminString,
      reportMetricExtDefReference      SnmpAdminString
    }

reportMetricExtDefID OBJECT-TYPE
    SYNTAX      ReportMetricDefID
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The index for this entry.  This object identifies
         the particular metric in this MIB module."
    ::= { reportMetricExtDefEntry 1 }

reportMetricExtDefType  OBJECT-TYPE
    SYNTAX      INTEGER  {
                          other(1),
                          singleObjMetric(2),
                          multipleObjMetric(3)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The basic type of metric indicated by this entry.

        The value 'other(1)' indicates that this metric cannot be
        characterized by any of the remaining enumerations specified
        for this object.

        The value 'connectMetric(2)' indicates that this metric
        measures connectivity characteristics.

        The value 'delayMetric(3)' indicates that this metric
        measures delay characteristics.
        "
    ::= { reportMetricExtDefEntry 2 }

reportMetricExtDefName  OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
       "The textual name of this metric.  For example, if
        this reportMetricDefEntry identified the IPPM metric for
        round trip delay, then this object should contain
        the value, e.g., 'Type-P-Round-Trip-Delay'."
    ::= { reportMetricExtDefEntry 3 }
```

```
reportMetricExtDefOperation  OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The textual description of the operations necessary
        to compute this metric.  For example, if
        this reportMetricDefEntry identified the IPPM metric for
        round trip delay, then this object should contain
        the value, e.g., 'Type-P-Round-Trip-Delay'."
    ::= { reportMetricExtDefEntry 4 }

reportMetricExtDefReference  OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object contains a reference to the document that
        defines this metric.  If this document is available online
        via electronic download, then a de-referencable URL
        should be specified in this object.  The implementation
        must support an HTTP URL type and may support additional
        types of de-referencable URLs such as an FTP type.

        For example, if this reportMetricDefName identified the IPPM
        metric 'Type-P-Round-Trip-Delay', then this object should
        contain the value, e.g.,
        'http://www.ietf.org/rfc/rfc2681.txt'."
    ::= { reportMetricExtDefEntry 5 }



-- Stats Control Group
--     This and the following tables are modeled
--     after the report control and collection
--     capabilities found in RMON 2, RFC 2021

reportStatsControlGroup OBJECT IDENTIFIER ::= {reportStatsGroup 2}

reportStatsControlTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF ReportStatsControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The reportStatsControlTable is the controlling entry
        that manages the population of studies in the
        Report for selected time intervals.
        Note that this is not like the typical RMON
        controlTable and dataTable in which each entry creates
        its own data table.  Each entry in this table enables the
        creation of multiple data tables on a study basis.  For each
        interval, the study is updated in place, and the current
        data content of the table becomes invalid.

        The control table entries are persistent across
        system reboots."
    ::= { reportStatsControlGroup 1 }

reportStatsControlEntry  OBJECT-TYPE
    SYNTAX      ReportStatsControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A conceptual row in the reportStatsControlTable.

        An example of the indexing of this entry is
        reportGenReportCntrInterval.1"
    INDEX { reportStatsControlIndex }
    ::= { reportStatsControlTable 1 }

ReportStatsControlEntry ::= SEQUENCE {
    reportStatsControlIndex             Unsigned32,
    reportStatsControlInterval          Unsigned32,
    reportStatsControlBinInterval       Unsigned32,
    reportStatsControlPriObjID          OBJECT IDENTIFIER,
    reportStatsControlPriObjIpAddrType  InetAddressType,
    reportStatsControlPriObjIPAddr      InetAddress,
    reportStatsControlSecObj1ID         OBJECT IDENTIFIER,
    reportStatsControlSecObj1IpAddrType InetAddressType,
    reportStatsControlSecObj1IPAddr     InetAddress,
    reportStatsControlSecObj2ID         OBJECT IDENTIFIER,
    reportStatsControlSecObj2IpAddrType InetAddressType,
    reportStatsControlSecObj2IPAddr     InetAddress,
    reportStatsControlSecObj3ID         OBJECT IDENTIFIER,
    reportStatsControlSecObj3IpAddrType InetAddressType,
    reportStatsControlSecObj3IPAddr     InetAddress,
    reportStatsControlSecObj4ID         OBJECT IDENTIFIER,
    reportStatsControlSecObj4IpAddrType InetAddressType,
    reportStatsControlSecObj4IPAddr     InetAddress,
    reportStatsControlSecObj5ID         OBJECT IDENTIFIER,
    reportStatsControlSecObj5IpAddrType InetAddressType,
    reportStatsControlSecObj5IPAddr     InetAddress,
    reportStatsControlMetricExt1        ReportMetricDefID,
    reportStatsControlMetricExt2        ReportMetricDefID,
    reportStatsControlMetricExt3        ReportMetricDefID,
    reportStatsControlMetricExt4        ReportMetricDefID,
    reportStatsControlMetricExt5        ReportMetricDefID,
    reportStatsControlReqReports        Unsigned32,
    reportStatsControlGrantedReports    Unsigned32,
    reportStatsControlStartTime         TimeStamp,
    reportStatsControlReportNumber      Unsigned32,
    reportStatsControlInsertsDenied     Counter32,
    reportStatsControlOwner             OwnerString,
    reportStatsControlStorageType       StorageType,
    reportStatsControlStatus            RowStatus
}

reportStatsControlIndex  OBJECT-TYPE
    SYNTAX      Unsigned32 (1..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "An index that uniquely identifies an entry in the
        reportStatsControlTable.  Each such entry defines a unique
        report whose results are placed in the reportGenReportTable
        on behalf of this reportStatsControlEntry."
    ::= { reportStatsControlEntry 1 }


reportStatsControlInterval  OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "Seconds"
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
       "The interval in seconds over which data is accumulated before
        being aggregated into a report in the reportGenReportTable.
        All reports with the same reportStatsControlIndex will be
        based on the same interval.

        The value of the reportStatsControlInterval should be
        an integral multiple of the value of the
        reportStatsControlBinInterval.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    DEFVAL { 3600 }
    ::= { reportStatsControlEntry 2 }

reportStatsControlBinInterval  OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "Seconds"
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
       "The interval in seconds between which the value of the
        reportStatsControlPriObjID and SecObjIDs are polled
        for the purpose of generating the metric values associated
        with this report.  All reports with the same
        reportStatsControlIndex will be based on the
        same bin interval.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    DEFVAL { 3600 }
    ::= { reportStatsControlEntry 3 }

reportStatsControlPriObjID  OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the primary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 4 }

reportStatsControlPriObjIpAddrType  OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        primary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 5 }

reportStatsControlPriObjIPAddr  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        primary counter object to be
        monitored within this report.
        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 6 }

reportStatsControlSecObj1ID  OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the secondary counter object to be
        monitored within this report associated with the
        specified reportStatsControlMetricExt1.  If the
        reportStatsControlMetricExt1 is a simple metric, then
        the value of this reportStatsControlSecObj1ID is
        set to '0'.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 7 }

reportStatsControlSecObj1IpAddrType  OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 8 }

reportStatsControlSecObj1IPAddr  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 9 }

reportStatsControlSecObj2ID  OBJECT-TYPE
```

```
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the secondary counter object to be
        monitored within this report associated with the
        specified reportStatsControlMetricExt2.  If the
        reportStatsControlMetricExt2 is a simple metric, then
        the value of this reportStatsControlSecObj2ID is
        set to '0'.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 10 }

reportStatsControlSecObj2IpAddrType  OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 11 }

reportStatsControlSecObj2IPAddr  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 12 }

reportStatsControlSecObj3ID  OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the secondary counter object to be
```

```
            monitored within this report associated with the
            specified reportStatsControlMetricExt3.  If the
            reportStatsControlMetricExt3 is a simple metric, then
            the value of this reportStatsControlSecObj3ID is
            set to '0'.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 13 }

    reportStatsControlSecObj3IpAddrType  OBJECT-TYPE
        SYNTAX      InetAddressType
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the IP address type
            of the IP address associated with the
            secondary counter object to be
            monitored within this report.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 14 }

    reportStatsControlSecObj3IPAddr  OBJECT-TYPE
        SYNTAX      InetAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the IP addree of the
            secondary counter object to be
            monitored within this report.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 15 }

    reportStatsControlSecObj4ID  OBJECT-TYPE
        SYNTAX      OBJECT IDENTIFIER
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the secondary counter object to be
            monitored within this report associated with the
            specified reportStatsControlMetricExt4.  If the
            reportStatsControlMetricExt4 is a simple metric, then
            the value of this reportStatsControlSecObj4ID is
            set to '0'.
            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 16 }

    reportStatsControlSecObj4IpAddrType  OBJECT-TYPE
        SYNTAX      InetAddressType
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the IP address type
            of the IP address associated with the
            secondary counter object to be
            monitored within this report.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 17 }

    reportStatsControlSecObj4IPAddr  OBJECT-TYPE
        SYNTAX      InetAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the IP addree of the
            secondary counter object to be
            monitored within this report.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 18 }

    reportStatsControlSecObj5ID  OBJECT-TYPE
        SYNTAX      OBJECT IDENTIFIER
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the secondary counter object to be
            monitored within this report associated with the
            specified reportStatsControlMetricExt5.  If the
            reportStatsControlMetricExt5 is a simple metric, then
            the value of this reportStatsControlSecObj5ID is
            set to '0'.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 19 }

    reportStatsControlSecObj5IpAddrType  OBJECT-TYPE
```

```
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 20 }

reportStatsControlSecObj5IPAddr  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 21 }

reportStatsControlMetricExt1  OBJECT-TYPE
    SYNTAX      ReportMetricDefID
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the first metric extension placed
        in the reportGenReportTable.  If no metric extension
        is requested, then this object value is set to '0'.

        If this metric is defined on a single counter object,
        then only the reportStatsControlPriObjID is set, while
        the value of the reportStatsControlSecObjID is
        set to '0'.  Else, the reportStatsControlSecObjID
        is set in accoradance with the instruction in the
        definition of the metric extension found in the
        reportCapabilitiesMetwircExtTable above.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 22 }

reportStatsControlMetricExt2  OBJECT-TYPE
```

```
    SYNTAX      ReportMetricDefID
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the second metric extension placed
        in the reportGenReportTable.  If no metric extension
        is requested, then this object value is set to '0'.

        If this metric is defined on a single counter object,
        then only the reportStatsControlPriObjID is set, while
        the value of the reportStatsControlSecObjID is
        set to '0'.  Else, the reportStatsControlSecObjID
        is set in accoradance with the instruction in the
        definition of the metric extension found in the
        reportCapabilitiesMetwircExtTable above.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 23 }

reportStatsControlMetricExt3  OBJECT-TYPE
    SYNTAX      ReportMetricDefID
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the third metric extension placed
        in the reportGenReportTable.  If no metric extension
        is requested, then this object value is set to '0'.

        If this metric is defined on a single counter object,
        then only the reportStatsControlPriObjID is set, while
        the value of the reportStatsControlSecObjID is
        set to '0'.  Else, the reportStatsControlSecObjID
        is set in accoradance with the instruction in the
        definition of the metric extension found in the
        reportCapabilitiesMetwircExtTable above.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 24 }

reportStatsControlMetricExt4  OBJECT-TYPE
    SYNTAX      ReportMetricDefID
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the fourth metric extension placed
        in the reportGenReportTable.  If no metric extension
```

```
            is requested, then this object value is set to '0'.

            If this metric is defined on a single counter object,
            then only the reportStatsControlPriObjID is set, while
            the value of the reportStatsControlSecObjID is
            set to '0'.  Else, the reportStatsControlSecObjID
            is set in accoradance with the instruction in the
            definition of the metric extension found in the
            reportCapabilitiesMetwircExtTable above.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 25 }

    reportStatsControlMetricExt5  OBJECT-TYPE
        SYNTAX      ReportMetricDefID
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This identifies the fifth metric extension placed
            in the reportGenReportTable.  If no metric extension
            is requested, then this object value is set to '0'.

            If this metric is defined on a single counter object,
            then only the reportStatsControlPriObjID is set, while
            the value of the reportStatsControlSecObjID is
            set to '0'.  Else, the reportStatsControlSecObjID
            is set in accoradance with the instruction in the
            definition of the metric extension found in the
            reportCapabilitiesMetwircExtTable above.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 26 }


    reportStatsControlReqReports  OBJECT-TYPE
        SYNTAX      Unsigned32 (1..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The number of saved reports requested to be allocated on
            behalf of this entry.

            This object may not be modified if the associated
            reportStatsControlStatus object is equal to active(1)."
        ::= { reportStatsControlEntry 27 }
```

```
reportStatsControlGrantedReports  OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of saved reports the agent has allocated based
        on the requested amount in reportStatsControlReqReports.
        Because each report can have many entries, the total number
        of entries allocated will be this number multiplied by the
        value of reportStatsControlGrantedSize, or by 1 if that
        object doesn't exist.

        When the associated reportStatsControlReqReports object is
        created or modified, the agent should set this object as
        closely to the requested value as is possible for the
        particular implementation and available resources.  When
        considering available resources, the agent must consider its
        ability to allocate this many reports, each with the number
        of entries represented by reportStatsControlGrantedSize, or
        by 1 if that object doesn't exist.

        Note that although the storage required for each report may
        fluctuate due to changing conditions, the agent must continue
        to have storage available to satisfy the full report size for
        all reports, when necessary.  Further, the agent must not
        lower this value except as a result of a set to the
        associated reportStatsControlReqSize object."
    ::= { reportStatsControlEntry 28 }

reportStatsControlStartTime  OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value of sysUpTime when the system began processing the
        report in progress.  Note that the report in progress is not
        available.

        This object may be used by the management station to figure
        out the start time for all previous reports saved for this
        reportStatsControlEntry, as reports are started at fixed
        intervals."
    ::= { reportStatsControlEntry 29 }

reportStatsControlReportNumber  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of the report in progress.  When an
        reportStatsControlEntry is activated, the first report will
        be numbered zero."
    ::= { reportStatsControlEntry 30 }

reportStatsControlInsertsDenied  OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of attempts to add an entry to reports for
        this ReportStatsControlEntry that failed because the
        number of entries would have exceeded
        reportStatsControlGrantedSize.

        This number is valuable in determining if enough
        entries have been allocated for reports
        in light of fluctuating network
        usage.  Note that an entry that is denied will
        often be attempted again, so this number will
        not predict the exact number of additional entries
        needed, but it can be used to
        understand the relative magnitude of the problem.

        Also note that there is no ordering specified for
        the entries in the report;
        thus, there are no rules for which entries
        will be omitted when not enough entries are available.
        As a consequence, the agent is not required
        to delete 'least valuable' entries first."
    ::= { reportStatsControlEntry 31 }

reportStatsControlOwner  OBJECT-TYPE
    SYNTAX      OwnerString
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The entity that configured this entry and is
        therefore using the resources assigned to it.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportStatsControlEntry 32 }

reportStatsControlStorageType  OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The storage type of this reportStatsControlEntry.  If the
        value of this object is 'permanent', no objects in this row
        need to be writable."
    ::= { reportStatsControlEntry 33 }

reportStatsControlStatus  OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The status of this performance control entry.

        An entry may not exist in the active state unless each
        object in the entry has an appropriate value.

        Once this object is set to active(1), no objects in the
        reportStatsControlTable can be changed.

        If this object is not equal to active(1), all associated
        entries in the reportGenReportTable shall be deleted."
    ::= { reportStatsControlEntry 34 }




-- Stats Data Group

reportStatsDataGroup  OBJECT IDENTIFIER ::= { reportStatsGroup 3 }


-- Report Stats Data Table

reportStatsDataTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF ReportStatsDataEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains completed
        studies for each of the control table entries in
        reportAggrReportCntrlTable.  These studies are
        provided based on the selections and parameters
        found for the entry in the
        reportAggregateReportsCntrlTable.

        The performance statistics are specified in the
        reportTransMetricDirTable associated with the
        application in question and indexed by
        appLocalIndex and reportTransMetricIndex."
    ::= { reportStatsDataGroup 1 }

reportStatsDataEntry  OBJECT-TYPE
    SYNTAX      ReportStatsDataEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A conceptual row in the reportStatsDataTable.

        The reportStatsControlIndex value in the
        index identifies the reportStatsControlEntry
        on whose behalf this entry was created.

        The reportStatsDataIndex value in the index
        identifies which report
        (in the series of reports) this entry is a part of.

        The reportStatsDataServerAddress value in the
        index identifies the network layer address of the
        device generatung this report.

