Internet Draft
Expiration: February 7, 1999

M. S. Corson, UMD
S. Papademetriou, UMD
P. Papadopoulos, ORNL
V. Park, NRL
A. Qayyum, INRIA

August 7, 1999

An Internet MANET Encapsulation Protocol (IMEP) Specification

draft-ietf-manet-imep-spec-01.txt

Status of this Memo

Abstract

This memo describes a multipurpose network-layer protocol---named the Internet MANET Encapsulation Protocol (IMEP)---designed to support the operation of many routing algorithms, network control protocols or other Upper Layer Protocols (ULP) (where ``upper" denotes *any* layer above IMEP) intended for use in Mobile Ad hoc Networks (MANET). The protocol incorporates mechanisms for supporting link status and neighbor connectivity sensing, control packet aggregation and encapsulation, one-hop neighbor broadcast (or multicast) reliability, multipoint relaying, network-layer address resolution and provides hooks for interrouter authentication procedures. Indirectly, the IMEP also puts forth a framework for MANET router and interface identification and addressing.

1. Introduction

The primary purpose of the Internet MANET Encapsulation Protocol (IMEP) is to improve overall network performance by reducing the *number* of network control packet broadcasts through encapsulation and aggregation of multiple MANET control packets (e.g. routing protocol packets, acknowledgements, link status sensing packets, ``network-level" address resolution, etc.) into larger IMEP messages. Usage of the IMEP is desirable because per-message, multiple access delay in contention-based schemes such as CSMA/CA, IEEE 802.11, FAMA etc. is significant, and thus favors the use of fewer, larger messages. It also may be useful in reservation-based, time-slotted access schemes where smaller packets must be aggregated into appropriately-sized IP packets for transmission in a given time slot. Upper Layer Protocols (ULP) *other than routing* may make use of this encapsulation functionality for the same purpose.

Its secondary purpose concerns the commonality of certain functionality in many network-level control algorithms. Many algorithms intended for use in a MANET will require common functionality such as link status sensing, security authentication with adjacent routers, one-hop neighbor broadcast (or multicast) reliability of control packets, etc.. This common functionality can be extracted from these individual protocols and put into a unified, generic protocol useful to all. MANET control algorithms would also benefit from a common approach to router and interface identification and addressing, and this protocol supports a framework for unifying the protocols under a common architecture.

The IMEP will run at the network layer (see Figure 1), and will be an adjunct to whichever network protocol is using it. ULP packets will be encapsulated in IMEP messages, which will be further encapsulated into IP packets.

```
      +------+ +-----+ +-----+     +-----+
      |Telnet| | FTP | | TFTP| ... | ... |
      +------+ +-----+ +-----+     +-----+    +---------+  +-----+
            |   |         |           |       | Routing |  | ULP |
           +-----+     +-----+     +-----+    +---------+  +-----+
           | TCP |     | UDP | ... | ... |            |      /
           +-----+     +-----+     +-----+          +---------+
              |           |           |     <-----  |  IMEP   |
        +---------------------------------+         +---------+
        | Internet Protocol & ICMP & IGMP |              |
        +---------------------------------+         +---------+
                          |                         |   IP    |
             +---------------------------+          +---------+
             |   Local Network Protocol  |
             +---------------------------+

              Protocol Relationships               Encapsulation
```

Figure 1

2.0 Terminology

This section provides definitions for the terminology used throughout this document. Many of these definitions may be replaced by or merged with those of the MANET working group's terminology draft now under development.

MANET router or router:

A device---identified by a ``unique Router ID" (RID)---that executes a MANET routing protocol and, under the direction of which, forwards IP packets. It may have multiple interfaces, each identified by an IP address. Associated with each interface is a physical-layer communication device. These devices may employ wireless or hardwired communications, and a router may simultaneously employ devices of differing technologies. For example, a MANET router may have four interfaces with differing communications technologies: two hardwired (Ethernet and FDDI) and two wireless (spread spectrum and impulse radio).

medium:

A communication channel such as free space, cable or fiber through which connections are established.

communications technology:

The means employed by two devices to transfer information between them.

connection:

A physical-layer connection---which may be through a wired or wireless medium---between a device attached to an interface of one MANET router and a device utilizing the same communications technology attached to an interface on another MANET router.

link:

A ``logical connection" consisting of the logical *union* of one or more connections between two MANET routers--identified by a (RID, RID) pair. Thus a link may consist of a heterogeneous combination of connections through differing media using different communications technologies.

neighbor:

From the perspective of a given MANET router, a ``neighbor" is any other router to which it has a link.

adjacency:

The name given to an ``interface on a neighboring router". From the perspective of a given router, a connection is a (interface, adjacency) pair.

topology:

A network can be viewed abstractly as a ``graph" whose ``topology" at any point in time is defined by set of ``points" connected by ``edges". (This term comes from the branch of mathematics bearing the same name that is concerned with those properties of geometric configurations (such as point sets) which are unaltered by elastic deformations (such as stretching) that are homeomorphisms.)

physical-layer topology:

A topology consisting of connections (the edges) through the *same* communications medium between devices (the points) communicating using the *same* communications technology. Multiple physical-layer topologies may exist for a given medium and communications technology if adaptive or proactive power control, frequency or code division, or other physical-layer mechanisms are employed.

network-layer topology:

A topology consisting of links (the edges) between MANET routers (the points) which is used as the basis for MANET routing. Since ``links" are the logical union of physical-layer ``connections", it follows that the ``network-layer topology" is the logical union of the various ``physical-layer topologies".

IP routing fabric:

The heterogeneous mixture of communications media and technologies through which IP packets are forwarded whose topology is defined by the network-layer topology.

Security Context:

A security context between two routers defines the manner in which two routers choose to mutually authentication each other, and indicates an authentication algorithm and mode.

Mobility Security Association:

A collection of security contexts, between a pair of routers, which may be applied to IMEP protocol messages exchanged between them.

Security Parameter Index (SPI):

An index identifying a security context between a pair of routers among the contexts possible in the Mobility Security Association.

3.0 Protocol Overview

The mechanisms contained in the IMEP are:

Message Aggregation (AGGR)

Network-layer Address Resolution (NARP)

Link/Connection Status Sensing (LCSS)

Broadcast Reliability (REL)

Multipoint Relaying (MPR)

Authentication (AUTH)

Message aggregation occurs as packets from ULPs become IMEP objects, and IMEP packs a number of objects into larger IMEP messages for transmission. NARP--a protocol to determine the *binding* of a RID with each of its IP interface address--occurs implicitly in the current specification as the router ID of a given router is put in the header of each IMEP message. As each IMEP packet is encapsulated in an IP packet, and its header contains the IP address of the transmitting interface in the source field of the IP packet, the binding can be made on reception of any IMEP packet (more on this later). Usage of the remaining mechanisms is *optional*. The following dependency graph shows their relationships.

```
                           AGGR---NARP
                             |
                      +------+-------+
                      |      |       |
                     REL   LCCS    MPR
                      |
                 -----+----
                 |        |
               AUTH      MPR
```

This simply means that everything uses IMEP's aggregation facility. NARP occurs implicitly in every IMEP transmission. Usage of reliability, LCCS, MRP and AUTH are optional. MRP traffic may be sent reliably or unreliably. Authentication, if enabled, occurs reliably.

