INTERNET-DRAFT   Zygmunt J. Haas, Cornell University
                                  Marc R. Pearlman, Cornell University
                                  Prince Samar,   Cornell University

The Interzone Routing Protocol (IERP) for Ad Hoc Networks
<draft-ietf-manet-zone-ierp-02.txt>

Status of this Memo

Abstract

   This document describes the Interzone Routing Protocol (IERP), the
   reactive routing component of the Zone Routing Protocol (ZRP).
   IERP adapts existing reactive routing protocol implementations to
   take advantage of the known topology of each node's surrounding
   R-hop neighborhood (routing zone), provided by the Intrazone
   Routing Protocol (IARP). The availability of routing zone routes
   allows IERP to suppress route queries for local destinations.
   When a global route discovery is required, the routing zone based
   bordercast service can be used to efficiently guide route queries
   outward, rather than blindly relaying queries from neighbor to
   neighbor. Once a route has been discovered, IERP can use routing
   zones to automatically redirect data around failed links.
   Similarly, suboptimal route segments can be identified and traffic
   re-routed along shorter paths.

                            Contents

Applicability Statement

A. **Networking Context**

   The Interzone Routing Protocol (IERP) is the global reactive
   routing component of the Zone Routing Protocol (ZRP).   IERP
   adapts existing reactive routing protocol implementations to
   take advantage of the known topology of each node's surrounding
   R-hop neighborhood (routing zone), provided by the Intrazone
   Routing Protocol (IARP).   The availability of routing zone routes
   allows IERP to suppress route queries for local destinations.
   When a global route discovery is required, the routing zone based
   bordercast service [2] can be used to efficiently guide route
   queries outward, rather than blindly relaying queries from neighbor
   to neighbor.   Once a route has been discovered, IERP can use
   routing zones to automatically redirect data around failed links.
   Similarly, suboptimal route segments can be identified and traffic
   re-routed along shorter paths.

   The effectiveness of bordercasting and zone based route maintenance
   improves with increased routing zone radius.   However, an increased
   routing radius requires additional proactive traffic to maintain a
   current view of the larger routing zone.   Based on this tradeoff, it
   follows that networks characterized by a highly dynamic topology and/
   or low route demand favor smaller routing zones.   In extreme cases,
   the routing zone reduces to zero hops (or one hop, for multiple
   channel networks) and route discovery degenerates into traditional
   flood searching.   As the demand for new routes increases and/or the
   network topology stabilizes, larger routing zones become more
   appropriate.

B. **Protocol Characteristics and Mechanisms**

   * Does the protocol provide support for unidirectional links?
     (if so, how?)
        Yes.   The IERP can provide "local" support for unidirectional
        links, based on routing information provided by the IARP.
        A unidirectional link can be used as long as the link

source and link destination lie within each other's routing zone.

* Does the protocol require the use of tunneling? (if so, how?)
    No.

* Does the protocol require using some form of source routing?
  (if so, how?)

    No.

* Does the protocol require the use of periodic messaging?
  (if so, how?)

      No.

* Does the protocol require the use of reliable or sequenced packet
  delivery? (if so, how?)

      No.

* Does the protocol provide support for routing through a multi-
  technology routing fabric? (if so, how?)

      Yes. It is assumed that each node's network interface is
      assigned a unique IP address.

* Does the protocol provide support for multiple hosts per router?
  (if so, how?)

      Yes. Multiple hosts may be associated with a router. These
      hosts can be identified by the neighbor discovery/maintenance
      process.

      By default, it is assumed that a host's association with a router
      is not permanent. As a result, IERP exchanges routing
      information for individual hosts in the same manner as routing
      information for routers.

      In cases where a router is permanently associated with a network
      (subnetwork), IERP may support the exchange of network
      (subnetwork) prefixes in place of all aggregated host IP
      addresses.

* Does the protocol support the IP addressing architecture?
  (if so, how?)

      Yes. Each node is assumed to have a unique IP address (or set of
      unique IP addresses in the case of multiple interfaces). IERP
      references all nodes/interfaces by their IP address.

* Does the protocol require link or neighbor status sensing
  (if so, how?)

      No.

   * Does the protocol have dependence on a central entity?
     (if so, how?)

        No.   IERP is a fully distributed protocol.

   * Does the protocol function reactively? (if so, how?)

        Yes. A route query is initiated, on demand, when a node requires
        routing information that is not immediately available in its
        routing table. The route query propagates through the network,
        using a special packet delivery service called "bordercasting".
        Bordercasting leverages knowledge of local network topology to
        direct route queries away from the source, thereby reducing
        redundancy.

   * Does the protocol function proactively? (if so, how?)

        No.

   * Does the protocol provide loop-free routing? (if so, how?)

        Yes.   In this draft, loop-freedom in the reactive route discovery
        process is ensured by inspection of accumulated source routes.
        For distributed distance vector approaches, loop-freedom can be
        ensured by labeling queries (replies) with the source
        (destination) address and locally unique sequence number.
        Nodes that relay these messages can temporarily cache these
        identifiers in order to identify and terminate loops.

   * Does the protocol provide for sleep period operation? (if so, how?)

        No. Sleep period operation is not addressed in this draft.
        However, sleep period support can be included as needed.

   * Does the protocol provide some form of security? (if so, how?)

        No. It is assumed that security issues are adequately addressed
        by the underlying protocols (IPsec, for example).

   * Does the protocol provide support for utilizing multi-channel,
     link-layer technologies? (if so, how?)

        Yes. It is assumed that each node's network interface is
        assigned a unique IP address.