        An example of the indexing of this entry is
        reportStatsDataStatN.3.15.34.262.18.4.128.2.6.7.3256521"
    INDEX { reportStatsControlIndex,
            reportStatsDataIndex
          }
    ::= { reportStatsDataTable 1 }

-- Note: Thinking about restructuring this
--    table somewhat, in order
--    to allow for a more complete report information to
--    simplify report collection from the remote
--    mgmt application.  Indicating below potential
--    additional objects.
ReportStatsDataEntry ::= SEQUENCE {
    reportStatsDataIndex                Unsigned32,
    -- reportStatsDataServerAddrType    inetAddressType,
    -- reportStatsDataServerAddress     inetAddress,
    reportStatsDataServerAddress        OCTET STRING,
    -- reportStatsDataReportStartTime   TimeStamp,
    -- reportStatsDataReportInterval    Unsigned32,
    reportStatsDataStatN                ZeroBasedCounter32,
    reportStatsDataStatSumX             ZeroBasedCounter32,
    reportStatsDataOverflowStatSumX     ZeroBasedCounter32,
    reportStatsDataHCStatSumX           ZeroBasedCounter64,
    reportStatsDataStatMaximum          ZeroBasedCounter32,
    reportStatsDataStatMinimum          ZeroBasedCounter32,
    reportStatsDataStatSumSq            ZeroBasedCounter32,
    reportStatsDataOverflowStatSumSq    ZeroBasedCounter32,
    reportStatsDataHCStatSumSq          ZeroBasedCounter64,
    reportStatsDataStatSumIX            ZeroBasedCounter32,
    reportStatsDataOverflowStatSumIX    ZeroBasedCounter32,
    reportStatsDataHCStatSumIX          ZeroBasedCounter64,
    reportStatsDataStatSumIXSq          ZeroBasedCounter32,
    reportStatsDataOverflowStatSumIXSq  ZeroBasedCounter32,
    reportStatsDataHCStatSumIXSq        ZeroBasedCounter64,
    reportStatsDataStatMetricExt1       ZeroBasedCounter32,
    reportStatsDataStatMetricExt2       ZeroBasedCounter32,
    reportStatsDataStatMetricExt3       ZeroBasedCounter32,
    reportStatsDataStatMetricExt4       ZeroBasedCounter32,
    reportStatsDataStatMetricExt5       ZeroBasedCounter32
}

reportStatsDataIndex  OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value of reportStatsControlReportNumber for the report to
        which this entry belongs."
    ::= { reportStatsDataEntry 1 }

-- [Note: Need to revisit the syntax for this object of type ’address’.]
reportStatsDataServerAddress  OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..108))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The network layer address of the server host in this
        conversation.

        This is represented as an octet string with specific
        semantics and length as identified by the
        protocolDirLocalIndex component of the index.

        Because this object is an index variable, it is encoded in
        the index according to the index encoding rules.  For
        example, if the protocolDirLocalIndex indicates an
        encapsulation of IPv4, this object is encoded as a length
        octet of 4, followed by the 4 octets of the IPv4 address,
        in network byte order.

        If the associated reportAggrReportCntrlAggrType is equal to
        application(4) or client(2), then this object will be a null
        string and will be encoded simply as a length octet of 0."
    ::= { reportStatsDataEntry 2 }

reportStatsDataStatN  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The count of the total number of data points for the
        specified metric.  This number is simply the value of
        reportCntrlReportsInterval divided by the value of
        reportCntrlReportsBinInterval, which should be integer
        valued.
        "
    ::= { reportStatsDataEntry 3 }

reportStatsDataStatSumX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The sum of all the data point values for the specified
        metric.  This number always represents the total values
        of the statistical datum analyzed.  Each metric
        specifies the exact meaning of this object.
        This value represents the results of one metric and is
        related directly to the specific parameters of the metric
        and the Server and Client addresses involved."
    ::= { reportStatsDataEntry 4 }

reportStatsDataOverflowStatSumX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of times the associated
        reportAggrReportStatSumX counter has overflowed.
        Note that this object will only be instantiated if the
        associated reportAggrReportHCStatSumX object is also
        instantiated for a particular dataSource."
    ::= { reportStatsDataEntry 5 }

reportStatsDataHCStatSumX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The high-capacity version of reportAggrReportStatSumX.
        Note that this object will only be instantiated if the
        agent supports High Capacity monitoring for a particular
        dataSource."
    ::= { reportStatsDataEntry 6 }

reportStatsDataStatMaximum  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The single maximum data point value observed during the
        study period for the specified metric.  This number always
        represents the maximum value of any single statistical
        datum analyzed.  Each metric specifies the exact meaning
        of this object.

        This value represents the results of one metric and is
        related directly to the specific parameters of the metric
        and the Server and Client addresses involved."
    ::= { reportStatsDataEntry 7 }

reportStatsDataStatMinimum  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The single minimum data point value observed during the
        study period for the specified metric.  This number always
        represents the minimum value of any single statistical
        datum analyzed.  Each metric specifies the exact meaning
        of this object.

        This value represents the results of one metric and is
        related directly to the specific parameters of the metric
        and the Server and Client addresses involved."
    ::= { reportStatsDataEntry 8 }

reportStatsDataStatSumSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The sum of all the squared data point values for the
        specified metric.  This number always represents the
        total of the squared values of the statistical datum
        analyzed.  Each metric specifies the exact meaning of
        this object.
        This value represents the results of one metric and is
        related directly to the specific parameters of the metric
        and the Server and Client addresses involved."
    ::= { reportStatsDataEntry 9 }

reportStatsDataOverflowStatSumSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of times the associated
        reportAggrReportStatSumSq counter has overflowed.
        Note that this object will only  be instantiated if
        the associated reportAggrReportHCStatSumSq object
        is also instantiated for a particular dataSource."
    ::= { reportStatsDataEntry 10 }

reportStatsDataHCStatSumSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The high-capacity version of reportAggrReportStatSumSq.
        Note that this object will only be instantiated if the
        agent supports High Capacity monitoring for a particular
        dataSource."
    ::= { reportStatsDataEntry 11 }

reportStatsDataStatSumIX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "For each interval, each data point is associated with a
        value I, I = 1..N where N is the number of data points;
        reportAggrReportStatSumIX is the multiplication of the
        data point value with the current I.  This value
        along with the other statistics values allow the
        calculation of the slope of the least-squares line
        through the data points."
    ::= { reportStatsDataEntry 12 }

reportStatsDataOverflowStatSumIX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of times the associated
        reportAggrReportStatSumIX counter has overflowed.
        Note that this object will only be instantiated if the
        associated reportAggrReportHCStatSumIX object is also
        instantiated for a particular dataSource."
    ::= { reportStatsDataEntry 13 }

reportStatsDataHCStatSumIX  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The high-capacity version of reportAggrReportStatSumIX.
        Note that this object will only be instantiated if the
        agent supports High Capacity monitoring for a particular
        dataSource."
    ::= { reportStatsDataEntry 14 }

reportStatsDataStatSumIXSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "For each interval, each data point is associated with a
        value I, I = 1..N where N is the number of data points;
        reportAggrReportStatSumIXSq is the multiplication
        of the data point value squared with the current I.
        This value along with the other statistics
        values allow the calculation of the slope of
        the least-squares line through the data points."
    ::= { reportStatsDataEntry 15 }

reportStatsDataOverflowStatSumIXSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of times the associated
        reportAggrReportStatSumIXSq counter has overflowed.
        Note that this object will only be instantiated if the
        associated reportAggrReportHCStatSumIXSq object is also
        instantiated for a particular dataSource."
    ::= { reportStatsDataEntry 16 }

reportStatsDataHCStatSumIXSq  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
```

```
            "The high-capacity version of reportAggrReportStatSumIXSq.
            Note that this object will only be instantiated if the
            agent supports High Capacity monitoring for a particular
            dataSource."
        ::= { reportStatsDataEntry 17 }

    reportStatsDataStatMetricExt1  OBJECT-TYPE
        SYNTAX      ZeroBasedCounter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The .... for the MetricExt1.
            "
        ::= { reportStatsDataEntry 18 }

    reportStatsDataStatMetricExt2  OBJECT-TYPE
        SYNTAX      ZeroBasedCounter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The .... for the MetricExt2.
            "
        ::= { reportStatsDataEntry 19 }

    reportStatsDataStatMetricExt3  OBJECT-TYPE
        SYNTAX      ZeroBasedCounter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The .... for the MetricExt3.
            "
        ::= { reportStatsDataEntry 20 }

    reportStatsDataStatMetricExt4  OBJECT-TYPE
        SYNTAX      ZeroBasedCounter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The .... for the MetricExt4.
            "
        ::= { reportStatsDataEntry 21 }

    reportStatsDataStatMetricExt5  OBJECT-TYPE
        SYNTAX      ZeroBasedCounter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The .... for the MetricExt5.
            "
        ::= { reportStatsDataEntry 22 }




    reportSampledGroup       OBJECT IDENTIFIER ::= { reportMIBObjects 2 }

    --      Then, the reportSampledGroup assignments are :
    --             reportSampledControlTable    - 1
    --             reportSampledObjectTable     - 2
    --             reportSampledDataTable       - 3


    -- REPORT-MIB Editors' Note:
    -- The reportSampledGroup is copied from the usrHistory
    -- group documented in RMON2 [RFC2021].  We have perserved all of
    -- the annotations and object descriptions, as any changes would
    -- only diminish the quality of the development.  The only changes
    -- made were to the naming of the objects themselves.  Here we have
    -- merely prefixed the original names with 'report' and changed the
    -- 'usrHistory' to 'Sampled' as we felt this better reflected the
    -- the nature of the capability being offered by this group.
    -- The remainder of this group development is essentially
    -- copied from [RFC2021]:

    --
    -- Sampled Collection Group (reportSampledGroup)
    --
    -- The reportSampled group combines mechanisms seen in the alarm and
    -- history groups to provide user-specified samplying collection,
    -- utilizing two additional control tables and one additional data
    -- table. This function has traditionally been done by NMS
    -- applications, via periodic polling.  The reportSampled group allows
    -- this task to be offloaded to a remote managed device.
    --
    -- Data (an ASN.1 INTEGER based object) is collected in the same
    -- manner as any data table (e.g. etherHistoryTable) except
    -- that the user specifies the MIB instances to be collected and their
    -- sampling frequency. Objects are collected in
    -- bucket-groups, with the intent that all MIB
    -- instances in the same bucket-group are collected as atomically as
    -- possible by the remote managed device.
    --
    -- The reportSampledControlTable is a one-dimensional read-create table.
    -- Each row configures a collection of sampling buckets; the creation
    -- of a row in this table will cause one or more associated instances in
    -- the reportSampledObjectTable to be created. The user specifies the
    -- number of bucket elements (rows in the reportSampledObjectTable)
    -- requested, as well as the number of buckets requested.
    --
    -- The reportSampledObjectTable is a 2-d read-write table.
    -- Each row configures a single MIB instance to be collected.
    -- All rows with the same major index constitute a bucket-group.
    --
    -- The reportSampledTable is a 3-d read-only table containing
    -- the data of associated reportSampledControlEntries. Each
    -- entry represents the value of a single MIB instance
    -- during a specific sampling interval (or the rate of
    -- change during the interval).
    --
    -- A sample value is stored in two objects - an absolute value and
    -- a status object. This allows numbers from -(2G-1) to +4G to be
    -- stored.  The status object also indicates whether a sample is
    -- valid. This allows data collection to continue if periodic
    -- retrieval of a particular instance fails for any reason.
    --
    -- Row Creation Order Relationships
    --
    -- The static nature of the reportSampledObjectTable creates
    -- some row creation/modification issues. The rows in this
    -- table need to be set before the associated
    -- reportSampledControlEntry can be activated.
    --
    -- Note that the reportSampledObject entries associated with a
    -- particular reportSampledControlEntry are not required to
    -- be active before the control entry is activated. However,
    -- the reportSampled data entries associated with an inactive
    -- reportSampledObject entry will be inactive (i.e.
    -- reportSampledValStatus == valueNotAvailable).
    --

    reportSampledControlTable OBJECT-TYPE
        SYNTAX SEQUENCE OF SampledControlEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "A list of data-collection configuration entries."
        ::= { reportSampledGroup 1 }


    reportSampledControlEntry OBJECT-TYPE
        SYNTAX SampledControlEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "A list of parameters that set up a group of user-defined
            MIB objects to be sampled periodically (called a
            bucket-group).

            For example, an instance of reportSampledControlInterval
            might be named reportSampledControlInterval.1"
        INDEX { reportSampledControlIndex }
        ::= { reportSampledControlTable 1 }

    SampledControlEntry ::= SEQUENCE {
        reportSampledControlIndex             Integer32,
        reportSampledControlObjects           Integer32,
        reportSampledControlBucketsRequested  Integer32,
        reportSampledControlBucketsGranted    Integer32,
        reportSampledControlInterval          Integer32,
        reportSampledControlRequestedNumber   Integer32,
        reportSampledControlReportNumber      Integer32,
        reportSampledControlOwner             OwnerString,
        reportSampledControlStatus            RowStatus
    }

    reportSampledControlIndex OBJECT-TYPE
        SYNTAX Integer32 (1..65535)
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "An index that uniquely identifies an entry in the
            reportSampledControlTable.  Each such entry defines a
            set of samples at a particular interval for a specified
            set of MIB instances available from the managed system."
        ::= { reportSampledControlEntry 1 }

    reportSampledControlObjects OBJECT-TYPE
        SYNTAX Integer32 (1..65535)
        MAX-ACCESS read-create
        STATUS current
        DESCRIPTION
            "The number of MIB objects to be collected
            in the portion of reportSampledTable associated with this
            reportSampledControlEntry.

            This object may not be modified if the associated instance
            of reportSampledControlStatus is equal to active(1)."
        ::= { reportSampledControlEntry 2 }

    reportSampledControlBucketsRequested OBJECT-TYPE
```

```
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The requested number of discrete time intervals
        over which data is to be saved in the part of the
        reportSampledTable associated with this
        reportSampledControlEntry.

        When this object is created or modified, the probe
        should set reportSampledControlBucketsGranted as closely to
        this object as is possible for the particular probe
        implementation and available resources."
    DEFVAL { 50 }
    ::= { reportSampledControlEntry 3 }

reportSampledControlBucketsGranted OBJECT-TYPE
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of discrete sampling intervals
        over which data shall be saved in the part of
        the reportSampledTable associated with this
        reportSampledControlEntry.

        When the associated reportSampledControlBucketsRequested
        object is created or modified, the probe should set
        this object as closely to the requested value as is
        possible for the particular probe implementation and
        available resources.  The probe must not lower this
        value except as a result of a modification to the associated
        reportSampledControlBucketsRequested object.

        The associated reportSampledControlBucketsRequested object
        should be set before or at the same time as this object
        to allow the probe to accurately estimate the resources
        required for this reportSampledControlEntry.

        There will be times when the actual number of buckets
        associated with this entry is less than the value of
        this object.  In this case, at the end of each sampling
        interval, a new bucket will be added to the
        reportSampledTable.

        When the number of buckets reaches the value of this object,
        this report is complete and a new report is begun."
    ::= { reportSampledControlEntry 4 }
```

```
reportSampledControlInterval OBJECT-TYPE
    SYNTAX Integer32 (1..2147483647)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The interval in seconds over which the data is
        sampled for each bucket in the part of the reportSampled
        table associated with this reportSampledControlEntry.

        Because the counters in a bucket may overflow at their
        maximum value with no indication, a prudent manager will
        take into account the possibility of overflow in any of
        the associated counters. It is important to consider the
        minimum time in which any counter could overflow on a
        particular media type and set the
        reportSampledControlInterval object to a value less
        than this interval.

        This object may not be modified if the associated
        reportSampledControlStatus object is equal to active(1)."
    DEFVAL { 1800 }
    ::= { reportSampledControlEntry 5 }

reportSampledControlRequestedNumber OBJECT-TYPE
    SYNTAX Integer32 (1..127)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The number of reports to be generated and stored by this
         agent for this report request.