3.1 Relationship with Upper Layer Protocols

IMEP is intended to support the operation of many ULPs. ULPs that wish to utilize IMEP must dynamically *register* with an IMEP implementation prior to using IMEP (more on registration in a moment).

3.1.1 Protocol Type Values

All ULPs which intend to utilize IMEP must have protocol type value, and must give this value to IMEP during registration. This value is used by a receiving IMEP implementation for purposes of demultiplexing ULP objects within a received IMEP message so that they may be passed to the appropriate ULPs. IMEP implementations receiving objects with unknown (i.e. unregistered) protocol type values will silently discard those objects. Several protocol types have already been assigned well-known values (see the protocol grammar section), but a protocol need not have a pre-assigned type value to make use of IMEP, nor must the well-known assignments be adhered to. IMEP currently does not specify how protocol type values are assigned or used within a given administrative domain.

3.1.2 Protocol Handles

ULPs registering with IMEP must pass to IMEP a protocol ``handle" which IMEP may then use to pass information back to the ULP. The mechanism used to implement the handle is not specified (this is implementation dependent)--it could be a pointer to a function with a known signature, an object reference in a middleware-based implementation, etc..

3.1.3 Protocol Epitaphs

ULPs registering with IMEP must specify an ``epitaph" object. The epitaph object specifies a signal to be broadcast reliably to all one-hop peer ULPs if the registered ULP fails. This permits peer ULPs (on neighboring routers) to take appropriate action in case of peer process failure. Protocols may re-register with IMEP at any time in order to change the epitaph object, or to remove it if desired.

Registration with an ``epitaph" object amounts to creating and maintaining a symbiotic relationship between IMEP and a registered ULP. There must exist a mechanism (not specified--implementation dependent) that guarantees ``mutual liveness" to each protocol so that, should either protocol fail, the other is reliably informed within the time of a BEACON_PERIOD (defined subsequently).

The principle purpose for epitaph-based registration is *bandwidth conservation*. Without such a mechanism, it is not possible for peer ULP processes--who have previously exchanged control information and remain connected via IMEP--to be assured of mutual vitality without exchanging keepalive packets over the communication channel.

3.1.4 IMEP Signalling Support

ULPs registering with IMEP must indicate the level of IMEP signalling support (ISS) they wish to receive from IMEP. IMEP signalling support is only meaningful if LCSS is enabled, and consists of signals being generated by IMEP in response to topological change events detected by LCSS, and then passed to subscribing ULPs (those ULPs requesting ISS). Three levels of support are possible:

0) Connection-level:
All connection-level topological change events are passed to the subscribing ULPs. Connection-level topological change events consist of ``connection" activation and failure (recall a connection consists of an (interface, adjacency) pair). Thus, all physical-layer topology information is passed to the ULPs, permitting these ULPs to have a complete internal view of the IP routing fabric.

1) Link-level:
All link-level topological change events are passed to the subscribing ULPs. Link-level topological change events consist of ``link" activation and failure (recall a link consists of a (RID, RID) pair). Thus, only network-layer topology information is passed to the ULPs, permitting these ULPs to have only an external view of the IP routing fabric.

2) Disabled:

No topological change events generated by IMEP as a result of LCSS are passed to the ULP. This is the default mode.

3.1.5 ULP Registration

ULPs must register with IMEP *prior* to usage. ULP registration consists of passing IMEP a protocol type value, a *handle* to the ULP allowing IMEP to pass received objects to it (handle mechanism not specified--implementation dependent), an *epitaph* object (this may be null), and a parameter indicating the level of IMEP signaling support desired by the ULP.

3.2 Message Aggregation

MANET routing (and other) control protocols exchange control information and other data in the form of routing control packets or ``objects". To minimize the number of channel accesses generated by control traffic, the IMEP aggregates and encapsulates these objects into larger IMEP ``messages". The objects are treated as ``opaque" objects by the IMEP protocol; i.e. IMEP is not aware of the contents of the objects, only of the protocol ``type" of the object block (necessary for protocol demultiplexing at a receiver) and the length of each object. These ULP object blocks are contained in yet larger IMEP messages which are passed to the IP layer for encapsulation and forwarding. A single IMEP message can contain a mixture of reliable and unreliable objects. The details can be found in the IMEP message format section.

3.3 Network-level Address Resolution

IMEP supports a framework or architecture for MANET router and interface identification and addressing. IMEP operates simultaneously on two different topological levels: the ``logical network" topology level---which is concerned with interrouter connectivity---and the ``physical" topology level---which is concerned with interface connectivity. Router IDs (RID) identify routers in the logical topology, and IP addresses identify interfaces in the physical topology. There may be *multiple* IP addresses associated with a given RID.

The purpose of a Network-level Address Resolution Protocol (NARP) is to discover the mapping between RIDs and IP addresses. This is envisioned typically only to be needed when a new connection is discovered, as it is necessary to be able to associate an interface (an IP address) with a router (an RID).

```
                   +----------+
                   |  Router  |                         RID
                   +----------+
                     |       |
         +--------------+   +--------------+
         |  Interface   |   |  Interface   |       IP Address
         +--------------+   +--------------+
                |                   |
         +--------------+   +--------------+
         | Phys Device  |   | Phys Device  |       MAC Address
         +--------------+   +--------------+
```

Figure 4: RIDs, IP and MAC addresses

While it is true that---as currently defined---RIDs are not ``addresses" in the strict sense, they do uniquely identify a router for purposes of internal routing computations and somewhat resemble a logical ``router address". Thus, the IP address-to-RID mapping is similar in spirit to IP address-to-MAC address mapping performed by the present ARP protocol. Each mapping simply associates an IP address with another identifier as shown in Figure 4. As with ARP, a ``reverse" mapping is also defined as the Reverse Network-level Address Resolution Protocol (RNARP). However, unlike RARP, a RNARP request seeks to discover the *set* of IP addresses associated with a given RID. The two mappings are shown in Figure 5.

```
ARP:  IP --> MAC     RARP:  MAC --> IP

NARP: IP --> RID     RNARP: RID --> {IP1,IP2,...,IPn}
```

Figure 5: ARP/RARP and NARP/RNARP

NARP is currently implemented *implicitly* through inclusion of the RID in every IMEP message header. RNARP is not required in the present specification, but may be specified and required in future versions if deemed necessary.

3.4 Link/Connection Status Sensing

3.4.1 Definition of Link/Connection Status

Link/Connection Status Sensing (LCSS) is an optional mode that may be enabled in IMEP. Many control protocols require accurate knowledge of the status of links/connections between neighboring routers. ``Link status" in the IP routing fabric is determined from the union of the status of physical-layer ``connections" between interfaces.

The relationship of interfaces, adjacencies, connections and links is depicted in Figure 2 from the perspective of router i. Router i has two interfaces f1 and f2, each of which has a physical-layer connection with multiple interfaces attached to other routers---these interfaces are referred to as adjacencies from router i's perspective and are numbered with a's. In this figure, there are two connections (f1,a1) and (f2,a2), the logical union of which composes the logical link (i,k) between routers i and k.

```
                +----------+
                | Router i |
                +----------+
    +--------------+    +--------------+
    | Interface f1 |    | Interface f2 |
    +--------------+    +--------------+
            |                   |
            |                   |
            |                   |
            |                   |
            |                   |
            |                   |
    +--------------+    +--------------+
    | Adjacency a1 |    | Adjacency a2 |
    +--------------+    +--------------+
                +----------+
                | Router k |
                +----------+
```

Figure 2: Shown from router i's perspective, interfaces f1 and f2 are connected to adjacencies a1 and a2 via connections (f1,a1) and (f1,a2)---the union of which forms link (i,k).

The status of a connection may be INcoming or OUTgoing (either of which meaning it is unidirectional) or BIdirectional. A unidirectional link is composed from one or more similarly-directed, unidirectional connections. A BIdirectional link may be composed from the union of one or more bidirectional connections, or two or more oppositely-directed, unidirectional connections, or some combination thereof. A connection or link which is present or ``active" (i.e. which has a non-null status, and is either uni or bidirectional), is referred to as ``UP". A connection or link which is not active (i.e. which has a null status) is referred to as ``DOWN".

The IMEP may be configured to run in the following ``connection notification" modes:

BI-directional:
This mode requires that physical-layer connectivity between an interface and an adjacency be established in *both* IN and OUT

directions before a connection is considered UP, and any registered ULPs are subsequently notified.

UNI-directional:
This mode requires that physical-layer connectivity between an interface and an adjacency need only be established in one direction (IN or OUT) before a connection is considered UP and the registered ULPs are subsequently notified.

As determined by the connection notification mode, the ULP is notified whenever there is a change (addition, modification, deletion) in the status of an interface's connections. This notification is implemented via a handle registered via the ULP/IMEP interface.

3.4.2 Link/Connection Status Sensing Packet Exchange Mechanism

The IMEP uses a combination of BEACON and ECHO packets (and other equivalent packets to be described shortly) to ascertain connection (and indirectly link) status. On initialization, an interface under the control of IMEP broadcasts a BEACON packet to all adjacencies. (Note: The format of a BEACON packet is specified in a later section, but it essentially consists of an *empty* IMEP message; i.e. an IMEP message containing only the IMEP message header.). Recall that adjacencies are interfaces that are only one hop away such as those on the same Ethernet subnet, or those within wireless transmission range of the broadcasting interface. (Note: Usage of the term ``broadcast" here means to transmit a *single* copy of a packet to *all* interfaces reachable over one hop.  As is the convention with other Internet routing protocols, this is done using IP multicast. An IP multicast address ``ALL_IMEP_ROUTERS" will be reserved with IANA, and all MANET router interfaces will be configured to listen for this address.) The purpose of a BEACON packet is to alert any adjacencies of the existence and identity of the broadcasting interface; an interface's identity is its IP address. The interface must ensure that a BEACON packet (or *any* other packet, since all packets are ``BEACON-equivalent") is transmitted at least once every BEACON_PERIOD (BP) time units; i.e. no more than BP time units may pass between subsequent transmissions of a BEACON (or ``BEACON-equivalent") packet.

Reception of a BEACON at an interface implies either reconfirmation or creation of ``IN" (read ``INcoming") status of a connection at that interface, depending on whether or not the connection already exists, respectively. Thus, BEACONs serve to tell a receiving interface that ``someone else is out there." Once present, the status remains for MAX_BEACON_TIME (MBT) time units, at which time it times out if no subsequent BEACONs have been received; i.e. the link is declared DOWN and is removed from the data structures. Creation or

loss of IN status may require notification of an upper level protocol, depending on its signalling support mode.

ECHO (or ``ECHO-equivalent") packets are used to respond to BEACONs. The purpose of an ECHO packet is to let a ``BEACONing" router know that someone hears its BEACON. An ECHO packet contains the identity (i.e. IP interface address) of the interface broadcasting the ECHO and the identity of the BEACONing interface to which it is responding. An ECHO packet is generated immediately in response to an initial BEACON reception. Subsequently, as long as the interface is considered UP (i.e. IN or BI), an ECHO packet must be generated at least once every BP time units; i.e. no more than BP time units may pass between subsequent generations of an ECHO or ECHO-equivalent packet.

Reception of an ECHO at an interface implies either reconfirmation or creation of ``BIdirectional" status of an connection at that interface, depending on whether or not the connection already exists, respectively. This is because reception of ECHO packet confirms that someone hears this interface (i.e. that is has OUTgoing status), and simultaneously confirms that it itself can receive them and, hence, also has INcoming status for that connection.

ECHO packets may be broadcast in accordance with one of two ``signalling" modes, which applies to both ECHO and ACK semantics (more on

ACKs later):

Single Interface (SI):

An interface only sends ECHOs in response to BEACONs it receives. This is the standard mode which permits efficient link-layer detection of BI connections. This mode should be enabled if the BI-directional connection notification mode is enabled.

Multiple Interface (MI):

An interface sends ECHOs in response to BEACONs it receives, and IMEP also sends Indirect ECHOs (IECHO) out *all* other interfaces. An IECHO carries the address of the interface being echo'ed (as does an ECHO) but, additionally, carries the address of the interface on the echoing router that received the transmission being echoed. This mode is necessary to permit ``IMEP-based detection" of BIdirectional links composed of oppositely-directed, unidirectional connections between neighboring routers. Note that by using this Echo mode (i.e. via reception of IECHOS at other interfaces), an interface can be informed (solely via IMEP) that it has an ``OUTgoing" connection without also having ``INcoming" status and, hence, a BIdirectional link. This mode should be enabled if the UNI-directional

connection notification mode is enabled.

To make this clear, consider Figure 3.

```
                      +-----------+
                      | Router i |
                      +-----------+
          +--------------+    +--------------+
          | Interface f1 |    | Interface f2 |
          +--------------+    +--------------+
                  |             IN    ^
                  |                   |
                  |                   |
                  |                   |
                  |                   |
            IN    V                   |
          +--------------+    +--------------+
```

```
        | Adjacency c1 |    | Adjacency c2 |
        +--------------+    +--------------+
                    +-----------+
                    | Router k |
                    +-----------+
```

Figure 3: A bidirectional link consisting of two oppositely-directed connections.