        This object may not be modified if the associated
        reportSampledControlStatus object is equal to active(1)."
    DEFVAL { 1 }
    ::= { reportSampledControlEntry 6 }

reportSampledControlReportNumber OBJECT-TYPE
    SYNTAX Integer32 (1..127)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The number of the current report in progress.  The first
         report is assigned a number equal to '1'.  Each successive
         report number is incremented by unity.  When the last report
         is completed, this value is set to
         reportSampledControlRequestedNumber + 1."
    ::= { reportSampledControlEntry 7 }
```

```
reportSampledControlOwner OBJECT-TYPE
    SYNTAX OwnerString
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The entity that configured this entry and is
        therefore using the resources assigned to it."
    ::= { reportSampledControlEntry 8 }

reportSampledControlStatus OBJECT-TYPE
    SYNTAX RowStatus
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The status of this variable history control entry.

        An entry may not exist in the active state unless all
        objects in the entry have an appropriate value.

        If this object is not equal to active(1), all associated
        entries in the reportSampledTable shall be deleted."
    ::= { reportSampledControlEntry 9 }


-- Object table

reportSampledObjectTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SampledObjectEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of data-collection configuration entries."
    ::= { reportSampledGroup 2 }

reportSampledObjectEntry OBJECT-TYPE
    SYNTAX SampledObjectEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of MIB instances to be sampled periodically.

        Entries in this table are created when an associated
        reportSampledControlObjects object is created.

        The reportSampledControlIndex value in the index is
        that of the associated reportSampledControlEntry.

        For example, an instance of reportSampledObjectVariable
        might be reportSampledObjectVariable.1.3"
    INDEX { reportSampledControlIndex, reportSampledObjectIndex }
    ::= { reportSampledObjectTable 1 }

SampledObjectEntry ::= SEQUENCE {
    reportSampledObjectIndex             Integer32,
    reportSampledObjectVariable          OBJECT IDENTIFIER,
    reportSampledObjectIpAddrType        InetAddressType,
    reportSampledObjectIPAddress         InetAddress,
    reportSampledObjectSampleType        INTEGER
}

reportSampledObjectIndex OBJECT-TYPE
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An index used to uniquely identify an entry in the
        reportSampledObject table.  Each such entry defines a
        MIB instance to be collected periodically."
    ::= { reportSampledObjectEntry 1 }


reportSampledObjectVariable OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The object identifier of the particular variable to be
        sampled.

        Only variables that resolve to an ASN.1 primitive type of
        Integer32 (Integer32, Counter, Gauge, or TimeTicks) may be
        sampled.

        Because SNMP access control is articulated entirely in terms
        of the contents of MIB views, no access control mechanism
        exists that can restrict the value of this object to identify
        only those objects that exist in a particular MIB view.
        Because there is thus no acceptable means of restricting the
        read access that could be obtained through the user history
        mechanism, the probe must only grant write access to this
        object in those views that have read access to all objects on
        the probe.

        During a set operation, if the supplied variable name is not
        available in the selected MIB view, a badValue error must be
        returned.
        This object may not be modified if the associated
        reportSampledControlStatus object is equal to active(1)."
    ::= { reportSampledObjectEntry 2 }

reportSampledObjectIpAddrType  OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportSampledObjectEntry 3 }

reportSampledObjectIPAddress  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportSampledObjectEntry 4 }

reportSampledObjectSampleType OBJECT-TYPE
    SYNTAX INTEGER {
               absoluteValue(1),
               deltaValue(2)
           }
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The method of sampling the selected variable for storage in
        the reportSampledTable.

        If the value of this object is absoluteValue(1), the value of
        the selected variable will be copied directly into the history
        bucket.

        If the value of this object is deltaValue(2), the value of the
        selected variable at the last sample will be subtracted from
        the current value, and the difference will be stored in the
        history bucket. If the associated reportSampledObjectVariable
        instance could not be obtained at the previous sample
        interval, then a delta sample is not possible, and the value
        of the associated reportSampledValStatus object for this
        interval will be valueNotAvailable(1).

        This object may not be modified if the associated
        reportSampledControlStatus object is equal to active(1)."
    ::= { reportSampledObjectEntry 5 }


-- data table

-- Note: Need to think about how to collect this report data.  It
--   is stored in individual buckets containing individual object
--   samples.  Want to avoid having to table walk to collect this
--   information.
reportSampledTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SampledEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of user defined history entries."
    ::= { reportSampledGroup 3 }

reportSampledEntry OBJECT-TYPE
    SYNTAX SampledEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A historical sample of user-defined variables.  This sample
        is associated with the reportSampledControlEntry which set
        up the parameters for a regular collection of these samples.

        The reportSampledControlIndex value in the index identifies
        the reportSampledControlEntry on whose behalf this entry
        was created.

        The reportSampledObjectIndex value in the index identifies
        the reportSampledObjectEntry on whose behalf this entry
        was created.

        For example, an instance of reportSampledAbsValue, which
        represents the 14th sample of a variable collected as
        specified by reportSampledControlEntry.1 and
        reportSampledObjectEntry.1.5, would be named
        reportSampledAbsValue.1.14.5"
    INDEX { reportSampledControlIndex, reportSampledReportIndex,
            reportSampledSampleIndex, reportSampledObjectIndex }
    ::= { reportSampledTable 1 }

SampledEntry ::= SEQUENCE {
    reportSampledReportIndex   Integer32,
    reportSampledSampleIndex   Integer32,
    reportSampledIntervalStart TimeStamp,
    reportSampledIntervalEnd   TimeStamp,
    reportSampledAbsValue      Gauge32,
    reportSampledValStatus     INTEGER
}

reportSampledReportIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..127)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "An index that uniquely identifies the particular report
        this entry is associated with among the set of reports
        requested through the reportSampledControlNumber in the
        reportSampledControlEntry. This index starts at 1 and
        increases by one as each new report is generated."
    ::= { reportSampledEntry 1 }

reportSampledSampleIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..2147483647)
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
        "An index that uniquely identifies the particular sample this
        entry represents among all samples associated with the same
        reportSampledControlEntry. This index starts at 1 and
        increases by one as each new sample is taken."
    ::= { reportSampledEntry 2 }

reportSampledIntervalStart OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of sysUpTime at the start of the interval over
        which this sample was measured.  If the probe keeps track of
        the time of day, it should start the first sample of the
        history at a time such that when the next hour of the day
        begins, a sample is started at that instant.

        Note that following this rule may require the probe to delay
        collecting the first sample of the history, as each sample
        must be of the same interval. Also note that the sample which
        is currently being collected is not accessible in this table
        until the end of its interval."
    ::= { reportSampledEntry 3 }

reportSampledIntervalEnd OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of sysUpTime at the end of the interval over which
        this sample was measured."
    ::= { reportSampledEntry 4 }

reportSampledAbsValue OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The absolute value (i.e. unsigned value) of the
        user-specified statistic during the last sampling period. The
        value during the current sampling period is not made available
        until the period is completed.

        To obtain the true value for this sampling interval, the
        associated instance of reportSampledValStatus must
        be checked, and reportSampledAbsValue adjusted as necessary.

        If the MIB instance could not be accessed during the sampling
        interval, then this object will have a value of zero and the
        associated instance of reportSampledValStatus will be set to
        'valueNotAvailable(1)'."
    ::= { reportSampledEntry 5 }


reportSampledValStatus OBJECT-TYPE
    SYNTAX INTEGER {
        valueNotAvailable(1),
        valuePositive(2),
        valueNegative(3)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This object indicates the validity and sign of the data in
        the associated instance of reportSampledAbsValue.
        If the MIB instance could not be accessed during the sampling
        interval, then 'valueNotAvailable(1)' will be returned.

        If the sample is valid and actual value of the sample is
        greater than or equal to zero then 'valuePositive(2)' is
        returned.

        If the sample is valid and the actual value of the sample is
        less than zero, 'valueNegative(3)' will be returned. The
        associated instance of reportSampledAbsValue should be
        multiplied by -1 to obtain the true sample value."
    ::= { reportSampledEntry 6 }

-- REPORT-MIB Editors' Note:  This ends the copy of definitions from
-- the usrHistory group from RMON2 [RFC 2021].




reportHistoryGroup       OBJECT IDENTIFIER ::= { reportMIBObjects 3 }

--      Then, the reportHistoryGroup assignments are :
--             reportHistoryControlTable    - 1
--             reportHistoryDataTable       - 2

-- Notes: The history group is intended to track changes in
--   identified objects ot type counter, gauge, other.  Each,
--   time the object is updated in the associated MIB, the
--   history group stores a table entry in the associated
--   historyDataTable capturing the time the change was
--   made to the identified object.

--   The historyControl Table ...
--
--   The historyData Table ....

reportHistoryControlTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HistoryControlEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of data-collection configuration entries."
    ::= { reportHistoryGroup 1 }

reportHistoryControlEntry OBJECT-TYPE
    SYNTAX HistoryControlEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of parameters that set up the collection
        of a history of changes
        in the user-defined MIB objects.

        For example, an instance of reportHistoryControlObject
        might be named reportHistoryControlObject.1"
    INDEX { reportHistoryControlIndex }
    ::= { reportHistoryControlTable 1 }

HistoryControlEntry ::= SEQUENCE {
    reportHistoryControlIndex             Integer32,
    reportHistoryControlObject            OBJECT IDENTIFIER,
    reportHistoryControlObjectIpAddrType  InetAddressType,
    reportHistoryControlObjectIPAddress   InetAddress,
    reportHistoryControlSizeRequested     Integer32,
    reportHistoryControlSizeGranted       Integer32,
    reportHistoryControlRequestedNumber   Integer32,
    reportHistoryControlReportNumber      Integer32,
    reportHistoryControlOwner             OwnerString,
    reportHistoryControlStatus            RowStatus
}

reportHistoryControlIndex OBJECT-TYPE
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "An index that uniquely identifies an entry in the
        reportHistoryControlTable.  Each such entry defines a
        set of histories at a particular interval for a specified
        MIB object instance available from the managed system."
    ::= { reportHistoryControlEntry 1 }

reportHistoryControlObject OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The MIB object to be monitored for the collection
        histories in the reportHistoryDataTable associated with this
        reportHistoryControlEntry.

        This object may not be modified if the associated instance
        of reportHistoryControlStatus is equal to active(1)."
    ::= { reportHistoryControlEntry 2 }
```

```
reportHistoryControlObjectIpAddrType  OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP address type
        of the IP address associated with the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportHistoryControlEntry 3 }

reportHistoryControlObjectIPAddress  OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "This identifies the IP addree of the
        secondary counter object to be
        monitored within this report.

        This object may not be modified if the associated
        reportStatsControlStatus object is equal to active(1)."
    ::= { reportHistoryControlEntry 4 }

reportHistoryControlSizeRequested OBJECT-TYPE
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The requested maximum number of history entries
        to be saved in the
        reportHistoryDataTable associated with this
        reportHistoryControlEntry.

        When this object is created or modified, the device
        should set reportHistoryControlSizeGranted as closely to
        this object as is possible for the particular device
        implementation and available resources."
    DEFVAL { 50 }
    ::= { reportHistoryControlEntry 5 }

reportHistoryControlSizeGranted OBJECT-TYPE
    SYNTAX Integer32 (1..65535)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The maximum allowed number of discrete history entries
        in the reportHistoryTable associated with this
        reportHistoryControlEntry.

        When the associated reportHistoryControlSizeRequested
        object is created or modified, the device should set
        this object as closely to the requested value as is
        possible for the particular device implementation and
        available resources.  The device must not lower this
        value except as a result of a modification to the associated
        reportHistoryControlSizeRequested object.

        The associated reportHistoryControlSizeRequested object
        should be set before or at the same time as this object
        to allow the device to accurately estimate the resources
        required for this reportHistoryControlEntry.

        When the number of histories reaches the value of this object
        and a new history is to be added to the reportHistoryTable,
        the oldest history associated with this
        reportHistoryControlEntry shall be deleted by the agent
        so that the new history can be added."
    ::= { reportHistoryControlEntry 6 }

reportHistoryControlRequestedNumber OBJECT-TYPE
    SYNTAX Integer32 (1..127)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The number of reports to be generated and stored by this
         agent for this report request.

        This object may not be modified if the associated
        reportHistoryControlStatus object is equal to active(1)."
    DEFVAL { 1 }
    ::= { reportHistoryControlEntry 7 }

reportHistoryControlReportNumber OBJECT-TYPE
    SYNTAX Integer32 (1..127)
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The number of the current report in progress.  The first
         report is assigned a number equal to '1'.  Each successive
         report number is incremented by unity.  When the last report
         is completed, this value is set to
         reportSampledControlRequestedNumber + 1."
    ::= { reportHistoryControlEntry 8 }

reportHistoryControlOwner OBJECT-TYPE
    SYNTAX OwnerString
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The entity that configured this entry and is
        therefore using the resources assigned to it."
    ::= { reportHistoryControlEntry 9 }

reportHistoryControlStatus OBJECT-TYPE
    SYNTAX RowStatus
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "The status of this variable history control entry.

        An entry may not exist in the active state unless all
        objects in the entry have an appropriate value.

        If this object is not equal to active(1), all associated
        entries in the reportHistoryTable shall be deleted."
    ::= { reportHistoryControlEntry 10 }



-- data table

-- Note: Similar to the note on the sampled report
--   collection above.  We need to consider what
--   model to use to transmit the report data to
--   the remote management application.  Currently
--   the data is stored in individuals events per
--   table row.  This will impact the design of the
--   table as well as the design of the
--   Notifications.
reportHistoryTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HistoryEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of user defined history entries."
    ::= { reportHistoryGroup 3 }

reportHistoryEntry OBJECT-TYPE
    SYNTAX HistoryEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A historical trail of user-defined variables.  This list
        is associated with the reportHistoryControlEntry which set
        up the parameters for a regular collection of these samples.

        The reportHistoryControlIndex value in the index identifies
        the reportHistoryControlEntry on whose behalf this entry
        was created.  This also identifies the MIB object
        being tracked by this reportHistoryEntry.

        For example, an instance of reportHistory...
        "
    INDEX { reportHistoryControlIndex,
            reportHistoryDataIndex }
    ::= { reportHistoryTable 1 }

HistoryEntry ::= SEQUENCE {
    reportHistoryDataIndex        Integer32,
    reportHistoryDataChangeTime   TimeStamp,
    reportHistoryDataValueType    INTEGER,
    reportHistoryDataValue        OCTET STRING,
    reportHistoryDataValStatus    INTEGER
}

reportHistoryDataIndex OBJECT-TYPE
    SYNTAX     Integer32 (1..2147483647)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
        "An index that uniquely identifies the particular sample this
        entry represents among all historical entries
        associated with the same
        reportHistoryControlEntry. This index starts at 1 and
        increases by one as each new sample is taken."
    ::= { reportHistoryEntry 1 }

reportHistoryDataChangeTime   OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of sysUpTime at the time that the MIB object was
        updated."
    ::= { reportHistoryEntry 2 }

-- Note: May want to move this to the reportHistoryControlTable,
--    as it is too redundant in this table.  Also, need to reconsider
--    the best way to indicate type and to represent values.
reportHistoryDataValueType OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The type of the data value stored in the
        reportHistoryDataValue string.  The user identifies
        the MIB object to be tracked by this table.
        Various types of objects can be track, so the
        application needs to know the data type being
        stored.  Types supported include counter, gauge,
        integer, float.
        "
    ::= { reportHistoryEntry 3 }

reportHistoryDataValue  OBJECT-TYPE
    SYNTAX OCTET STRING
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The absolute value of the
        user-specified MIB object tracked by this
        table entry.  This holds the new object
        value following this change in value.

        If the MIB instance could not be accessed ....
        "
    ::= { reportHistoryEntry 4 }

-- Note: Need to consider in detail the ability of the
--   device to track the times of object change in
--   enough detial to be useful.  What happens if the
--   device gets too busy and delays updating MIB object
--   values tracked by this table entry.  Needs more work.
reportHistoryDataValStatus OBJECT-TYPE
    SYNTAX INTEGER {
        valueAvailable(1),
        valueDelayed(2)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This object indicates the validity of the data in
        the associated instance of reportHistoryAbsValue.

        If the MIB instance could not be accessed promptely,
        then 'valueDelayed(2)' will be returned.
        If the sample is valid and actual value of the sample
        was promptly recorded, then 'valueAvailable(1)' is
        returned.
        "
    ::= { reportHistoryEntry 5 }



--
-- Notifications
--

-- NOTE: What is the report transmission model we want to
--       support for this MIB?  Want to minimize chatter
--       on the network.  Potentially want to see if
--       can pack reports into Notifications(?).
--       The statsReports are  formatted in a way to
--       support bulk transmissions.  However, as noted
--       above, the sampledReports and the historyReports
--       are stored as individual measurements per row and
--       storage is continually rotaed as more measurements
--       are made in these two report types.  This
--       may complicate report transmission and
--       Notifications definitions.