Assume that SI Echo mode is being used, and the wireless directional connectivity is as shown. From router i's perspective, it can only receive over interface f2, and thus classifies connection (f2,c2) as IN. It is unaware that its BEACON packets being broadcast from interface f1 are being received at interface c1 on router k. However, if MI mode is used, then router k will advertise (via IECHO transmissions from c2) the reception of BEACON packets from f1 at c1 thereby informing router i that connection (f1,c1) should be classified as OUT. Of course, the reverse but same is true from router k's perspective.

The additional functionality provided by the MI mode comes at the cost of broadcasting IECHOs out one or more interfaces in addition to the ECHO sent over the interface over which the corresponding BEACON was received. This creates more ECHO overhead. For a given network, this cost must be balanced against the frequency of occurrence of the situation depicted in figure 3.

Additional activity at an ULP (involving communication over multiple hops) is necessary to detect purely UNIdirectional links (i.e. links consisting of one or more unidirectional connections) between adjacent routers.

3.4.3 BEACON and ECHO ``Equivalency"

BEACON and ECHO packets are necessary for ascertaining current connection status. From the perspective of a given router, BEACONs announce the presence of a broadcasting interface, and ECHOs simultaneously announce the presence of an adjacency *and* that the adjacency can receive from the broadcasting interface. However, it should be clear that the same information can be gleaned from other IMEP packets. Specifically, all transmissions signal the presence of a broadcasting interface and are, in this sense, ``equivalent" to BEACON packets. Similarly, ACKnowledgements both announce the presence of an adjacency and, through the process of acknowledgement, confirm that the adjacency recently received from the broadcasting interface. Thus, in this sense, ACKs are equivalent to ECHOs. The equivalency is depicted in the Figure 6.

```
                              BEACON  -->
                              ALL/OBJ -->
+----------+ +-------------+              +-------------+
| Router i |-| Interface f |  -  -  -  -  | Adjacency c |
+----------+ +-------------+              +-------------+
                              <-- ECHO or IECHOS
                              <-- ACK or IACKS
```

Figure 6: BEACON and ECHO Equivalency

Transmission or reception of a BEACON or ECHO-equivalent packet affects the link-status sensing timers as would transmission or reception of a BEACON or ECHO, respectively. Thus, during periods of heavy data traffic, it is expected that BEACONs and ECHOs will rarely be transmitted as their respective ``equivalent" packets will serve their role in link status sensing. During periods of light or no traffic, BEACONs or ECHOs will be transmitted as necessary to satisfy the aforementioned timing requirements.

If MI mode is in use, the Indirect ECHOS are being sent out all interfaces. In a corresponding fashion, Indirect ACKS (IACKS) must be sent out all interfaces to provided reliability over BIdirection links consisting of oppositely-direction, UNIdirectional connections. These IACKS are also ``echo equivalent" and must indicate the address of the interface they are IACKing, as well as the interface address on the IACKing router which received the object being indirectly ACKed.

3.4.4 Connection Failure Detection

Expiration of the MBT timer signals connection failure. Note that separate timers are used to monitor IN and OUT connection status. Thus, a connection may lose its OUT status while still retaining IN status and vice versa. Obviously, a connection satisfying both IN and OUT timing requirements is marked as BI.

3.5 Neighbor Broadcast Reliability

IMEP supports two broadcast delivery modes:

BROADCAST (IMPLICIT):
Delivery to the current one-hop neighbor set.

MULTICAST (EXPLICIT):
Delivery to a pre-specified subset of the one-hop neighbor set. A ULP may specify one, some or all current neighbors.

Of course, both are delivered using one-hop scoped, multicast addressing as is every IMEP message.

IMEP supports two reliability modes:

UNRELIABLE:
Unreliable, unsequenced delivery of either neighbor broadcast or neighbor multicast data.

RELIABLE:
Reliable, sequenced delivery of either neighbor broadcast or neighbor multicast data.

Thus, delivery may be implicit or explicit, and reliable or unreliable: all four combinations are possible. These modes are used for delivery of opaque protocol objects, where reliable delivery-- i.e., broadcast or multicast --is also guaranteed to be delivery ``in order" of transmission. (Note: This should not be confused with transport-layer, reliable multicast across an entire multihop network.)

IMEP uses a ``point-to-multipoint selective repeat" algorithm to guarantee broadcast or multicast reliability and ordered delivery. This approach eliminates unnecessary retransmissions of the type commonly associated with ``go back n" algorithms, and is in keeping with the greater IMEP goal of minimizing the number of required channel accesses.

To support reliability, each object block is given a SEQUENCE number, and is broadcast with that number. To provide in-order delivery, a connection protocol is utilized to synchronize receivers with the current broadcast SEQUENCE number. The connection and transmission protocol is designed so that an explicit receiver list does not have to be appended to every reliable object block. Instead, an implicit list is used by ``coloring" all messages. If a message is received with the correct color, then the SEQUENCE number has meaning and its objects can be forwarded up the protocol stack. If the color is incorrect, the receiver does not forward its objects up the protocol stack. The connection protocol reliably transmits the current group color to all members of the group.

When broadcast, a copy of the object block with a response list (i.e. the set of neighbors that are required to acknowledge this block) is stored. A retransmission timer is set to RETRANS_PERIOD (RP) time units which, upon expiration, will cause the object to be rebroadcast to any neighbors which have not acknowledged the object (this causes the retransmission timer to be set again to RP). The time the packet was initially broadcast is also stored. If the object's response list is not empty (i.e it has not been acknowledged by some adjacencies) after MAX_RETRANS_TIME (MRT) time units, the connections to those adjacencies are declared DOWN.

Acknowledgements (ACKs) are sent in response to object block receptions when (i) reliable delivery is indicated and (ii) when the receiver is contained in the response list (either implicitly or explicitly). Once a neighboring router has ACKed a given block, it will be removed from the block's response list so that it will not be required to ACK any future retransmissions.

3.5.1 The Reliable Delivery Neighborhood

Each router keeps track of the neighbors that can be reached reliably through regular Beacon-Echo exchanges. For discussion purposes, consider a single router, termed a ``base-router", B and any number of ``neighbor routers", N(i), i=1,2, ..., P, where P is the number of routers that can currently hear transmissions from B. Each router N(i), will respond with an ECHO packet within the time constraints of the BEACON-ECHO protocol outlined previously. If B hears an ECHO packet from N(i), then N(i) is a candidate member of B's reliable delivery neighborhood (RDN). For N(i) to become a member of B's reliable delivery neighborhood (i.e., connected to B), B must broadcast a group COLOR with an explicit membership list. This object is called a NEWCOLOR and must be acknowledged by every router on the explicit membership list before B considers a reliable delivery neighborhood to be formed.