-- NOTE:  What notifications do we want for this MIB?
--        Checkout what is done in the APM-MIB for Notifications?
--        Examples may include a) report completion
--                             b) overflow counters exceeded

reportNotificationControl OBJECT IDENTIFIER
                                 ::= {reportMIBNotifications 1}
reportNotificationObjects OBJECT IDENTIFIER
                                 ::= {reportMIBNotifications 2}
reportNotificationStates  OBJECT IDENTIFIER
                                 ::= {reportMIBNotifications 3}


   -- reportNotificationControl

   reportSetNotification OBJECT-TYPE
          SYNTAX       OCTET STRING (SIZE(4))
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A 4-octet string serving as a bit map for
             the notification events defined by the REPORT
             notifications. This object is used to enable
             and disable specific REPORT notifications where
             a 1 in the bit field represents enabled. The
             right-most bit (least significant) represents
             notification 0.

             This object is persistent and when written
             the entity SHOULD save the change to
             non-volatile storage.
             "
           ::= { reportNotificationControl 1 }


   -- reportNotificationObjects

   reportNewStatsDataReport NOTIFICATION-TYPE
          OBJECTS { reportStatsControlIndex, -- The index of the
                             --   control table for this report
                    reportStatsDataIndex     -- The index of the
                             --   data table for this report
                  }
          STATUS       current
          DESCRIPTION
             "reportNewStatsDataReport is a notification sent
              when a new report is completed from the
              reportStatsControlTable.  The notification carries
              the index from the control table that established
              this report and the index from the data table that
              holds this report."
          ::= { reportNotificationObjects 1 }

   reportNewSampledDataReport NOTIFICATION-TYPE
          OBJECTS { reportSampledControlIndex, -- The index of the
                             --   control table for this report
                    reportSampledReportIndex   -- The index of the
                             --   data table for this report
                  }
          STATUS       current
          DESCRIPTION
             "reportNewSampledDataReport is a notification sent
              when a new report is completed from the
              reportSampledControlTable.  The notification carries
              the index from the control table that established
              this report and the index from the data table that
              holds this report.  Indication of the new report
              is when the reportSampledControlReportNumber
              is incremented."
          ::= { reportNotificationObjects 2 }
   reportNewHistoryDataReport NOTIFICATION-TYPE
          OBJECTS { reportHistoryControlIndex, -- The index of the
                             --   control table for this report
                    reportHistoryDataIndex   -- The index of the
                             --   data table for this report
                  }
          STATUS       current
          DESCRIPTION
             "reportNewHistoryDataReport is a notification sent
              when a new report is completed from the
              reportHistoryControlTable.  The notification carries
              the index from the control table that established
              this report and the index from the data table that
              holds this report.  Indication of the new report
              is when the reportHistoryControlReportNumber
              is incremented."
          ::= { reportNotificationObjects 3 }


    -- reportNotificationStates
    --   none to define



--
-- Compliance Statements
--

-- [NOTE: Current thoughts on Conformance follow:
--    Mandatory for Stats will include no extensions,
--    or high capacity objects.
--    Hence, the reports will have only the hard-coded statistics.
--    Optional for Stats will be extensions definition table and high
--    capacity objects.
--
--    Mandatory for Sampled will include all.
--
--    Mandatory for History will include all.]


reportCompliances  OBJECT IDENTIFIER ::= { reportMIBConformance 1 }
reportMIBGroups    OBJECT IDENTIFIER ::= { reportMIBConformance 2 }

reportStatsBasicCompliance  MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION "The Stats basic implementation requirements for
                managed network entities that implement
                the REPORT process."
   MODULE  -- this module
   MANDATORY-GROUPS {reportStatsCapabilitiesBaseObjectsGroup,
                     reportStatsControlBaseObjectsGroup,
                     reportStatsDataBaseObjectsGroup,
                     reportNotificationGroup,
                     reportStatsNotificationGroup }
::= { reportCompliances 1 }

reportStatsHCCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION "The HC implementation requirements for
                managed network entities that implement
                the REPORT process."
   MODULE  -- this module
   MANDATORY-GROUPS {reportStatsCapabilitiesBaseObjectsGroup,
                     reportStatsControlBaseObjectsGroup,
                     reportStatsDataBaseObjectsGroup,
                     reportNotificationGroup,
                     reportStatsNotificationGroup,
                     reportStatsDataHCObjectsGroup }
::= { reportCompliances 2 }

reportStatsExtendedMetricsCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION "The extended metrics implementation requirements for
                managed network entities that implement
                the REPORT process."
   MODULE  -- this module
   MANDATORY-GROUPS {reportStatsCapabilitiesBaseObjectsGroup,
                     reportStatsControlBaseObjectsGroup,
                     reportStatsDataBaseObjectsGroup,
                     reportNotificationGroup,
                     reportStatsNotificationGroup,
                     reportStatsExtendedMetricsCapabilitiesObjectsGroup,
                     reportStatsExtendedMetricsControlObjectsGroup,
                     reportStatsExtendedMetricsDataObjectsGroup }
::= { reportCompliances 3 }

reportSampledBasicCompliance  MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION "The Sampled basic implementation requirements for
                managed network entities that implement
                the REPORT process."
   MODULE  -- this module
   MANDATORY-GROUPS {reportSampledControlBaseObjectsGroup,
                     reportSampledObjectIDBaseObjectsGroup,
                     reportSampledDataBaseObjectsGroup,
                     reportNotificationGroup,
                     reportSampledNotificationGroup }
::= { reportCompliances 4 }


reportHistoryBasicCompliance  MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION "The History basic implementation requirements for
                managed network entities that implement
                the REPORT process."
   MODULE  -- this module
   MANDATORY-GROUPS {reportHistoryControlBaseObjectsGroup,
                     reportHistoryDataBaseObjectsGroup,
                     reportNotificationGroup,
                     reportHistoryNotificationGroup }
::= { reportCompliances 5 }



-- Units of Conformance

reportStatsCapabilitiesBaseObjectsGroup OBJECT-GROUP
   OBJECTS {
            reportClockResolution,
            reportClockMaxSkew,
            reportClockSource
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT configuration objects implemented
       in this module."
::= { reportMIBGroups 1 }

reportStatsControlBaseObjectsGroup OBJECT-GROUP
   OBJECTS {
            reportStatsControlIndex,
            reportStatsControlInterval,
            reportStatsControlBinInterval,
            reportStatsControlPriObjID,
            reportStatsControlPriObjIpAddrType,
            reportStatsControlPriObjIPAddr,
            reportStatsControlReqReports,
            reportStatsControlGrantedReports,
            reportStatsControlStartTime,
            reportStatsControlReportNumber,
            reportStatsControlInsertsDenied,
            reportStatsControlOwner,
            reportStatsControlStorageType,
            reportStatsControlStatus
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT Stats Control base objects implemented
       in this module."
::= { reportMIBGroups 2 }

reportStatsDataBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportStatsDataIndex,
            reportStatsDataStatN,
            reportStatsDataStatSumX,
            reportStatsDataOverflowStatSumX,
            reportStatsDataStatMaximum,
            reportStatsDataStatMinimum,
            reportStatsDataStatSumSq,
            reportStatsDataOverflowStatSumSq,
            reportStatsDataStatSumIX,
            reportStatsDataOverflowStatSumIX,
            reportStatsDataStatSumIXSq,
            reportStatsDataOverflowStatSumIXSq
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 3 }

reportNotificationGroup  OBJECT-GROUP
   OBJECTS {
            reportSetNotification
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT notifications implemented
       in this module for the Statistics reports."
::= { reportMIBGroups 4 }

reportStatsNotificationGroup  NOTIFICATION-GROUP
   NOTIFICATIONS {
            reportNewStatsDataReport
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT notifications implemented
       in this module for the Statistics reports."
::= { reportMIBGroups 5 }
```

```
reportStatsDataHCObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportStatsDataHCStatSumX,
            reportStatsDataHCStatSumSq,
            reportStatsDataHCStatSumIX,
            reportStatsDataHCStatSumIXSq
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 6 }

reportStatsExtendedMetricsCapabilitiesObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportMetricExtDefType,
            reportMetricExtDefName,
            reportMetricExtDefOperation,
            reportMetricExtDefReference,
            reportMetricDirLastChange
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 7 }

reportStatsExtendedMetricsControlObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportStatsControlSecObj1ID,
            reportStatsControlSecObj1IpAddrType,
            reportStatsControlSecObj1IPAddr,
            reportStatsControlSecObj2ID,
            reportStatsControlSecObj2IpAddrType,
            reportStatsControlSecObj2IPAddr,
            reportStatsControlSecObj3ID,
            reportStatsControlSecObj3IpAddrType,
            reportStatsControlSecObj3IPAddr,
            reportStatsControlSecObj4ID,
            reportStatsControlSecObj4IpAddrType,
            reportStatsControlSecObj4IPAddr,
            reportStatsControlSecObj5ID,
            reportStatsControlSecObj5IpAddrType,
            reportStatsControlSecObj5IPAddr,
            reportStatsControlMetricExt1,
            reportStatsControlMetricExt2,
            reportStatsControlMetricExt3,
            reportStatsControlMetricExt4,
            reportStatsControlMetricExt5
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 8 }

reportStatsExtendedMetricsDataObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportStatsDataStatMetricExt1,
            reportStatsDataStatMetricExt2,
            reportStatsDataStatMetricExt3,
            reportStatsDataStatMetricExt4,
            reportStatsDataStatMetricExt5
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 9 }

reportSampledControlBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportSampledControlIndex,
            reportSampledControlObjects,
            reportSampledControlBucketsRequested,
            reportSampledControlBucketsGranted,
            reportSampledControlInterval,
            reportSampledControlRequestedNumber,
            reportSampledControlReportNumber,
            reportSampledControlOwner,
            reportSampledControlStatus
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 10 }

reportSampledObjectIDBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportSampledObjectVariable,
            reportSampledObjectIpAddrType,
            reportSampledObjectIPAddress,
            reportSampledObjectSampleType
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 11 }

reportSampledDataBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportSampledReportIndex,
            reportSampledIntervalStart,
            reportSampledIntervalEnd,
            reportSampledAbsValue,
            reportSampledValStatus
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 12 }

reportSampledNotificationGroup  NOTIFICATION-GROUP
   NOTIFICATIONS {
            reportNewSampledDataReport
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT notifications implemented
       in this module for the Sampled reports."
::= { reportMIBGroups 13 }

reportHistoryControlBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportHistoryControlIndex,
            reportHistoryControlObject,
            reportHistoryControlObjectIpAddrType,
            reportHistoryControlObjectIPAddress,
            reportHistoryControlSizeRequested,
            reportHistoryControlSizeGranted,
            reportHistoryControlRequestedNumber,
            reportHistoryControlReportNumber,
            reportHistoryControlOwner,
            reportHistoryControlStatus
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 14 }
```

```
reportHistoryDataBaseObjectsGroup  OBJECT-GROUP
   OBJECTS {
            reportHistoryDataIndex,
            reportHistoryDataChangeTime,
            reportHistoryDataValueType,
            reportHistoryDataValue,
            reportHistoryDataValStatus
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT state objects implemented
       in this module."
::= { reportMIBGroups 15 }

reportHistoryNotificationGroup  NOTIFICATION-GROUP
   NOTIFICATIONS {
            reportNewHistoryDataReport
   }
   STATUS  current
   DESCRIPTION
      "Set of REPORT notifications implemented
       in this module for the History reports."
::= { reportMIBGroups 16 }



END
```

8. Security Considerations

[TODO] Each specification that defines one or more MIB modules MUST contain a section that discusses security considerations relevant to those modules. This section MUST be patterned after the latest approved template (available at http://www.ops.ietf.org/mib-security.html). Remember that the objective is not to blindly copy text from the template, but rather to think and evaluate the risks/vulnerabilities and then state/ document the result of this evaluation.

[TODO] if you have any read-write and/or read-create objects, please include the following boilerplate paragraph.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- o [TODO] writable MIB objects that could be especially disruptive if abused MUST be explicitly listed by name and the associated security risks MUST be spelled out; RFC 2669 has a very good example.

- o [TODO] list the writable tables and objects and state why they are sensitive.

[TODO] else if there are no read-write objects in your MIB module, use the following boilerplate paragraph.

There are no management objects defined in this MIB module that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB module is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB module via direct SNMP SET operations.

[TODO] if you have any sensitive readable objects, please include the following boilerplate paragraph.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- o [TODO] you must explicitly list by name any readable objects that are sensitive or vulnerable and the associated security risks MUST be spelled out (for instance, if they might reveal customer information or violate personal privacy laws such as those of the European Union if exposed to unauthorized parties)

- o [TODO] list the tables and objects and state why they are sensitive.

[TODO] discuss what security the protocol used to carry the information should have. The following three boilerplate paragraphs should not be changed without very good reason. Changes will almost certainly require justification during IESG review.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

9. IANA Considerations

[TODO] In order to comply with IESG policy as set forth in http://www.ietf.org/ID-Checklist.html, every Internet-Draft that is submitted to the IESG for publication MUST contain an IANA Considerations section. The requirements for this section vary depending what actions are required of the IANA. see RFC4181 section 3.5 for more information on writing an IANA clause for a MIB module document.

[TODO] select an option and provide the necessary details.

Option #1:

```
     The MIB module in this document uses the following IANA-assigned
     OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

     Descriptor        OBJECT IDENTIFIER value
     ----------        -----------------------

     sampleMIB  { mib-2 XXX }
```

Option #2:

Editor's Note (to be removed prior to publication): the IANA is requested to assign a value for "XXX" under the 'mib-2' sub-tree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

Note well: prior to official assignment by the IANA, a draft document MUST use placeholders (such as "XXX" above) rather than actual numbers. See RFC4181 Section 4.5 for an example of how this is done in a draft MIB module.

Option #3:

This memo includes no request to IANA.

10. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

11. Acknowledgements

We would like to thank Bert Wijnen and Andy Bierman for pointing out the existence of the usrHistory group within RMON2 and in answering our numerous questions on the usrHistory group. Further, we wish to thank U. Herberg for his forcing additions to this MIB through his thoughtful consideration of performance monitoring requirements for other MIBs, e.g., NHDP and OLSR MIBs.

12. References

12.1. Normative References

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

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

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

12.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC1757] Waldbusser, S., "Remote Network Monitoring Management Information Base", RFC 1757, February 1995.

[RFC2021] Waldbusser, S., "Remote Network Monitoring Management Information Base Version 2 using SMIv2", RFC 2021, January 1997.

[RFC4150] Dietz, R. and R. Cole, "Transport Performance Metrics MIB", RFC 4150, August 2005.

Appendix A. Change Log

Changes from draft-ietf-manet-report-mib-00 to draft-ietf-manet-report-mib-01 draft.

1. Proposed additions to the statsReports in order to potentially simplify data transmission to management applications.

2. Added some Notification definitions and their relationship to the three reports' structure, i.e., statsReports, sampledReports, and historyReports.

3. In the process of adding notifications for the Sampled and the History reports, decided to restructure the reports from their previously rolling storage model to the fixed interval reporting used all along in the Statistics reporting. This allows the agent to notify the management application that a report has completed and that it is ready to be pulled from the agent storage.

4. Ran MIB through smilint checker and cleaned up all errors and most warnings. A few warnings remain to be addressed.

5. Cleaned up textual material.

Changes from draft-cole-manet-report-mib-02 to draft-ietf-manet-report-mib-00 draft.

1. Major change was the incorporation of the IP address objects associated with all objects of type 'OBJECT IDENTIFIER'. This allows the REPORT-MIB to exist as a proxy report generation capability on a device separate but in close proximity to the device monitoring the referenced object.

2. Cleaned up the up front text, reducing the repetition with the object descriptions in the MIB.

3. Worked on and added sections discussing the relationship to other MIBs.

Changes from draft-cole-manet-report-mib-01 to draft-cole-manet-report-mib-02 draft.

1. Restructured the MIB somewhat to now offer the three reporting capabilities in increasing order of detail: a) statistical reports, b) sampled reports, and c) historical reports.

2. Renamed the usrHistoryGroup and elements to samplingGroup. This is in line with its actual capabilities.

3. Added a new historyGroup which provides a history of change events.

4. Updated the4 Conformance section to reflect the above changes and additions. But did not yet run smilint to check MIB syntax.

Changes from draft-cole-manet-report-mib-00 to draft-cole-manet-report-mib-01 draft.

1. Added (copied) the usrHistory group from RMON2 into the REPORT-MIB.

2. Restructured the MIB to account for the inclusion of the reportSampledGroup.

3. Dropped the reportCurReportsTable as this did not make sense within the context of the REPORT-MIB.

4. Added the Compliance and Conformance material. Defined several Compliance Groups to all for base implementations of the REPORT-MIB for only statistical reports, for only historical reports or for both. Allow for enhanced implementations to address higher capacity issues and extension to metric reporting for statistical reporting.

5. Ran the MIB through the smilint checker and in the process corrected numerous typos, omissions, TEXTUAL CONVENTIONS, IMPORTS, etc.

6. Updated main text to reflect changes.

Appendix B. Open Issues

This section contains the set of open issues related to the development and design of the REPORT-MIB. This section will not be present in the final version of the MIB and will be removed once all the open issues have been resolved.

1. Need to add an index associated with object IDs of interest which are contained within a table, e.g., IfPacketsIn in an InterfaceTable which is indexed by IfIndex. (Note: (RGC)I think adding the IP address associated with the referenced object addresses this issue.)

2. Complete notification group. Need to develop the preferred data report transmission model. This will influence the design of the Notifications. The initial form for the notifications has been laid out in draft-ietf-manet-report-mib-02.

3. Update the text of the document to reflect the final state of the MIB.

4. Identify all objects requiring non-volatile storage in their DESCRIPTION clauses.

5. Complete the security analysis and section.

6. Cleanup all the [TODOs] from the MIB template.

Appendix C.

```
   **************************************************************
   * Note to the RFC Editor (to be removed prior to publication) *
   *                                                            *
   * 1) The reference to RFCXXXX within the DESCRIPTION clauses  *
   * of the MIB module point to this draft and are to be        *
   * assigned by the RFC Editor.                                *
   *                                                            *
   * 2) The reference to RFCXXX2 throughout this document point  *
   * to the current draft-ietf-manet-report-xx.txt.  This       *
   * need to be replaced with the XXX RFC number.               *
   *                                                            *
   **************************************************************
```

Authors' Addresses

Robert G. Cole
US Army CERDEC
328 Hopkins Road
Aberdeen Proving Ground, Maryland 21005
USA

Phone: +1 410 278 6779
EMail: robert.g.cole@us.army.mil
URI: http://www.cs.jhu.edu/˜rgcole/

Joseph Macker
Naval Research Laboratory
Washington, D.C. 20375
USA

EMail: macker@itd.nrl.navy.mil

Al Morton
AT&T Laboratories
Middletown, N.J. 07724
USA

EMail: amorton@att.com

Internet Engineering Task Force R. Cole
Internet-Draft US Army CERDEC
Intended status: Standards Track J. Macker
Expires: July 20, 2011 B. Adamson
Naval Research Laboratory
S. Harnedy
Booz Allen Hamilton
January 16, 2011

Definition of Managed Objects for the Manet Simplified Multicast Framework Relay Set Process
draft-ietf-manet-smf-mib-02

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of the Simplified Multicast Forwarding (SMF) process for Mobile Ad-Hoc Networks (MANETs). The SMF-MIB also reports state information, performance metrics, and notifications. In addition to configuration, the additional state and performance information is useful to operators troubleshooting multicast forwarding problems.

Status of This Memo

This Internet-Draft will expire on July 20, 2011.

Copyright Notice

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of a process implementing Simplified Multicast Forwarding (SMF) [I-D.ietf-manet-smf] for Mobile Ad-Hoc Networks (MANETs). SMF provides multicast Duplicate Packet Detection (DPD) and supports algorithms for constructing an estimate of a MANET Minimum Connected Dominating Set (MCDS) for efficient multicast forwarding. The SMF-MIB also reports state information, performance metrics, and notifications. In addition to configuration, this additional state and performance information is useful to operators troubleshooting multicast forwarding problems.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Conventions

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].

4. Overview

SMF provides methods for implementing DPD-based multicast forwarding with the optional use of Connected Dominating Set (CDS)-based relay sets. The CDS provides a complete connected coverage of the nodes comprising the MANET. The MCDS is the smallest set of MANET nodes (comprising a connected cluster) which cover all the nodes in the cluster with their transmissions. As the density of the MANET nodes increase, the fraction of nodes required in an MCDS decreases. Using the MCDS as a multicast forwarding set then becomes an efficient multicast mechanism for MANETs.

Various algorithms for the construction of estimates of the MCDS exist. The Simplified Multicast Framework [I-D.ietf-manet-smf] describes some of these. It further defines various operational modes for a node which is participating in the collective creation of the MCDS estimates. These modes depend upon the set of related MANET routing and discovery protocols and mechanisms in operation in the specific MANET node.

A SMF router's MIB contains SMF process configuration parameters (e.g. specific CDS algorithm), state information (e.g., current membership in the CDS), performance counters (e.g., packet counters), and notifications.

4.1. SMF Management Model

This section describes the management model for the SMF node process.

Figure 1 (reproduced from Figure 4 of [I-D.ietf-manet-smf]) shows the relationship between the SMF Relay Set selection algorithm and the related algorithms, processes and protocols running in the MANET nodes. The Relay Set Selection Algorithm (RSSA) can rely upon topology information gotten from the MANET Neighborhood Discovery Protocol (NHDP), from the specific MANET routing protocol running on the node, or from Layer 2 information passed up to the higher layer protocol processes.

RGC Note: update this figure from the latest SMF draft.

```
       ______________                ____________
      |              |              |            |
      | Neighborhood |              | Relay Set  |
      |  Discovery   |------------->| Selection  |
      |   Protocol   |   neighbor   | Algorithm  |
      |______________|     info     |____________|
             \                               /
              \                             /
       neighbor\                           / forwarding
         info*  \      _____________      /    status
                 \    |             |    /
                  `-->| Forwarding  |<--'
                      |   Process   |
    ----------------->|_____________|----------------->
     incoming packet,                   forwarded packets
     interface id*, and
     previous hop*
```

Figure 1: SMF Node Architecture

4.2. Terms

The following definitions apply throughout this document:

- Configuration Objects - switches, tables, objects which are initialized to default settings or set through the management interface defined by this MIB.

- Tunable Configuration Objects - objects whose values affect timing or attempt bounds on the SMF RS process.

- State Objects - automatically generated values which define the current operating state of the SMF RS process in the router.

- Performance Objects - automatically generated values which help an administrator or automated tool to assess the performance of the CDS multicast process on the router and the overall multicasting performance within the MANET routing domain.

5. Structure of the MIB Module

This section presents the structure of the SMF-MIB module. The objects are arranged into the following groups:

- smfMIBNotifications - defines the notifications associated with the SMF-MIB.

- smfMIBObjects - defines the objects forming the basis for the SMF-MIB. These objects are divided up by function into the following groups:

-
  * Capabilities Group - This group contains the SMF objects that the device uses to advertise its local capabilities with respect to, e.g., the supported RSSAs.

  * Configuration Group - This group contains the SMF objects that configure specific options that determine the overall operation of the SMF RSSA and the resulting multicast performance.

  * State Group - Contains information describing the current state of the SMF RSSA process such as the Neighbor Table.

  * Performance Group - Contains objects which help to characterize the performance of the SMF RSSA process, typically statistics counters.

- smfMIBConformance - defines minimal and full conformance of implementations to this SMF-MIB.

5.1. Textual Conventions

The textual conventions defined within the SMF-MIB are as follows:

- The SmfStatus is defined within the SMF-MIB. This contains the current operational status of the SMF process on an interface.

- The SmfOpModeID represents an index that identifies a specific SMF operational mode.

- The SmfRssaID represents an index that identifies, through reference, a specific RSSA available for operation on the device.

5.2. The Capabilities Group

The SMF device supports a set of capabilities. The list of capabilities which the device can advertise are:

- Operational Mode - topology information from NHDP, CDS-aware unicast routing or Cross-layer from Layer 2.

- SMF RSSA - the specific RSSA operational on the device. Note that configuration, state and performance objects related to a specific RSSA must be defined within another separate MIB.

5.3. The Configuration Group

The SMF device is configured with a set of controls. Some of the prominent configuration controls for the SMF device follow:

- o Operational Mode - topology information from NHDP, CDS-aware unicast routing or Cross-layer from Layer 2.
- o SMF RSSA - the specific RSSA operational on the device.
- o Duplicate Packet detection for IPv4 - Identification-based or Hash-based DPD.
- o Duplicate Packet detection for IPv6 - Identification-based or Hash-based DPD.
- o SMF Type Message TLV - if NHDP mode is selected, then is the SMF Type Message TLV may be included in the NHDP exchanges.
- o SMF Address Block TLV - if NHDP mode is selected, then is the SMF Address Block TLV included in the NHDP exchanges. (Note: is this correct?)

5.4. The State Group

The State Subtree reports current state information, e.g.,

- o Node RSS State - is the node currently in or out of the Relay Set.
- o Neighbors Table - a table containing current neighbors and their operational RSSA.

5.5. The Performance Group

The Performance subtree reports primarily counters that relate to SMF RSSA performance. The SMF performance counters consists of per node and per interface objects:

- o Total multicast packets received.
- o Total multicast packets forwarded.
- o Total duplicate multicast packets detected.
- o Per interface statistics table with the following entries:
- o

- Multicast packets received.

- Multicast packets forwarded.

- Duplicate multicast packets detected.

5.6. The Notifications Group

The Notifications Subtree contains the list of notifications supported within the SMF-MIB and their intended purpose or utility.

6. Relationship to Other MIB Modules

[TODO]: The text of this section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. Definitions imported from other MIB modules and other MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

6.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The SMF-MIB does not duplicate those objects.

6.2. MIB modules required for IMPORTS

The textual conventions imported for use in the SMF-MIB are as follows. The MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, Counter32, Unsigned32, Integer32 and mib-2 textual conventions are imported from RFC 2578 [RFC2578]. The TEXTUAL-CONVENTION, RowStatus and TruthValue textual conventions are imported from RFC 2579 [RFC2579]. The MODULE-COMPLIANCE, OBJECT-GROUP and NOTIFICATION-GROUP textual conventions are imported from RFC 2580 [RFC2580]. The InterfaceIndexOrZero textual convention is imported from RFC 2863 [RFC2863]. The SnmpAdminString textual convention is imported from RFC 3411 [RFC3411]. The InetAddress, InetAddressType and InetAddressPrefixLength textual conventions are imported from RFC 4001 [RFC4001].

6.3. Relationship to the Future RSSA-MIBs

In a sense, the SMF-MIB is a general front-end to a set of, yet to be developed, RSSA-specific MIBs. These RSSA-specific MIBs will define the objects for the configuration, state, performance and notification objects required for the operation of these specific RSSAs. The SMF-MIB Capabilities Group allows the remote management station the ability to query the router to discover the set of supported RSSAs.

7. Definitions

```
   MANET-SMF-MIB DEFINITIONS ::= BEGIN

   IMPORTS

      MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
      Counter32, Unsigned32, Integer32, TimeTicks, mib-2
         FROM SNMPv2-SMI                            -- [RFC2578]

      TEXTUAL-CONVENTION, RowStatus, TruthValue
         FROM SNMPv2-TC                             -- [RFC2579]

      MODULE-COMPLIANCE, OBJECT-GROUP,
      NOTIFICATION-GROUP
         FROM SNMPv2-CONF                           -- [RFC2580]

      InterfaceIndexOrZero
         FROM IF-MIB                                -- [RFC2863]

      SnmpAdminString
         FROM SNMP-FRAMEWORK-MIB                    -- [RFC3411]

      InetAddress, InetAddressType,
      InetAddressPrefixLength
         FROM INET-ADDRESS-MIB                      -- [RFC4001]
      ;

   manetSmfMIB MODULE-IDENTITY
      LAST-UPDATED "201101161300Z"  -- January 16, 2011
      ORGANIZATION "IETF MANET Working Group"
      CONTACT-INFO
         "WG E-Mail: manet@ietf.org

          WG Chairs: ian.chakeres@gmail.com
                     jmacker@nrl.navy.mil


          Editors:   Robert G. Cole
                     US Army CERDEC
                     Space and Terrestrial Communications
                     328 Hopkins Road
                     Bldg 245, Room 16
                     Aberdeen Proving Ground, MD 21005
                     USA
                     +1 410 278-6779
                     robert.g.cole@us.army.mil
                     http://www.cs.jhu.edu/~rgcole/

                     Joseph Macker
                     Naval Research Laboratory
                     Washington, D.C. 20375
                     USA
                     macker@itd.nrl.navy.mil

                     Brian Adamson
                     Naval Research Laboratory
                     Washington, D.C. 20375
                     USA
                     adamson@itd.nrl.navy.mil

                     Sean Harnedy
                     Booz Allen Hamilton
                     333 City Boulevard West
                     Orange, CA 92868
                     USA
                     +1 714 938-3898
                     harnedy_sean@bah.com"

      DESCRIPTION
         "This MIB module contains managed object definitions for
          the Manet SMF RSSA process defined in:

          [SMF] Macker, J.(ed.),
          Simplified Multicast Forwarding draft-ietf-manet-smf-10,
          March 06, 2010.

          Copyright (C) The IETF Trust (2008). This version
          of this MIB module is part of RFC xxxx; see the RFC
          itself for full legal notices."

        -- Revision History
        REVISION    "201101161300Z"   -- January 16, 2011
        DESCRIPTION
           "Updated 5th revision of the
            draft of this MIB module published as
            draft-ietf-manet-smf-mib-02.txt. The changes
            made in this revision include:
              - Added the Notification Group and cleaned
```

```
                up the Conformance section
              - Completed the TEXTUAL CONVENTION for the
                smfOpMode.
              - Completed the Description clauses of
                several objects within the MIB.
              - Removed the routerPriority object.
              - Added the definition of a smfRouterID
                object and associated smfRouterIDAddrType
                object.
            "
        REVISION    "200910261300Z"   -- October 26, 2009
        DESCRIPTION
           "Updated draft of this MIB module published as
            draft-ietf-manet-smf-mib-01.txt. A few changes
            were made in the development of this draft.
            Specifically, the following changes were made:
               - Updated the textual material, included
                 section on IMPORTS, relationship to other
                 MIBs, etc.
           "
        REVISION    "200904211300Z"   -- April 21, 2009
        DESCRIPTION
           "Updated draft of this MIB module published as
            draft-ietf-manet-smf-mib-00.txt. A few changes
            were made in the development of this draft.
            Specifically, the following changes were made:
               - Removed the smfGatewayFilterTable from this
                 draft.  It is a useful construct, e.g.,
                 an IPTABLES-MIB, but might best be handled
                 as a seperate MIB and worked within a
                 security focused working group.
               - Removed the smfReportsGroup. This capability
                 is being replaced with a new and more general
                 method for offline reporting.  This is being
                 worked as a new MIB module refered to as the
                 REPORT-MIB.
               - Rev'd as a new MANET WG document.
           "
        REVISION    "200902271300Z"   -- February 27, 2009
        DESCRIPTION
           "Updated draft of this MIB module published as
            draft-cole-manet-smf-mib-02.txt. Fairly extensive
            revisions and additions to this MIB were made
            in this version. Specifically, the following
            changes were made in development of this version:
               - added a Capabilities Group within the Objects
                 Group to allow the device to report supported
                 capabilities, e.g., RSSAs supported.
               - added administrative status objects for device
                 and interfaces
               - added multicast address forwarding tables, both
                 for configured (within Configuration Group) and
                 discovered (within the State Group).
               - added additional Performance counters related
                 to DPD functions.
               - Split up the performance counters into IPv4
                 and IPv6, for both global and per interface
                 statistics.
               - Split out the reports capability into a seperate
                 Reports Group under the Objects Group.
           "
        REVISION    "200811031300Z"   -- November 03, 2008
        DESCRIPTION
           "Updated draft of this MIB module published as
            draft-cole-manet-smf-mib-01.txt. Added gateway filter
            table and reports capabilities following rmon."
        REVISION    "200807071200Z"   -- July 07, 2008
        DESCRIPTION
           "Initial draft of this MIB module published as
            draft-cole-manet-smf-mib-00.txt."
        -- RFC-Editor assigns XXXX
        ::= { mib-2 998 }   -- to be assigned by IANA



   --
   -- TEXTUAL CONVENTIONs
   --

   SmfStatus ::= TEXTUAL-CONVENTION
       STATUS       current
       DESCRIPTION
          "An indication of the operability of a SMF
          function or feature.  For example, the status
          of an interface: 'enabled' indicates that
          it is performing SMF functions,
          and 'disabled' indicates that it is not."
       SYNTAX  INTEGER {
                        enabled (1),
                        disabled (2)
               }

   SmfOpModeID ::= TEXTUAL-CONVENTION
       STATUS       current
       DESCRIPTION
           "An index that identifies through reference to a specific
```

```
            SMF operations mode.  There are basically three styles
            of SMF operation with reduced relay sets:

              Independent operation - SMF performs its own relay
                  set selection using information from an associated
                  MANET NHDP process.