From N(i)'s perspective, the neighborhood rooted at B is has COLOR K. N(i) is a member of this neighborhood if the NEWCOLOR object explicitly contains N(i) as a member. A reliable delivery neighborhood

rooted at B with COLOR K and current sequence J is specified in the triple RDN(B,K,J). The COLOR K is updated by B every time a change to its RDN is discovered (either a new router comes in range or an existing router moves out of range or becomes hidden). Every router R in a MANET network will have a single RDN rooted at R. R can be a member of any number of RDN's that are not rooted at R. Every router keeps track of its RDN and of the RDN's for which it is a member. If a router hears a router R1 but itself is not an explicit member of RDN(R1,K,J), then it marks the current COLOR of RDN(R1,K,J) as colorless or as RDN(R1,0,J). The format for a NEWCOLOR object is given in a later section.

3.5.2 Neighborhood definitions

RDN(B):
Reliable delivery neighborhood rooted at MANET router B.

RDN(B,K):
Reliable delivery neighborhood rooted at MANET router B, with COLOR K.

RDN(B,K,J):
Reliable delivery neighborhood rooted at MANET router B, with COLOR K, and current broadcast sequence number J.

3.5.3 Reliable, Sequenced Delivery

Objects passed to IMEP from an ULP may be delivered reliably or unreliably, and is specified by the ULP. This section addresses reliable, sequenced delivery of ULP objects by IMEP to all members of a RDN. Every reliable object in IMEP delivered from B to the RDN(B,K,J) is colored with COLOR K and sequence number J. A router N(i) is an intended receiver of the object if its notion of the COLOR K associated with RDN(B) matches exactly the color contained in the broadcast object. Therefore, N(i) may deliver a reliable object to its ULP only if the object from B matches the COLOR and SEQUENCE that

N(i) has recorded for the RDN(B). If an object arrives with the correct COLOR but the incorrect SEQUENCE number, then N(i) must determine if the object is a duplicate or simply out of sequence. If a duplicate, then N(i) discards the object. If out of sequence, then N(i) retains the object until all earlier objects arrive. If an object arrives with the incorrect COLOR, then N(i) discards the object.

From the ULP's perspective, objects are delivered reliably and in sequence to *only* those members of the RDN(B) that exists at the time when the object was received by IMEP (Note this may not be the time when the object was sent to IMEP from the ULP's perspective, due possibly to interprocess communication delay between IMEP and the local ULP). This is referred to as an (implicit) ``neighbor broadcast" object.

If the ULP requires a object to be delivered to a specific subset of one-hop neighbors, then it should use ``neighbor multicast" objects (see below). This latter delivery semantic frees ULPs from having to decide whether or not a object is valid. Every reliable object passed to the ULP from IMEP is guaranteed to be intended for the ULP, as specified by the sender.

Reliability is established between *routers*, not interfaces. Thus, the reliability semantics are the same regardless of whether BIdirection notification with SI signalling or UNIdirectional notification with MI signalling is in use.

3.5.3.1 Sequence Numbers and Associations using Broadcast Semantics

The coloring of the RDN(B) corresponds to a single sender with a number of ``associated" receivers. ECHOs from a router can be viewed as a association request. If an association is already established from B to N(i), then this request is vacuous. If, however, no association from B to N(i) exists, the ECHO then acts like a association request. A NEWCOLOR object with N(i) on the list completes the association from B to N(i) (from N(i)'s perspective) and N(i)'s acknowledgement of the NEWCOLOR object completes the association from B's perspective.

The RDN(B) maintains a single sequence number that all members of

RDN(B) must track. NEWCOLOR objects contain not only a new group COLOR, but also the next expected SEQUENCE number. This allows sender and receivers to synchronize the sequence numbers to provide in-order delivery.

There are (subtle) consequences of these semantics.

1) An RDN(B) maintains a *single* sequence number for the neighborhood. Hence, every N(i) must acknowledge *every* reliable object to ensure that all members of RDN(B) maintain the sequence order. Of course, multiple reliable objects contained in the same IMEP message are acknowledged simultaneously with a single ACK. If an object is intended for a single recipient, all must acknowledge (to keep sequence numbers synchronized) and information specific to this object must further designate the intended

recipient. This is due to the fact that the current scheme is optimized for implicit neighbor broadcast delivery, not explicit neighbor multicast.

2) When RDN(B,K0) is updated to RDN(B,K1) (color changes from K0 to K1), then all reliable objects must first be retired from B's retry queue before the NEWCOLOR object can be transmitted.

3) The explicit association (via a colored neighborhood) means that the first time a reliable object is transmitted, an explicit recipient list can be (and is) omitted. This reduces the size of objects and allows the receiver to determine if it should forward the object up the protocol stack based on only the COLOR and SEQUENCE number of the object. An additional feature of this association is that if a single receiver fails to acknowledge an object, an explicit recipient list may be appended to the reliable object to indicate those routers that should re-ack the object. In the case of delivery failure, this reduces the number of a media accesses by requiring only those who have not acknowledged a object to explicitly respond.

3.6 Multipoint Relaying

IMEP supports Multipoint Relaying (MR)--an optional mode or mechanism designed to minimize the overhead of packet *flooding* throughout a MANET by optimizing/reducing the number of duplicate retransmissions.

As control overhead expenditure is required to support MR, it is recommended that this mode be enabled only when sufficient flooding traffic exists so that the benefit derived from MR justifies its cost.

Before describing MR in detail, we first give some terminology specific to MR:

MultiPoint Relay (MPR):
A router which is selected by a one-hop neighbor to forward or retransmit that neighbor's packets.

Multipoint Relay Selector (MPRS):
Each MPR has one or more neighbors which have selected it as a MPR--each such neighbor is referred to as a ``Multipoint Relay Selector". Each MPR keeps a table of RIDs identifying the members of its MRS set so that it knows which packets to retransmit via MR.

Source of the Multipoint Relay (SMR):
Each router which originally transmits a data packet via MR is known as the ``Source of the Multipoint Relay" for that packet, and is so identified in the packet.

Every router has a set of nodes one hop away N1 (its one-hop neighbor set) and a set of nodes two hops away N2 (its two-hop neighbor set). The objective of a router participating in MR is to select a minimal subset M of MPRs from N1 so that their retransmissions cover N2.

Multipoint relaying proceeds as follows:

Each MR router periodically broadcasts a Multipoint Relaying Advertisement (MRA) packet once every Multipoint Relaying Period (MRP) containing its RID, the RIDs of all its one-hop neighbors in N1, and the subset M of these neighbors it has selected as its MPRs. This is an implicit broadcast to the current one-hop neighbor set N1 which may occur reliably or unreliably as desired. It can easily be seen that with each MR router transmitting the identity of its set N1, every MR router learns its set N2.

The algorithm for selection of the set M is not prescribed. It is required only that the set M be chosen so as to cover N2. The aim is to select the ``minimum" number of MPRs to do so.

One possible algorithm is:

1. Start with an empty set M.
2. First select as MPRs those routers from F1 which provide the ``only path" to reach some routers in N2.
3. While there still exist some routers in N2 that are not covered by M:
   - 3.1 For each router in N1, calculate the number of routers in N2 reachable through this router which are not yet covered by M;
   - 3.2 Select as a MPR that router which reaches the maximum number of uncovered routers in R2.