              CDS-aware unicast routing operation - a coexistent
                  unicast routing protocol provides dynamic relay
                  set state based upon its own control plane
                  CDS or neighborhood discovery information.

              Cross-layer operation -  SMF operates using
                  neighborhood status and triggers from a
                  cross-layer information base for dynamic relay
                  set selection and maintenance
           "
       SYNTAX  INTEGER {
                         independent (1),
                         routing (2),
                         crossLayer (3)
                         -- future (4-255)
                }

   SmfRssaID ::= TEXTUAL-CONVENTION
       STATUS       current
       DESCRIPTION
           "An index that identifies through reference to a specific
            RSSA algorithms.  Several are currently defined
            in the appendix of
           "
       SYNTAX      INTEGER {
                            cF(1),
                            sMPR(2),
                            eCDS(3),
                            mprCDS(4)
                            -- future(5-127)
                            -- noStdAction(128-239)
                            -- experimental(240-255)
                    }



   --
   -- Top-Level Object Identifier Assignments
   --

   smfMIBNotifications OBJECT IDENTIFIER ::= { manetSmfMIB 0 }
   smfMIBObjects       OBJECT IDENTIFIER ::= { manetSmfMIB 1 }
   smfMIBConformance   OBJECT IDENTIFIER ::= { manetSmfMIB 2 }



   --
   -- smfMIBObjects Assignments:
   --      smfCapabilitiesGroup  - 1
   --      smfConfigurationGroup - 2
   --      smfStateGroup         - 3
   --      smfPerformanceGroup   - 4
   --

   --
   -- smfCapabilitiesGroup
   --
   --    This group contains the SMF objects that identify specific
   --    capabilities within this device related to SMF functions.
   --

   smfCapabilitiesGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 1 }

  --
   -- SMF Operational Mode Capabilities Table
   --

   smfOpModeCapabilitiesTable OBJECT-TYPE
       SYNTAX      SEQUENCE OF SmfOpModeCapabilitiesEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "The smfOpModeCapabilitiesTable identifies the
            resident set of SMF Operational Modes on this
            router.
           "
       ::= { smfCapabilitiesGroup 1 }

   smfOpModeCapabilitiesEntry OBJECT-TYPE
       SYNTAX      SmfOpModeCapabilitiesEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "Information about a particular operational
            mode.
           "
       INDEX   { smfOpModeCapabilitiesID }
       ::= { smfOpModeCapabilitiesTable 1 }
   SmfOpModeCapabilitiesEntry ::= SEQUENCE {
         smfOpModeCapabilitiesID              SmfOpModeID,
         smfOpModeCapabilitiesName            SnmpAdminString,
         smfOpModeCapabilitiesReference       SnmpAdminString
   }

   smfOpModeCapabilitiesID     OBJECT-TYPE
       SYNTAX      SmfOpModeID
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "The index for this entry.  This object identifies
            the particular operational mode for this device.
           "
       ::= { smfOpModeCapabilitiesEntry 1 }

   smfOpModeCapabilitiesName OBJECT-TYPE
       SYNTAX      SnmpAdminString
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
           "The textual name of this operational
            mode.  Current operational modes include:
            Independent Mode, CDS-aware Routing Mode,
            and Cross-layer Mode.  Others may be defined
            in future revisions of [SMF].
           "
       ::= { smfOpModeCapabilitiesEntry 2 }

   smfOpModeCapabilitiesReference OBJECT-TYPE
       SYNTAX      SnmpAdminString
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
           "This object contains a reference to the document that
            defines this operational mode.
           "
       ::= { smfOpModeCapabilitiesEntry 3 }


   --
   -- SMF RSSA Capabilities Table
   --

   smfRssaCapabilitiesTable OBJECT-TYPE
       SYNTAX      SEQUENCE OF SmfRssaCapabilitiesEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "The smfRssaCapabilitiesTable contains
            reference to the specific set of RSSAs
            currently supported on this device.
           "
       ::= { smfCapabilitiesGroup 2 }

   smfRssaCapabilitiesEntry OBJECT-TYPE
       SYNTAX      SmfRssaCapabilitiesEntry
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "Information about a particular RSSA
            algorithm."
       INDEX   { smfRssaCapabilitiesID }
       ::= { smfRssaCapabilitiesTable 1 }

   SmfRssaCapabilitiesEntry ::= SEQUENCE {
         smfRssaCapabilitiesID              SmfRssaID,
         smfRssaCapabilitiesName            SnmpAdminString,
         smfRssaCapabilitiesReference       SnmpAdminString
   }

   smfRssaCapabilitiesID     OBJECT-TYPE
       SYNTAX      SmfRssaID
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
           "The index for this entry.  This object identifies
            the particular RSSA algorithm in this MIB
            module.  Example RSSAs are found in the
            appendix of [SMF]."
       ::= { smfRssaCapabilitiesEntry 1 }

   smfRssaCapabilitiesName OBJECT-TYPE
       SYNTAX      SnmpAdminString
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
           "The textual name of this RSSA algorithm.
            Currently defined names are:
                Classical Flooding - cF,
                Source-based MultiPoint
                    Relay - sMPR,
                Essential Connecting Dominating
                    Set - eCDS,
                MultiPoint Relay Connected
                    Dominating Set - mprCDS.
           "
       ::= { smfRssaCapabilitiesEntry 2 }

   smfRssaCapabilitiesReference OBJECT-TYPE
       SYNTAX      SnmpAdminString
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
           "This object contains a published reference
            to the document that defines this algorithm.
           "
       ::= { smfRssaCapabilitiesEntry 3 }




   --
   -- smfConfigurationGroup
   --
   --    This group contains the SMF objects that configure specific
   --    options that determine the overall performance and operation
   --    of the multicast forwarding process for the router device
   --    and its interfaces.
   --

   smfConfigurationGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 2 }

   smfAdminStatus  OBJECT-TYPE
      SYNTAX      SmfStatus
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The configured status of the SMF process
          on this device.  Enabled(1) means that
          SMF is configured to run on this device.
          Disabled(2) mean that the SMF process
          is configured off.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 1 }

   -- Note: need to better define the algorithm to
   --   choose the smfRouterID.
   smfRouterIDAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The address type of the address used for
          SMF ID of this router as specified
          in the 'smfRouterID' next.

          This can be set by the management station, must
          the smfRouterID must be a routable address
          assigned to this router.  If the management
          station does not assign this value, then the
          router should choose the highest IP address
          assigned to this router.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 2 }

   smfRouterID  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The IP address used as the SMF router ID.
          this can be set by the management station.
          If not explicitly set, then the device
          should select a routable IP address
          assigned to this router for use as
          the 'smfRouterID'.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 3 }

   smfConfiguredOpMode  OBJECT-TYPE
      SYNTAX      INTEGER {
                          withNHDP(1),
                          cdsAwareRouting(2),
                          crossLayer(3),
                          other(4)
                          }
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
```

```
         "The SMF RSS node operational mode as defined
          in the TEXTUAL CONVENTION for `SmfOpModeID'
          and in [SMF]..

          The value withNHDP(1) indicates Independent
          Mode of operation.

          The value cdsAwareRouting(2) indicates
          CDS-aware Routing Mode of operation.

          The value crossLayer(3) indicates
          Cross-layer Mode of operation.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 4 }


   smfConfiguredRssa  OBJECT-TYPE
      SYNTAX      SmfRssaID
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The SMF RSS currently operational algorithm
          as defined in the TEXTUAL CONVENTION for
          `SmfRssaID' and in [SMF].

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 5 }

   smfRssaMember  OBJECT-TYPE
      SYNTAX      INTEGER {
                          potential(1),
                          always(2),
                          never(3)
                          }
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The RSSA downselects a set of forwarders for
          multicast forwarding.  Sometimes it is useful
          to force an agent to be included or excluded
          from the resulting RSS.  This object is a
          switch to allow for this behavior.

          The value potential(1) allows the selected
          RSSA to determine if this agent is included
          or excluded from the RSS.

          The value always(1) forces the selected
          RSSA include this agent in the RSS.

          The value never(3) forces the selected
          RSSA to exclude this agent from the RSS.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 6 }

   smfIpv4Dpd  OBJECT-TYPE
      SYNTAX      INTEGER {
                          identificationBased(1),
                          hashBased(2)
                          }
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The current method for IPv4 duplicate packet
          detection.

          The value identificationBased(1)
          indicates that the duplicate packet
          detection relies upon header information
          in the multicast packets to identify
          previously received packets.

          The value 'hashBased(2) indicates that the
          routers duplicate packet detection is based
          upon comparing a hash over the packet fields.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 7 }

   smfIpv6Dpd  OBJECT-TYPE
      SYNTAX      INTEGER {
                          identificationBased(1),
                          hashBased(2)
                          }
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The current method for IPv6 duplicate packet
          detection.

          The values indicate the type of method used
          for duplicate packet detection as described
          the previous description for the object
          `smfIpv4Dpd'.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 8 }

   smfMaxPktLifetime  OBJECT-TYPE
      SYNTAX      Integer32 (0..65535)
      UNITS       "Seconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The estimate of the network packet
          traversal time.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
      DEFVAL { 60 }
   ::= { smfConfigurationGroup 9 }

   smfDpdMaxMemorySize  OBJECT-TYPE
      SYNTAX      Integer32 (0..65535)
      UNITS       "Kilo-Bytes"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The locally reserved memory for storage
          of cached DPD records for both IPv4 and
          IPv6 methods.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
      DEFVAL { 1024 }
   ::= { smfConfigurationGroup 10 }

   smfDpdEntryMaxLifetime  OBJECT-TYPE
      SYNTAX      Integer32 (0..65525)
      UNITS       "Seconds"
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "The maximum lifetime of a cached DPD
          record in the local device storage.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
      DEFVAL { 600 }
   ::= { smfConfigurationGroup 11 }


   --
   -- Configuration of messages to be included in
   -- NHDP message exchanges in support of SMF
   -- operations.
   --

   -- Note: need to clarify whether this is an option
   --  or is required when the smfOpMode is set
   --  to 'independent'.
   smfNhdpRssaMesgTLVIncluded  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "Indicates whether the associated NHDP messages
          include the RSSA Message TLV, or not.  This
          is an optional SMF operational setting.
          The value true(1) indicates that this TLV is
          included; the value false(2) indicates that it
          is not included.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 12 }
   smfNhdpRssaAddrBlockTLVIncluded  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-write
      STATUS      current
      DESCRIPTION
         "Indicates whether the associated NHDP messages
          include the RSSA Address Block TLV, or not.
          This is an optional SMF operational setting.
          The value true(1) indicates that this TLV is
          included; the value false(2) indicates that it
          is not included.

          This object is persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 13 }



   --
   -- Table identifying configured multicast addresses to be forwarded.
   --

   smfConfiguredAddrForwardingTable  OBJECT-TYPE
      SYNTAX     SEQUENCE OF SmfConfiguredAddrForwardingEntry
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The (conceptual) table containing information on multicast
          addresses which are to be forwarded by the SMF process.

          Entries in this table are configured.  As well, addresses
          to be forwarded by the SMF device can be dynamically
          discovered by other means.  The corresponding state
          table, smfDiscoveredAddrForwardingTable, contains
          these additional, dynamically discovered address for
          forwarding.

          Each row is associated with a range of multicast
          addresses, and ranges for different rows must be disjoint.

          The objects in this table are persistent and when written
          the entity SHOULD save the change to
          non-volatile storage.
         "
   ::= { smfConfigurationGroup 15 }
   smfConfiguredAddrForwardingEntry OBJECT-TYPE
      SYNTAX     SmfConfiguredAddrForwardingEntry
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "An entry (conceptual row) containing the information on a
          particular multicast scope."
      INDEX { smfConfiguredAddrForwardingAddrType,
              smfConfiguredAddrForwardingFirstAddr }
      ::= { smfConfiguredAddrForwardingTable 1 }

   SmfConfiguredAddrForwardingEntry ::= SEQUENCE {
      smfConfiguredAddrForwardingAddrType      InetAddressType,
      smfConfiguredAddrForwardingFirstAddr     InetAddress,
      smfConfiguredAddrForwardingLastAddr      InetAddress,
      smfConfiguredAddrForwardingStatus        RowStatus
   }

   smfConfiguredAddrForwardingAddrType OBJECT-TYPE
      SYNTAX     InetAddressType
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The type of the addresses in the multicast forwarding
          range.  Legal values correspond to the subset of
          address families for which multicast address allocation
          is supported."
   ::= { smfConfiguredAddrForwardingEntry 1 }

   smfConfiguredAddrForwardingFirstAddr OBJECT-TYPE
      SYNTAX     InetAddress (SIZE(0..20))
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The first address in the multicast scope range.  The type
          of this address is determined by the value of the
          smfConfiguredAddrForwardingAddrType object."
   ::= { smfConfiguredAddrForwardingEntry 2 }

   smfConfiguredAddrForwardingLastAddr OBJECT-TYPE
      SYNTAX     InetAddress (SIZE(0..20))
      MAX-ACCESS read-create
      STATUS     current
      DESCRIPTION
         "The last address in the multicast scope range.
          The type of this address is determined by the
          value of the smfConfiguredAddrForwardingAddrType
          object."
   ::= { smfConfiguredAddrForwardingEntry 3 }

   smfConfiguredAddrForwardingStatus OBJECT-TYPE
      SYNTAX     RowStatus
      MAX-ACCESS read-create
      STATUS     current
      DESCRIPTION
         "The status of this row, by which new entries may be
          created, or old entries deleted from this table.  If write
          access is supported, the other writable objects in this
          table may be modified even while the status is 'active'."
   ::= { smfConfiguredAddrForwardingEntry 4 }



   --
   -- SMF Interfaces Configuration Table
   --

   smfInterfaceTable  OBJECT-TYPE
      SYNTAX      SEQUENCE OF SmfInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The SMF Interface Table describes the SMF
          interfaces that are participating in the
          SMF packet forwarding process. The ifIndex is
          from the interfaces group defined in the
          Interfaces Group MIB.

          The objects in this table are persistent
          and when written the entity SHOULD save
          the change to non-volatile storage.
          "
      REFERENCE
         "RFC 2863 - The Interfaces Group MIB, McCloghrie,
          K., and F. Kastenholtz, June 2000."
   ::= { smfConfigurationGroup 16 }

   smfInterfaceEntry OBJECT-TYPE
      SYNTAX      SmfInterfaceEntry
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The SMF interface entry describes one SMF
          interface as indexed by its ifIndex."
      INDEX { smfIfIndex }
   ::= { smfInterfaceTable 1 }
   SmfInterfaceEntry ::=
      SEQUENCE {
         smfIfIndex        InterfaceIndexOrZero,
         smfIfAdminStatus  SmfStatus,
         smfIfRowStatus    RowStatus
         }

   smfIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The ifIndex for this SMF interface."
      ::= { smfInterfaceEntry 1 }

   smfIfAdminStatus OBJECT-TYPE
      SYNTAX      SmfStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
          "The SMF interface's administrative status.
          The value 'enabled' denotes that the interface
          is running the SMF forwarding process.
          The value 'disabled' denotes that the interface is
          external to the SMF forwarding process.
          "
      ::= { smfInterfaceEntry 2 }

   smfIfRowStatus  OBJECT-TYPE
      SYNTAX      RowStatus
      MAX-ACCESS  read-create
      STATUS      current
      DESCRIPTION
         "This object permits management of the table
          by facilitating actions such as row creation,
          construction, and destruction. The value of
          this object has no effect on whether other
          objects in this conceptual row can be
          modified."
   ::= { smfInterfaceEntry 3 }



   --
   -- smfStateGroup
   --
   --    Contains information describing the current state of the SMF
   --    process such as the current inclusion in the RS or not.
   --

   smfStateGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 3 }

   smfNodeRsStatusIncluded  OBJECT-TYPE
      SYNTAX      TruthValue
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The current status of the SMF node in the context of
          the MANETs relay set. A value of true(1) indicates
          that the node is currently part of the MANET Relay
          Set. A value of false(2) indicates that the node
          is currently not part of the MANET Relay Set."
   ::= { smfStateGroup 1 }

   smfDpdMemoryOverflow  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "The number of times that the memory for caching
          records for DPD overran and records had to be flushed.
          The number of records to be flushed upon a buffer
          overflow is an implementation specific decision.
         "
   ::= { smfStateGroup 2 }



   --
   -- Dynamically Discovered Multicast Addr Table
   --

   smfDiscoveredAddrForwardingTable  OBJECT-TYPE
      SYNTAX     SEQUENCE OF SmfDiscoveredAddrForwardingEntry
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The (conceptual) table containing information on multicast
          addresses which are to be forwarded by the SMF process.

          Entries in this table are configured.  As well, addresses
          to be forwarded by the SMF device can be dynamically
          discovered by other means.  The corresponding state
          table, smfDiscoveredAddrForwardingTable contains
          these additional, dynamically discovered address for
          forwarding.
          Each row is associated with a range of
          multicast addresses, and ranges for different rows
          must be disjoint.
         "
   ::= { smfStateGroup 3 }

   smfDiscoveredAddrForwardingEntry OBJECT-TYPE
      SYNTAX     SmfDiscoveredAddrForwardingEntry
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "An entry (conceptual row) containing the information on a
          particular multicast scope."
      INDEX { smfDiscoveredAddrForwardingAddrType,
              smfDiscoveredAddrForwardingFirstAddr }
      ::= { smfDiscoveredAddrForwardingTable 1 }

   SmfDiscoveredAddrForwardingEntry ::= SEQUENCE {
      smfDiscoveredAddrForwardingAddrType   InetAddressType,
      smfDiscoveredAddrForwardingFirstAddr  InetAddress,
      smfDiscoveredAddrForwardingLastAddr   InetAddress,
      smfDiscoveredAddrForwardingStatus     RowStatus
   }

   smfDiscoveredAddrForwardingAddrType OBJECT-TYPE
      SYNTAX     InetAddressType
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The type of the addresses in the multicast forwarding
          range.  Legal values correspond to the subset of
          address families for which multicast address allocation
          is supported."
   ::= { smfDiscoveredAddrForwardingEntry 1 }

   smfDiscoveredAddrForwardingFirstAddr OBJECT-TYPE
      SYNTAX     InetAddress (SIZE(0..20))
      MAX-ACCESS not-accessible
      STATUS     current
      DESCRIPTION
         "The first address in the multicast scope range.  The type
          of this address is determined by the value of the
          smfConfiguredAddrForwardingAddrType object."
   ::= { smfDiscoveredAddrForwardingEntry 2 }

   smfDiscoveredAddrForwardingLastAddr OBJECT-TYPE
      SYNTAX     InetAddress (SIZE(0..20))
      MAX-ACCESS read-create
      STATUS     current
      DESCRIPTION
         "The last address in the multicast scope range.
          The type of this address is determined by the
          value of the smfConfiguredAddrForwardingAddrType
          object."
   ::= { smfDiscoveredAddrForwardingEntry 3 }

   smfDiscoveredAddrForwardingStatus OBJECT-TYPE
      SYNTAX     RowStatus
      MAX-ACCESS read-create
      STATUS     current
      DESCRIPTION
         "The status of this row, by which new entries may be
          created, or old entries deleted from this table.  If write
          access is supported, the other writable objects in this
          table may be modified even while the status is 'active'."
   ::= { smfDiscoveredAddrForwardingEntry 4 }



   --
   -- SMF Neighbor Table
   --

   smfNeighborTable  OBJECT-TYPE
      SYNTAX       SEQUENCE OF SmfNeighborEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF NeighborTable describes the
          current neighbor nodes, their address
          and SMF RSSA and the interface on which
          they can be reached."
      REFERENCE
         "Simplified Multicast Forwarding for MANET
          (SMF), Macker, J., July 2009.
          Section 7: SMF Neighborhood Discovery
          Requirements."
   ::= { smfStateGroup 4 }

   smfNeighborEntry  OBJECT-TYPE
      SYNTAX       SmfNeighborEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF Neighbor Table contains the
          set of one-hop neighbors, the interface
          they are reachable on and the SMF RSSA
          they are currently running."
      INDEX { smfNeighborIpAddrType,
              smfNeighborIpAddr,
              smfNeighborPrefixLen }
   ::= { smfNeighborTable 1 }

   SmfNeighborEntry ::=
      SEQUENCE {
         smfNeighborIpAddrType        InetAddressType,
         smfNeighborIpAddr            InetAddress,
         smfNeighborPrefixLen         InetAddressPrefixLength,
         smfNeighborRSSA              SmfRssaID,
         smfNeighborNextHopInterface  InterfaceIndexOrZero
         }

   smfNeighborIpAddrType  OBJECT-TYPE
      SYNTAX      InetAddressType
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The neighbor IP address type."
   ::= { smfNeighborEntry 1 }

   smfNeighborIpAddr  OBJECT-TYPE
      SYNTAX      InetAddress
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The neighbor Inet IPv4 or IPv6 address."
   ::= { smfNeighborEntry 2 }

   smfNeighborPrefixLen  OBJECT-TYPE
      SYNTAX      InetAddressPrefixLength
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The prefix length. This is a decimal value that
          indicates the number of contiguous, higher-order
          bits of the address that make up the network
          portion of the address."
   ::= { smfNeighborEntry 3 }

   smfNeighborRSSA  OBJECT-TYPE
      SYNTAX       SmfRssaID
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
```

```
         "The current RSSA running on the neighbor.
          The list is identical to that described
          above for the smfRssa object."
   ::= { smfNeighborEntry 4 }

   smfNeighborNextHopInterface OBJECT-TYPE
      SYNTAX       InterfaceIndexOrZero
      MAX-ACCESS   read-only
      STATUS       current
      DESCRIPTION
         "The interface ifIndex over which the
          neighbor is reachable in one-hop."
   ::= { smfNeighborEntry 5 }




   --
   -- SMF Performance Group
   --
   --    Contains objects which help to characterize the
   --    performance of the SMF RSSA process, such as statistics
   --    counters. There are two types of SMF RSSA statistics:
   --    global counters and per interface counters.
   --

   smfPerformanceGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 4 }

   smfGlobalPerfGroup  OBJECT IDENTIFIER ::= { smfPerformanceGroup 1 }

   --
   -- IPv4 packet counters
   --

   smfIpv4MultiPktsRecvTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of
          multicast IPv4 packets received by the
          device."
   ::= { smfGlobalPerfGroup 1 }

   smfIpv4MultiPktsForwardedTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of
          multicast IPv4 packets forwarded by the
          device."
   ::= { smfGlobalPerfGroup 2 }

   smfIpv4DuplMultiPktsDetectedTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of duplicate
          multicast IPv4 packets detected by the
          device."
   ::= { smfGlobalPerfGroup 3 }

   smfIpv4DroppedMultiPktsTTLExceededTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of dropped
          multicast IPv4 packets by the
          device due to TTL exceeded."
   ::= { smfGlobalPerfGroup 4 }

   smfIpv4TTLLargerThanPreviousTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv4 packets
          recieved which have a TTL larger than that
          of a previously received identical packet.
         "
   ::= { smfGlobalPerfGroup 5 }

   --
   -- IPv6 packet counters
   --

   smfIpv6MultiPktsRecvTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of
          multicast IPv6 packets received by the
          device."
   ::= { smfGlobalPerfGroup 6 }

   smfIpv6MultiPktsForwardedTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of
          multicast IPv6 packets forwarded by the
          device."
   ::= { smfGlobalPerfGroup 7 }

   smfIpv6DuplMultiPktsDetectedTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of duplicate
          multicast IPv6 packets detected by the
          device."
   ::= { smfGlobalPerfGroup 8 }

   smfIpv6DroppedMultiPktsTTLExceededTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of dropped
          multicast IPv6 packets by the
          device due to TTL exceeded."
   ::= { smfGlobalPerfGroup 9 }

   smfIpv6TTLLargerThanPreviousTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which have a TTL larger than that
          of a previously recived identical packet.
         "
   ::= { smfGlobalPerfGroup 10 }

   smfIpv6HAVAssistsReqdTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which required the HAV assist for DPD.
         "
   ::= { smfGlobalPerfGroup 11 }

   smfIpv6DpdHeaderInsertionsTotal  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which the device inserted the
          DPD header option.
         "
   ::= { smfGlobalPerfGroup 12 }


   --
   -- Per SMF Interface Performance Table
   --

   smfInterfacePerfGroup OBJECT IDENTIFIER ::= { smfPerformanceGroup 2 }

   smfIpv4InterfacePerfTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF SmfIpv4InterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF Interface Performance Table
          describes the SMF statistics per
          interface."
   ::= { smfInterfacePerfGroup 1 }

   smfIpv4InterfacePerfEntry OBJECT-TYPE
      SYNTAX       SmfIpv4InterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF Interface Performance entry
          describes the statistics for a particular
          node interface."
      INDEX { smfIpv4IfPerfIfIndex }
   ::= { smfIpv4InterfacePerfTable 1 }

   SmfIpv4InterfacePerfEntry ::=
      SEQUENCE {
         smfIpv4IfPerfIfIndex                    InterfaceIndexOrZero,
         smfIpv4MultiPktsRecvPerIf               Counter32,
         smfIpv4MultiPktsForwardedPerIf          Counter32,
         smfIpv4DuplMultiPktsDetectedPerIf       Counter32,
         smfIpv4DroppedMultiPktsTTLExceededPerIf Counter32,
         smfIpv4TTLLargerThanPreviousPerIf       Counter32
         }

   smfIpv4IfPerfIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The ifIndex for this node interface
          that is collecting this set of
          performance management statistics."
   ::= { smfIpv4InterfacePerfEntry 1 }

   smfIpv4MultiPktsRecvPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          multicast IP packets received by the
          device on this interface."
   ::= { smfIpv4InterfacePerfEntry 2 }

   smfIpv4MultiPktsForwardedPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          multicast IP packets forwarded by the
          device on this interface."
   ::= { smfIpv4InterfacePerfEntry 3 }

   smfIpv4DuplMultiPktsDetectedPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of duplicate
          multicast IP packets detected by the
          device on this interface."
   ::= { smfIpv4InterfacePerfEntry 4 }

   smfIpv4DroppedMultiPktsTTLExceededPerIf  OBJECT-TYPE
```

```
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of dropped
          multicast IPv4 packets by the
          device due to TTL exceeded."
   ::= { smfIpv4InterfacePerfEntry 5 }

   smfIpv4TTLLargerThanPreviousPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv4 packets
          recieved which have a TTL larger than that
          of a previously recived identical packet.
         "
   ::= { smfIpv4InterfacePerfEntry 6 }


   smfIpv6InterfacePerfTable OBJECT-TYPE
      SYNTAX       SEQUENCE OF SmfIpv6InterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF Interface Performance Table
          describes the SMF statistics per
          interface."
   ::= { smfInterfacePerfGroup 2 }

   smfIpv6InterfacePerfEntry OBJECT-TYPE
      SYNTAX       SmfIpv6InterfacePerfEntry
      MAX-ACCESS   not-accessible
      STATUS       current
      DESCRIPTION
         "The SMF Interface Performance entry
          describes the statistics for a particular
          node interface."
      INDEX { smfIpv6IfPerfIfIndex }
   ::= { smfIpv6InterfacePerfTable 1 }

   SmfIpv6InterfacePerfEntry ::=
      SEQUENCE {
         smfIpv6IfPerfIfIndex                   InterfaceIndexOrZero,
         smfIpv6MultiPktsRecvPerIf              Counter32,
         smfIpv6MultiPktsForwardedPerIf         Counter32,
         smfIpv6DuplMultiPktsDetectedPerIf      Counter32,
         smfIpv6DroppedMultiPktsTTLExceededPerIf Counter32,
         smfIpv6TTLLargerThanPreviousPerIf       Counter32,
         smfIpv6HAVAssistsReqdPerIf              Counter32,
         smfIpv6DpdHeaderInsertionsPerIf         Counter32
         }

   smfIpv6IfPerfIfIndex  OBJECT-TYPE
      SYNTAX      InterfaceIndexOrZero
      MAX-ACCESS  not-accessible
      STATUS      current
      DESCRIPTION
         "The ifIndex for this node interface
          that is collecting this set of
          performance management statistics.

          For packets generated locally at
          this node, performance counters
          are assigned to the loopback
          interface.
         "
   ::= { smfIpv6InterfacePerfEntry 1 }

   smfIpv6MultiPktsRecvPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          multicast IP packets received by the
          device on this interface."
   ::= { smfIpv6InterfacePerfEntry 2 }

   smfIpv6MultiPktsForwardedPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of
          multicast IP packets forwarded by the
          device on this interface."
   ::= { smfIpv6InterfacePerfEntry 3 }

   smfIpv6DuplMultiPktsDetectedPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of duplicate
```

```
          multicast IP packets detected by the
          device on this interface."
   ::= { smfIpv6InterfacePerfEntry 4 }

   smfIpv6DroppedMultiPktsTTLExceededPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the number of dropped
          multicast IP packets by the
          device on this interface due to TTL
          exceeded."
   ::= { smfIpv6InterfacePerfEntry 5 }

   smfIpv6TTLLargerThanPreviousPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which have a TTL larger than that
          of a previously recived identical packet.
         "
   ::= { smfIpv6InterfacePerfEntry 6 }

   smfIpv6HAVAssistsReqdPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which required the HAV assist for DPD.
         "
   ::= { smfIpv6InterfacePerfEntry 7 }

   smfIpv6DpdHeaderInsertionsPerIf  OBJECT-TYPE
      SYNTAX      Counter32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
         "A counter of the total number of IPv6 packets
          recieved which the device inserted the
          DPD header option.
         "
   ::= { smfIpv6InterfacePerfEntry 8 }
   --
   -- Notifications
   --

smfMIBNotifControl OBJECT IDENTIFIER ::= { smfMIBNotifications 1 }
smfMIBNotifObjects OBJECT IDENTIFIER ::= { smfMIBNotifications 2 }
smfMIBNotifStates  OBJECT IDENTIFIER ::= { smfMIBNotifications 3 }


   -- smfMIBNotifControl
   smfSetNotification OBJECT-TYPE
          SYNTAX       OCTET STRING (SIZE(4))
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A 4-octet string serving as a bit map for
             the notification events defined by the SMF MIB
             notifications. This object is used to enable
             and disable specific SMF MIB notifications where
             a 1 in the bit field represents enabled. The
             right-most bit (least significant) represents
             notification 0.