A ``flood termination" mechanism is also required and is implemented simply by including a SMR field and a sequence number in every MR object. This enables routers to maintain a list of recently-received MR objects. MR objects are passed to the appropriate ULP the *first* time they are recieved at a router, and are silently discarded thereafter.

3.7 Authentication

Authentication is optional. If authentication is enabled, MANET routers have the choice of implementing multiple authentication options ranging from simple to complex. IMEP messages between MANET routers are authenticated with the IMEP Authentication object, which contains the option is use. This object immediately follows all non-authentication objects.

4. IMEP Message Format

The following describes the message format of the proposed protocol. An IMEP message format consists of several fixed, mandatory fields followed by a self-formatting byte stream. The stream is aligned along ``byte" boundaries---not 32-bit word boundaries---to save transmission overhead at the cost of extra processing at a router. An IMEP message typically contains at least one of several optional object blocks. A message containing no objects is a BEACON message.  The following ``grammar" describes the syntax of an IMEP message.

```
<IMEP message>      :  <IMEP_MSGHDR> <IMEP_OBJECTLIST>

<IMEP_MSGHDR>       :  <IMEP_VERSION> <COLOR> <MESSAGE_LENGTH> <RID>

<IMEP_OBJECTLIST>   :  <IMEP_OBJECTLIST> <IMEP_OBJECT>
                    |  <IMEP_OBJECT>

<IMEP_OBJECT>       :  <OBJECT_HDR> <RELIABLE_OBJECT>
                    |  <OBJECT_HDR> <UNRELIABLE_OBJECT>

<OBJECT_HDR>        :  <OBJTYPE> <SEQUENCE> <OBJECT_LENGTH>

<RELIABLE_OBJECT>   :  <DATA>
                    |  <DATA> <ACK List>

<UNRELIABLE_OBJECT> :  <DATA>

<DATA>              :  <ECHO>
                    |  <BCAST>
                    |  <MCAST> <DELIVERY_LIST>
                    |  <MR>
                    |  <ACK>
                    |  <NEWCOLOR>
                    |  <MRA>
                    |  <AUTH>

<BCAST>             :  <PROTOCOL> <OBJLEN> <OBJDATA>

<MCAST>             :  <PROTOCOL> <OBJLEN> <DELIVERY_LIST_LEN>
                       <OBJDATA>

<MR>                :  <SMRRID> <MRSEQUENCE> <PROTOCOL>
                       <OBJLEN> <OBJDATA>
```

4.1 <IMEP_MSGHDR>

Every IMEP message contains header information. A message with no objects is termed a BEACON message. Included in every header is the <RID> of the sending IP interface.

```
<IMEP_MSGHDR> : <IMEP_VERSION> <COLOR> <MESSAGE_LENGTH> <RID>

 31           24 23           16 15            8 7            0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (a)      |      (b)     |             (c)               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

 31           24 23           16 15            8 7            0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (d)                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+


(a) <IMEP_VERSION> Protocol version (8 bits)

(b) <COLOR> Group color (8 bits)
    ==  0      - colorless
    otherwise  - reliability sequence numbers are prefixed by
                 this color

(c) <MESSAGE_LENGTH> Total message length (in bytes) of this
    IMEP packet (16 bits) which lies in the following range:

          3 < IMEP_LENGTH <= MAX_IMEP_LENGTH <= 65535

(d) <RID> Router Id associated with the sender's IP interface.
```

4.1.1 <OBJECT_HDR>

```
 31           24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (a)      |      (b)      |              (c)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
(a) <OBJTYPE> object type (8 bits)
    0       - reserved
    1-127   - object does not carry reliability information,
              seq# ignored
    128-255 - object must be delivered reliably, in order,
              according to color and seq #
```

(b) <SEQUENCE> Sequence number for this object (8 bits)

(c) <OBJECT_LENGTH> Length (in bytes) of this object (16 bits). <OBJECT_LENGTH> does not include the length of the SUBMESSAGE HEADER, but does include the length of the explicit ack list, if any.

(<OBJECT_LENGTH> <= <MESSAGE_LENGTH> - 4)

4.1.2 <OBJTYPE>

The following object types are defined for this version of IMEP.

Unreliable Object Types:

1 - SM_ECHO : <ECHO> object
2 - SM_ACK : <ACK> object
3 - SM_UBCAST : <BCAST> object, delivered unreliably
4 - SM_UMCAST : <MCAST> object, delivered unreliably
5 - SM_UMRA : <MRA> object, delivered unreliably
6 - SM_UMR : <MR> object, delivered unreliably
7 - SM_IECHO : <IECHO> object
8 - SM_IACK : <IACK> object
[65,73] : (future) IPV6 Versions of the above objects

Reliable Object Types:

128 - SM_NEWCOLOR : <NEWCOLOR> object
129 - SM_BCAST : <BCAST> object delivered reliably
130 - SM_MCAST : <MCAST> object delivered reliably
131 - SM_AUTH : <AUTH> object delivered reliably
132 - SM_MRA : <MRA> object, delivered reliably
133 - SM_MR : <MR> object delivered reliably
[192,197] : (future) IPV6 Versions of the above objects

4.2 IMEP objects

This section describes the ordering of IMEP objects a MANET router may include in an IMEP message. This following ordering MUST be followed:

a) The fixed-length IMEP message header, followed by

b) If present, any non-authentication objects, followed by

c) The IMEP Authentication object.

The authentication in the IMEP messages MUST be checked. The receiving router MUST check for the presence of a valid IMEP Authentication object, and perform the indicated authentication. Exactly one IMEP Authentication object MUST be present in the IMEP message, and the home agent MUST check the Authenticator value in the object. If no IMEP Authentication object is found, or if more than one IMEP Authentication object is found, or if the Authenticator is invalid, the receiving router MUST discard the IMEP message and SHOULD log the

---

error as a security exception.

4.2.1 <ECHO>

The <ECHO> block may contain any number (subject to message length restrictions) of Addresses

```
<ECHO>     : <ECHO_LIST>

<ECHO_LIST> : <ECHO_LIST> <ECHO_ENTRY>
            | <ECHO_ENTRY>

<ECHO_ENTRY> : <ECHO_IF>
```

A <ECHO_ENTRY> is a 32-bit address that contains the interface being echo'ed.

```
 31           24 23          16 15           8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (a)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <ECHO_IF> IPV4 of interface that is being echo'ed (4 bytes)

The number of addresses in this list are inferred from the <OBJECT_LENGTH> field.

4.2.2 <ACK>

The ACK Block format is:

```
<ACK>       :  <Ack List>

<Ack List> :  <Ack List> <Ack Entry>
           |  <Ack Entry>

<Ack Entry> : <ACK_IPADDR> <ACK_COLOR> <ACK_SEQUENCE>
```

<Ack Entry> is defined as follows: This block may contain any number (up to total length restrictions) of acknowledgements interfaces and sequence #'s

numAcks = <OBJECT_LENGTH>/6

ACK Block 6-byte byte block:

```
31             24 23             16 15              8 7               0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (a)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

15              8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (b)       |      (c)      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <ACK_IPADDR> IPV4 address of interface being ACKed (4 bytes)
(b) <ACK_COLOR> Group Color (8 bits)
(c) <ACK_SEQUENCE> object sequence# (8 bits)

4.2.3 <IECHO>

The <IECHO> block may contain any number (subject to message length restrictions) of <IECHO_ENTRY>s.

```
<IECHO>       : <IECHO_LIST>

<IECHO_LIST>  : <IECHO_LIST> <IECHO_ENTRY>
              | <IECHO_ENTRY>

<IECHO_ENTRY> : <ECHO_IF> <RCV_IF>
```

A <IECHO_ENTRY> consists of two 32-bit addresses that contain the interface being echo'ed by the router and the interface which received the BEACON-equivalent, for which this is an *indirect* echo.

```
 31            24 23            16 15             8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (a)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 31           24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (b)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <ECHO_IF> IPV4 of interface that is being echo'ed (4 bytes)

(b) <RCV_IF> IPV4 of interface of the receiving interface (4 bytes)

The number of entries in this list are inferred from the <OBJECT_LENGTH> field.

4.2.4 <IACK>

The <IACK> Block format is:

```
<IACK>       :  <IACK_LIST>

<IACK_LIST>  :  <IACK_LIST> <IACK_ENTRY>
             |  <IACK_ENTRY>

<IACK_ENTRY> : <ACK_IPADDR> <RCV_IPADDR> <ACK_COLOR> <ACK_SEQUENCE>
```

<IACK_ENTRY> is defined as follows: This block may contain any number (up to total length restrictions) of indirect acknowledgements.

numIAcks = <OBJECT_LENGTH>/10

IACK Block 10-byte byte block:

```
      31            24 23           16 15            8 7             0
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                              (a)                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

      31            24 23           16 15            8 7             0
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                              (b)                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

      15             8 7             0
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |      (c)      |      (d)      |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

       (a) <ACK_IPADDR> IPV4 address of interface being IACKed (4 bytes)
       (b) <RCV_IPADDR> IPV4 address of receiving interface (4 bytes)
       (c) <ACK_COLOR> Group Color (8 bits)
       (d) <ACK_SEQUENCE> object sequence# (8 bits)
```

4.2.5 <NEWCOLOR>

<NEWCOLOR> : <NEW_COLOR> <NEW_SEQUENCE>

This contains the information about a new COLOR and SEQUENCE for a multicast group. The membership list is done as an explicit <ACK_LIST> and is not handled here.

numMembers = (<OBJECT_LENGTH> - 2)/4

```
   15             8 7             0
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      (a)      |      (b)      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <NEW_COLOR> New group color (8 bits)

(b) <NEW_SEQUENCE> Next valid sequence# (8 bits)

4.2.6 <MRA>

The MRA Block format is:

```
<MRA>                 :   <MRSRID> <NUM_NBRS> <NUM_MPRFLAGWORDS>
                          <NBR List> <MPRFLAGWORDS List>

<NBR List>            :   <NBR List> <NBR Entry>
                      |   <NBR Entry>

<MPRFLAGWORDS List> :   <MPRFLAGWORDS List> <MPRFLAGWORD>
                      |   <MPRFLAGWORD>
```

<MRA> is defined as follows: This block contains the RID of the advertising MRS, followed by a counter indicating the number of neighbors and a counter indicating the number of words required to hold the MPR flags indicating which of those neighbors are MPRs. The MRA may contain any number (up to total length restrictions) of one-hop neighbor RIDs, and associated flags specifying which of these neighbors have been selected as MPRs.

```
31            24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (a)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
31            24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              (b)              |              (c)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <MRSRID> Router ID of advertising MRS (4 bytes)
(b) <NUM_NBRS> Number of one-hop neighbors (16 bits)
(c) <NUM_MPRFLAGWORDS> Number of 32-bit words required for MPRFLAGS (16 bits)

NUM_MPRFLAGWORDS = (NUM_NBRS+31)/32

```
31            24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (d)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(d) <NBR Entry> Neighbor Router ID (4 bytes)
One entry per neighbor.

```
31            24 23           16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              (e)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(e) <MPRFLAGWORD> 32-bit word containing 32 1-bit MPR flags
One word required for 32 neighbors.
The i-th bit in the j-th word indicates the MPR status of the n-th (n = j*32 + i) neighbor in the neighbor list where 1 indicates the neighbor is a MPR, and 0 indicates otherwise.

4.2.7 <BCAST>

A broadcast object block is used for delivering encapsulated data to an upper-layer protocol (ULP). This block will be received and passed to the appropriate ULP by all receivers. If the <BCAST> is sent reliably, then only those routers with a matching color may forward the message to the appropriate ULP. Each object block may be independently- sequenced by virtue of its object header. However, all blocks with reliability share the same group color.

<BCAST> : <PROTOCOL> <OBJLEN> <OBJDATA>

```
23            16 15            8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (a)      |              (b)              |       (c) ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <PROTOCOL> protocol type (8 bits)
    0   - reserved
    1   - TORA
    2   - AODV
    3-255 - unassigned

(b) <OBJLEN> block length (in bytes) (16 bits)

(c) <OBJDATA> This is <OBJLEN> bytes of data encapsulated by IMEP

<BCAST> blocks are delivered reliably, and can therefore have an explicit acknowledgement list. The <OBJLEN> in (b) can be subtracted from the <OBJECT_LENGTH> to determine the number of explicit addresses that should generate acknowledgments.

numExplicitAcks = (<OBJECT_LENGTH> - (<OBJLEN> + 3))/4

4.2.8 <MCAST>

A multicast (or explicit) object block is very similar to a broadcast object in that it is also used for delivering encapsulated data to an upper-layer protocol (ULP). The difference is that the <MCAST> contains an *explicit* delivery list. This implies that the object data block can be passed to the appropriate ULP only by receivers that are members of the <DELIVERY_LIST>. If the <MCAST> is sent reliably, then only those routers with a matching color may forward the message to the appropriate ULP. Each object block may be independently-sequenced by virtue of its object header. However, all blocks with reliability share the same group color. It should be noted that if this block is sent with reliability, then all receivers, not just those on the <DELIVERY_LIST>, must ACKnowledge receipt of the message.