             This object is persistent and when written
             the entity SHOULD save the change to
             non-volatile storage.
             "
           ::= { smfMIBNotifControl 1 }

   smfDpdMemoryOverflowThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              `smfDpdmemoryOverflowEvents' object.
              If the number of occurences exceeds
              this threshold within the previous
              number of seconds
              'smfDpdMemoryOverflowWindow',
              then the `smfDpdMemoryOverflowEvent'
              notification is sent.
             "
           ::= { smfMIBNotifControl 2 }

   smfDpdMemoryOverflowWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window value for the
              ‘smfDpdmemoryOverflowEvents’ object.
              If the number of occurences exceeds
              the ‘smfDpdMemoryOverflowThreshold’
              within the previous number of seconds
              ’smfDpdMemoryOverflowWindow’,
              then the ‘smfDpdMemoryOverflowEvent’
              notification is sent.
             "
           ::= { smfMIBNotifControl 3 }

   smfIpv4DuplMultiPktsDetectedTotalThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              ‘smfIpv4DuplMultiPktsDetectedTotal’
              object.  If the number of occurences
              exceeds this threshold within the
              previous number of seconds
              ‘smfIpv4DuplMultiPktsDetectedTotalWindow’,
              then the
              ‘smfIpv4DuplMultiPktsDetectedTotalEvent’
              notification is sent.
             "
           ::= { smfMIBNotifControl 4 }

   smfIpv4DuplMultiPktsDetectedTotalWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window value for the
              ‘smfIpv4DuplMultiPktsDetectedTotalEvents’
              object.  If the number of occurences
              exceeds the
              ‘smfIpv4DuplMultiPktsDetectedTotalThreshold’
              within the previous number of seconds
              ’smfIpv4DuplMultiPktsDetectedTotalWindow’,
              then the
              ‘smfIpv4DuplMultiPktsDetectedTotalEvent’
              notification is sent.
             "
           ::= { smfMIBNotifControl 5 }
   smfIpv6DuplMultiPktsDetectedTotalThreshold OBJECT-TYPE
          SYNTAX       Integer32 (0..255)
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A threshold value for the
              `smfIpv6DuplMultiPktsDetectedTotal'
              object.  If the number of occurences
              exceeds this threshold within the
              previous number of seconds
              `smfIpv6DuplMultiPktsDetectedTotalWindow',
              then the
              `smfIpv6DuplMultiPktsDetectedTotalEvent'
              notification is sent.
             "
           ::= { smfMIBNotifControl 6 }

   smfIpv6DuplMultiPktsDetectedTotalWindow OBJECT-TYPE
          SYNTAX       TimeTicks
          MAX-ACCESS   read-write
          STATUS       current
          DESCRIPTION
             "A time window value for the
              `smfIpv6DuplMultiPktsDetectedTotalEvents'
              object.  If the number of occurences
              exceeds the
              `smfIpv6DuplMultiPktsDetectedTotalThreshold'
              within the previous number of seconds
              'smfIpv6DuplMultiPktsDetectedTotalWindow',
              then the
              `smfIpv6DuplMultiPktsDetectedTotalEvent'
              notification is sent.
             "
           ::= { smfMIBNotifControl 7 }



   -- smfMIBNotifObjects

   smfAdminStatusChange NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,         -- The originator of
                                         --     the notification.
                    smfAdminStatus       -- The new status of the
                                         --     SMF process.
                  }
          STATUS       current
          DESCRIPTION
             "smfAdminStatusChange is a notification sent when a
              the 'smfAdminStatus' object changes.
             "
          ::= { smfMIBNotifObjects 1 }

   smfConfiguredOpModeChange NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,         -- The originator of
                                         --     the notification.
                    smfConfiguredOpMode  -- The new Operations
                                         --     Mode of the SMF
                                         --     process.
                  }
          STATUS       current
          DESCRIPTION
             "smfConfiguredOpModeChange is a notification
              sent when a the 'smfConfiguredOpMode' object
              changes.
             "
          ::= { smfMIBNotifObjects 2 }

   smfConfiguredRssaChange NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,         -- The originator of
                                         --     the notification.
                    smfConfiguredRssa -- The new RSSA for
                                         --     the SMF process.
                  }
          STATUS       current
          DESCRIPTION
             "smfAdminStatusChange is a notification sent when a
              the 'smfConfiguredRssa' object changes.
             "
          ::= { smfMIBNotifObjects 3 }

   smfIfAdminStatusChange NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,          -- The originator of
                                         --     the notification.
                    smfIfIndex,          -- The interface whose
                                         --     status has changed.
                    smfIfAdminStatus   -- The new status of the
                                         --     SMF interface.
                  }
          STATUS       current
          DESCRIPTION
             "smfIfAdminStatusChange is a notification sent when a
              the 'smfIfAdminStatus' object changes.
             "
          ::= { smfMIBNotifObjects 4 }

    smfDpdMemoryOverflowEvent NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,         -- The originator of
                                         --     the notification.
                    smfDpdMemoryOverflow -- The counter of
                                         --     the overflows.
             }
          STATUS       current
          DESCRIPTION
             "smfDpdMemoryOverflowEvents is sent when the
              number of memory overflow events exceeds the
              the 'smfDpdMemoryOverflowThreshold' within the
              previous number of seconds defined by the
              'smfDpdMemoryOverflowWindow'.
             "
          ::= { smfMIBNotifObjects 5 }

   smfIpv4DuplMultiPktsDetectedTotalEvents NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,         -- The originator of
                                         --     the notification.
                    smfIpv4DuplMultiPktsDetectedTotal -- The
                                         --     counter of detected
                                         --     duplicates.
             }
          STATUS       current
          DESCRIPTION
             "smfIpv4DuplMultiPktsDetectedTotal is a
              notification sent when the number of
              IPv4 duplicate packets detected exceeds the
              'smfIpv4DuplMultiPktsDetectedTotalThreshold'
              during the previous number of seconds
              'smfIpv4DuplPktsDetectedTotalWindow'.
             "
          ::= { smfMIBNotifObjects 6 }

   smfIpv6DuplMultiPktsDetectedTotalEvents NOTIFICATION-TYPE
          OBJECTS { smfRouterIDAddrType, -- The originator of
                                         --     the notification.
                    smfRouterID,      -- The originator of
                                      --     the notification.
                    smfIpv6DuplMultiPktsDetectedTotal -- The
                                      --     counter of detected
                                      --     duplicates.
             }
          STATUS       current
          DESCRIPTION
             "smfIpv6DuplMultiPktsDetectedTotal is a
              notification sent when the number of
              IPv6 duplicate packets detected exceeds the
              'smfIpv6DuplMultiPktsDetectedTotalThreshold'
              during the previous number of seconds
              'smfIpv6DuplPktsDetectedTotalWindow'.
             "
          ::= { smfMIBNotifObjects 7 }



    -- smfMIBNotifStates
    --   is empty.




   --
   -- Compliance Statements
   --

   smfCompliances  OBJECT IDENTIFIER ::= { smfMIBConformance 1 }
   smfMIBGroups    OBJECT IDENTIFIER ::= { smfMIBConformance 2 }

   smfBasicCompliance  MODULE-COMPLIANCE
      STATUS current
      DESCRIPTION "The basic implementation requirements for
                   managed network entities that implement
                   the SMF RSSA process."
      MODULE  -- this module
      MANDATORY-GROUPS { smfCapabObjectsGroup,
                         smfConfigObjectsGroup }
   ::= { smfCompliances 1 }

   smfFullCompliance MODULE-COMPLIANCE
      STATUS current
      DESCRIPTION "The full implementation requirements for
                   managed network entities that implement
                   the SMF RSSA process."
      MODULE  -- this module
      MANDATORY-GROUPS { smfCapabObjectsGroup,
                         smfConfigObjectsGroup,
                         smfStateObjectsGroup,
                         smfPerfObjectsGroup,
                         smfNotifObjectsGroup,
                         smfNotificationsGroup
                       }
   ::= { smfCompliances 2 }

   --
   -- Units of Conformance
   --

   smfCapabObjectsGroup OBJECT-GROUP
      OBJECTS {
              smfOpModeCapabilitiesName,
              smfOpModeCapabilitiesReference,

              smfRssaCapabilitiesName,
              smfRssaCapabilitiesReference
      }
      STATUS  current
      DESCRIPTION
         "Set of SMF configuration objects implemented
          in this module."
   ::= { smfMIBGroups 1 }

   smfConfigObjectsGroup OBJECT-GROUP
      OBJECTS {
              smfAdminStatus,
              smfRouterIDAddrType,
              smfRouterID,
              smfIfIndex,
              smfConfiguredOpMode,
              smfConfiguredRssa,
              smfRssaMember,
              smfIpv4Dpd,
              smfIpv6Dpd,
              smfMaxPktLifetime,
              smfDpdMaxMemorySize,
              smfDpdEntryMaxLifetime,
              smfNhdpRssaMesgTLVIncluded,
              smfNhdpRssaAddrBlockTLVIncluded,

              smfConfiguredAddrForwardingLastAddr,
              smfConfiguredAddrForwardingStatus,
                smfIfAdminStatus,
                smfIfRowStatus
          }
          STATUS  current
          DESCRIPTION
             "Set of SMF configuration objects implemented
              in this module."
       ::= { smfMIBGroups 2 }

       smfStateObjectsGroup  OBJECT-GROUP
          OBJECTS {
                smfNodeRsStatusIncluded,
                smfDpdMemoryOverflow,

                smfDiscoveredAddrForwardingLastAddr,
                smfDiscoveredAddrForwardingStatus,

                smfNeighborRSSA,
                smfNeighborNextHopInterface
          }
          STATUS  current
          DESCRIPTION
             "Set of SMF state objects implemented
              in this module."
       ::= { smfMIBGroups 3 }

       smfPerfObjectsGroup  OBJECT-GROUP
          OBJECTS {
                smfIpv4MultiPktsRecvTotal,
                smfIpv4MultiPktsForwardedTotal,
                smfIpv4DuplMultiPktsDetectedTotal,
                smfIpv4DroppedMultiPktsTTLExceededTotal,
                smfIpv4TTLLargerThanPreviousTotal,

                smfIpv6MultiPktsRecvTotal,
                smfIpv6MultiPktsForwardedTotal,
                smfIpv6DuplMultiPktsDetectedTotal,
                smfIpv6DroppedMultiPktsTTLExceededTotal,
                smfIpv6TTLLargerThanPreviousTotal,
                smfIpv6HAVAssistsReqdTotal,
                smfIpv6DpdHeaderInsertionsTotal,

                smfIpv4MultiPktsRecvPerIf,
                smfIpv4MultiPktsForwardedPerIf,
                smfIpv4DuplMultiPktsDetectedPerIf,
                smfIpv4DroppedMultiPktsTTLExceededPerIf,
                smfIpv4TTLLargerThanPreviousPerIf,
                smfIpv6MultiPktsRecvPerIf,
                smfIpv6MultiPktsForwardedPerIf,
                smfIpv6DuplMultiPktsDetectedPerIf,
                smfIpv6DroppedMultiPktsTTLExceededPerIf,
                smfIpv6TTLLargerThanPreviousPerIf,
                smfIpv6HAVAssistsReqdPerIf,
                smfIpv6DpdHeaderInsertionsPerIf
      }
      STATUS  current
      DESCRIPTION
         "Set of SMF performance objects implemented
          in this module by total and per interface."
   ::= { smfMIBGroups 4 }

   smfNotifObjectsGroup  OBJECT-GROUP
      OBJECTS {
                smfSetNotification,
                smfDpdMemoryOverflowThreshold,
                smfDpdMemoryOverflowWindow,
                smfIpv4DuplMultiPktsDetectedTotalThreshold,
                smfIpv4DuplMultiPktsDetectedTotalWindow,
                smfIpv6DuplMultiPktsDetectedTotalThreshold,
                smfIpv6DuplMultiPktsDetectedTotalWindow
      }
      STATUS  current
      DESCRIPTION
         "Set of SMF notification control
          objects implemented in this module."
   ::= { smfMIBGroups 5 }

   smfNotificationsGroup  NOTIFICATION-GROUP
      NOTIFICATIONS {
                smfAdminStatusChange,
                smfConfiguredOpModeChange,
                smfConfiguredRssaChange,
                smfIfAdminStatusChange,
                smfDpdMemoryOverflowEvent,
                smfIpv4DuplMultiPktsDetectedTotalEvents,
                smfIpv6DuplMultiPktsDetectedTotalEvents
      }
      STATUS  current
      DESCRIPTION
         "Set of SMF notifications implemented
          in this module."
   ::= { smfMIBGroups 6 }
```

END

8. Security Considerations

[TODO] Each specification that defines one or more MIB modules MUST contain a section that discusses security considerations relevant to those modules. This section MUST be patterned after the latest approved template (available at http://www.ops.ietf.org/mib-security.html). Remember that the objective is not to blindly copy text from the template, but rather to think and evaluate the risks/vulnerabilities and then state/document the result of this evaluation.

[TODO] if you have any read-write and/or read-create objects, please include the following boilerplate paragraph.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- [TODO] writable MIB objects that could be especially disruptive if abused MUST be explicitly listed by name and the associated security risks MUST be spelled out; RFC 2669 has a very good example.

- [TODO] list the writable tables and objects and state why they are sensitive.

[TODO] else if there are no read-write objects in your MIB module, use the following boilerplate paragraph.

There are no management objects defined in this MIB module that have a MAX-ACCESS clause of read-write and/or read-create. So, if this MIB module is implemented correctly, then there is no risk that an intruder can alter or create any management objects of this MIB module via direct SNMP SET operations.

[TODO] if you have any sensitive readable objects, please include the following boilerplate paragraph.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- [TODO] you must explicitly list by name any readable objects that are sensitive or vulnerable and the associated security risks MUST be spelled out (for instance, if they might reveal customer information or violate personal privacy laws such as those of the European Union if exposed to unauthorized parties)

- [TODO] list the tables and objects and state why they are sensitive.

[TODO] discuss what security the protocol used to carry the information should have. The following three boilerplate paragraphs should not be changed without very good reason. Changes will almost certainly require justification during IESG review.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

9. IANA Considerations

[TODO] In order to comply with IESG policy as set forth in http://www.ietf.org/ID-Checklist.html, every Internet-Draft that is submitted to the IESG for publication MUST contain an IANA Considerations section. The requirements for this section vary depending what actions are required of the IANA. see RFC4181 section 3.5 for more information on writing an IANA clause for a MIB module document.

[TODO] select an option and provide the necessary details.

Option #1:

> The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:
>
> | Descriptor | OBJECT IDENTIFIER value |
> |---|---|
> | sampleMIB | { mib-2 XXX } |

Option #2:

Editor's Note (to be removed prior to publication): the IANA is requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

Note well: prior to official assignment by the IANA, a draft document MUST use placeholders (such as "XXX" above) rather than actual numbers. See RFC4181 Section 4.5 for an example of how this is done in a draft MIB module.

Option #3:

This memo includes no request to IANA.

10. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

11. Acknowledgements

12. References

12.1. Normative References

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network

Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

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

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[I-D.ietf-manet-smf] Macker, J. and S. Team, "Simplified Multicast Forwarding", draft-ietf-manet-smf-10 (work in progress), March 2010.

12.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

Appendix A. Change Log

This section tracks the revision history in the development of this SMF-MIB. It will be removed from the final version of this document.

These changes were made from draft-ietf-manet-smf-mib-01 to draft-ietf-manet-smf-mib-02.

1. Added the NotificationGroup to the MIB and updated the ConformanceGroup.

2. Added the definition of an smfRouterID to the MIB. This is later used in the Notifications to indicate the origin of the event to the management station.

3. Removed the Router Priority object as this was used only in the eCDS algorithm and hence should be contained within the future eCDS-MIB.

4. Cleaned up the TEXTUAL CONVENTION for the ‘SmfOpMode’.

5. Filled in some of the missing text in various object descriptions.

These changes were made from draft-ietf-manet-smf-mib-00 to draft-ietf-manet-dsmf-mib-01.

1. Editorial changes to the textual material. These included the addition of the paragraphs on TEXTUAL-CONVENTIONS defined and imported into this MIB and relationships to other MIBs.

2. Identified those objects in the SMF-MIB requiring non-volatile storage.

3. Changed the name of the TEXTUAL-CONVENTION ’Status’, defined within this MIB to ’SmfStatus’.

Appendix B. Open Issues

This section contains the set of open issues related to the development and design of the SMF-MIB. This section will not be present in the final version of the MIB and will be removed once all the open issues have been resolved.

1. The SMF draft states that use of the SMF Type Message TLV is optional and is used when the router runs NHDP. But the draft does not clearly state if the use of the SMF Address Block TLV is also optional.

2. Is it useful to track the effectiveness of the coverage of the current RSSA? Is it possible to track this?

3. Complete the security analysis and section.

4. Cleanup all the [TODOs] from the MIB template.

Appendix C.

```
   **************************************************************
   * Note to the RFC Editor (to be removed prior to publication) *
   *                                                            *
   * 1) The reference to RFCXXXX within the DESCRIPTION clauses  *
   * of the MIB module point to this draft and are to be         *
   * assigned by the RFC Editor.                                 *
   *                                                            *
   * 2) The reference to RFCXXX2 throughout this document point  *
   * to the current draft-ietf-manet-smf-xx.txt.  This           *
   * need to be replaced with the XXX RFC number.                *
   *                                                            *
   **************************************************************
```

Authors' Addresses

Robert G. Cole
US Army CERDEC
328 Hopkins Road, Bldg 245
Aberdeen Proving Ground, Maryland 21005
USA

Phone: +1 410 278 6779
EMail: robert.g.cole@us.army.mil
URI: http://www.cs.jhu.edu/~rgcole/

Joseph Macker
Naval Research Laboratory
Washington, D.C. 20375
USA

EMail: macker@itd.nrl.navy.mil

Brian Adamson
Naval Research Laboratory
Washington, D.C. 20375
USA

EMail: adamson@itd.nrl.navy.mil

Sean Harnedy
Booz Allen Hamilton
333 City Boulevard West
Orange, CA 92868
USA

EMail: harnedy_sean@bah.com