<MCAST> : <PROTOCOL> <OBJLEN> <DELIVERY_LIST_LEN> <OBJDATA>

```
   23             16 15              8 7              0
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      (a)       |               (b)               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    15             8 7              0
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |               (c)              |     (d) ....
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

(a) <PROTOCOL> protocol type (8 bits)
    0   - reserved
    1   - TORA
    2   - AODV
    3   - DSR
    4   - ZRP
    5-255 - unassigned

(b) <OBJLEN> block length (in bytes) (16 bits)

(c) <DELIVERY_LIST_LEN> - Length of the explicit delivery list (in bytes). (16 bits)

(d) <OBJDATA> This is <OBJLEN> bytes of data encapsulated by IMEP

<MCAST> blocks may be delivered reliably, and can therefore have an explicit acknowledgement list. The <OBJLEN> in (b) and the <DELIVERY_LIST_LEN> in (c) can be subtracted from the from the <OBJECT_LENGTH> to determine the number of explicit addresses that should generate acknowledgments.

numExplicitAcks = (<OBJECT_LENGTH> - (<OBJLEN> + <DELIVERY_LIST_LEN> + 3))/4

4.2.9 <MR>

A multipoint relaying object block is also similar to a broadcast object in that it is also used for delivering encapsulated data to an upper-layer protocol (ULP). The difference is that the <MR> contains an implicit delivery list as determined by the MR algorithm. The object data block is only passed to the appropriate ULP the *first* time it is received at a router--any subsequently received copies are silently discarded. Routers maintain a list of recently-received <MR> blocks indexed by SMR and MRSEQUENCE to determine whether a block was previously received.

If the <MR> is sent reliably, then only those routers with a matching color may forward the object to the appropriate ULP. Each object block may be independently-sequenced by virtue of its object header. However, all blocks with reliability share the same group color. It should be noted that if this block is sent with reliability, then all receivers, not just the MPRs, must ACKnowledge receipt of the message.

<MR> : <SMRRID> <MRSEQUENCE> <OBJLEN> <OBJDATA>

```
 31             24 23             16 15             8 7              0
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                              (a)                              |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

31            24 23            16 15             8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (b)      |      (c)      |              (d)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

31            24 23            16 15             8 7             0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      (e)....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

(a) <SMRRID> protocol type (32 bits)
Router ID of Source of the Multipoint Relay packet.

(b) <MRSEQUENCE> Multipoint Relay packet sequence# (8 bits)

(c) <PROTOCOL> protocol type (8 bits)

```
0     - reserved
1     - TORA
2     - AODV
3     - DSR
4     - ZRP
5-255 - unassigned
```

(d) <OBJLEN> block length (in bytes) (16 bits)

(e) <OBJDATA> This is <OBJLEN> bytes of data encapsulated by IMEP

4.2.10 <ACK List>, <DELIVERY_LIST>

Lists are arrays of IPV4 addresses. Each entry is a 32-bit address in network byte order. The length of the list is either stored as part of the object information (see <DELIVERY_LIST_LEN>) or inferred from other available lengths (see <OBJECT_LENGTH> and <OBJLEN>).

4.2.11 <AUTH> (The IMEP Authentication object)

The IMEP Authentication object is used to authenticate all IMEP objects. The types of authentication to be supported will be specified in a proposed MANET Authentication Architecture under development.

4.3 ULP/IMEP Interface

Other than registration, IMEP interacts with ULPs in several fundamental ways. Here this interaction is specified in a format which loosely follows the Object Management Group's (OMG) Interface Definition Language (IDL).

4.3.1 Registration

ULPs must register with IMEP prior to use. Registration consists of calling the following register function.

```
typedef enum SignallingSupport { CONN, LINK, DISABLED };

void register (in <PROTOCOL> type,
                  // indicates Protocol type of data object
                  // if not valid, an InvalidProtocolType exception
                  // is thrown.
               in any ULPhandle,
                  // *implementation-dependent*
                  // a handle is passed to IMEP depending on the
                  // implementation of the ULP/IMEP system that allows
```

```
                     // IMEP to pass signals to the ULP.
                     // if not valid (and this is detectable by IMEP),
                     // an InvalidULPhandle exception is thrown.
                  in <OBJLEN> epitaphLength,
                     // indicates length of the epitaph object;
                     // if length = 0, this indicates no epitaph message and
                     // the OBJDATA field is ignored.
                     // if length > MAX_EPITAPH_LENGTH, then
                     // an InvalidByteLength exception is thrown
                  in <OBJDATA> epitaph,
                     // opaque epitaph data object
                  in SignallingSupport mode)
                     // indicates IMEP Signalling Support mode
                     // if incorrect, an InvalidSignallingSupport exception
                     // is thrown
                  raises (InvalidProtocolType,
                          InvalidULPhandle,
                          InvalidByteLength,
                          InvalidSignallingSupport);
```

4.3.2 Encapsulation

IMEP principally aggregates and encapsulates ULP objects into longer IMEP messages. From a ULP's perspective, these may be delivered reliably or unreliably, and either implicitly broadcast to the entire one-hop neighbor set, or explicitly multicast to a one-hop neighbor subset. Thus, an object being given to IMEP for transmission must come with this additional information. The following specifies the operation ``encapsulate".

```
typedef enum Boolean { TRUE, FALSE };
typedef enum ForwardingMode { BCAST, MCAST, MR };

void encapsulate (in <PROTOCOL> type,
                    // indicates Protocol type of data object
                    // if not valid, an InvalidProtocolType exception
                    // is thrown.
                 in <OBJLEN> length,
                    // indicates length of data object;
                    // if length > MAX_IMEP_LENGTH, then
                    // an InvalidByteLength exception is thrown
                 in <OBJDATA> data,
                    // data object to be transmitted
                 in ForwardingMode mode,
                    // indicates IMEP forwarding mode
                    // if incorrect, an InvalidForwardingMode exception
                    // is thrown
                 in <DELIVERY_LIST> list,
                    // List of IPv4 addresses to which object
                    // should be explicitly delivered via MCAST.
                    // If one or more addresses are incorrect,
                    // an InvalidInterface exception is thrown
                 in Boolean reliability)
                    // indicates whether reliable delivery is desired
                 raises (InvalidProtocolType,
                         InvalidByteLength,
                         InvalidForwardingMode,
                         InvalidInterface);
```

5. Security Considerations

The MANET computing environment is very different from the ordinary computing environment. In many cases, mobile computers will be connected to the network via wireless links. Such links are particularly vulnerable to passive eavesdropping, active replay attacks, and other active attacks. Among its many uses, the networking protocol described in this document enables inter-router communication for purposes of network control. This control function could be a

---

significant vulnerability if IMEP messages are not authenticated.

Authors' Addresses:

M. Scott Corson
Institute for Systems Research
A.V. Williams Building (115)
University of Maryland
College Park, MD 20742, USA
(301) 405-6630
corson@isr.umd.edu

S. Papademetriou
Institute for Systems Research
A.V. Williams Building (115)
University of Maryland
College Park, MD 20742, USA
(301) 405-7933
spyro@isr.umd.edu

Philip Papadopoulos
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6367, USA
(423) 241-3972
papadopoulpm@ornl.gov

Vincent Park
Information Technology Division
Code 5540
Naval Research Laboratory
Washington, DC 20375, USA
(202) 767-5098
vpark@itd.nrl.navy.mil

Amir Qayyum
INRIA
Sophia-Antipolis, France
Amir.Qayyum@inria.fr