[1](#). **Introduction**

   The design of ad hoc network routing protocols is influenced by link
   instability (due to node mobility) and limitations in available
   bandwidth and transmission power. Traditional wired networks use
   proactive routing protocols, like OSPF [7] and RIP [16] to maintain
   up-to-date routes to all network nodes. More efficient proactive
   routing protocols have been developed for ad hoc networks [1][5][8]
   [9][15]. However, continuously tracking the frequent topology changes
   in a practical ad hoc network can still produce an overwhelming amount
   of control traffic. Even worse, most of the acquired route
   information expires before it is ever used, making the proactive
   control traffic a poor investment of bandwidth. In contrast, reactive
   routing protocols [6][10][14] only initiate a global, query-based,
   route discovery as routes are needed. While some delay is incurred in
   route acquisition, the amount of overhead traffic is generally much
   less than proactive routing protocols, because routing information is
   not wasted. For this reason, reactive protocols are generally viewed
   as being more suitable than proactive routing protocols for the power/
   bandwidth limited mobile ad hoc network.

   Although a global proactive routing protocol may overwhelm an ad hoc
   network's resources, a LIMITED SCOPE proactive routing protocol can be
   used to the advantage of a global reactive routing protocol. This
   hybrid routing approach forms the basis for the Zone Routing Protocol
   (ZRP) framework [4].

   The cost for each node to proactively track the topology of its
   surrounding R-hop neighborhood (routing zone) can be justified by
   improved route discovery efficiency and more effective route
   maintenance [11][12]. Routes to local destinations are immediately
   available, avoiding route discoveries. When the global route
   discovery is needed, the routing zones can be used to efficiently
   guide route queries outwards through bordercasting [2], rather than
   blindly relaying queries from neighbor to neighbor. The proactive
   maintenance of routing zones also helps improve the quality and
   survivability of discovered routes, by making them more robust to
   changes in network topology. Once routes have been discovered,
   routing zone offers enhanced, real-time, route maintenance. Link
   failures can be bypassed by multiple hop paths within the routing
   zone. Similarly, suboptimal route segments can be identified and

traffic re-routed along shorter paths.

Within the context of the ZRP, we refer to the globally reactive
routing component as the Interzone Routing Protocol (IERP). The IERP
is not a specific routing protocol. Rather, it is a family of
reactive routing protocols that offer enhanced route discovery and
route maintenance services based on local connectivity monitored by
the proactive Intrazone Routing Protocol (IARP) [3]. In this document,
we provide an example IERP implementation. In addition, we provide a
set of basic guidelines which can be used to convert an existing
reactive routing protocol to an IERP.

2. On Demand Route Discovery and the IntErzone Routing Protocol (IERP)

The reactive route discovery process consists of two phases: the route
request phase and the route reply phase. The route request phase is
initiated when a node requires a route to a destination, but does not
have the route stored in its route table. This query source issues a
route request packet and sends this packet to each of its neighbors.
When a node with an active route to the query destination receives the
request, it may respond with a reply. Otherwise, it forwards the
request packet to its neighbors. Subsequent copies of the route
request are considered to be redundant and are discarded.

When a queried node can provide a route to the destination, a reply
containing information about the discovered route is sent back to the
query source. In order to relay the reply, the request needs to
accumulate route information as it progresses through the network.
Before forwarding a query packet, a node appends its address and
relevant node/link metrics to the packet. When a query packet reaches
the destination, the sequence of recorded nodes represents a route
from the source to the destination. This route may be reversed and
used to send the reply back to the query source. Transmission
resources can be saved during the route request phase by distributing
previous hop information among the intermediate nodes, instead of
appending node addresses to increasingly longer packets. A similar
approach can be used during the reply phase. The query source may
receive an entire source route to the query destination, or each route

node can record the next-hop address to the destination in its routing
table.

A route request broadcast traverses all network links, allowing any
reachable destination to be discovered. However, the undirected
nature of broadcasting results in redundant coverage. Nodes are sent
copies of the same route request by each neighbor. An optimal probing
mechanism would direct the query outward, away from the query source
and away from regions that have already been covered by the query.

A hybrid routing protocol can provide directed query propagation by
exploiting knowledge of routing zone topology. When a node processes
a route request through its route cache, it is effectively responding
on behalf of its routing zone. As the routing zone is effectively
covered, the routing zone nodes no longer need to be explicitly
interrogated. Instead, the route request is "bordercast" to the
routing zone's peripheral nodes*, along multicast trees constructed
from the known routing zone topology. Redundant query coverage is
minimized by pruning tree branches ending in the routing zones of
previously queried nodes. Bordercasting and zone-based query control
are described in the Bordercast Resolution Protocol (BRP)
specification [2].

For a routing zone radius of one hop, bordercasting degenerates
into flood searching. Increasing the routing zone radius improves
route discovery efficiency. However, this requires additional
proactive traffic to maintain a current view of the larger routing
zone. The tradeoff between routing zone maintenance and route
discovery efficiency lies at the heart of the hybrid ZRP framework
[4].

* Peripheral nodes are those routing zone members that mark the edge
  of the routing zone. In the case of uniform zone radii, the
  peripheral nodes of an r-hop routing zone are those nodes which lie
  at a minimum distance of r hops.

3. Routing Zone Based Route Maintenance

After a route is acquired, knowledge of the local topology can be used

to bypass link failures and sub-optimal route segments. The resulting
increase in route lifetime and reduction in route length translates to
a more stable, lower latency, higher throughput network application.

When neighboring nodes in a route move out of direct radio contact,
the resulting link failure interrupts data flow across the route. For
a purely reactive routing protocol, any routes that include the broken
link immediately fail. To maintain end-to-end connectivity, a new
route discovery / repair would have to be initiated. Until a
replacement route or route segment is discovered, incoming data
packets are either delayed or dropped, degrading application
performance.

Because the routing zone provides a node with a view beyond its own
neighborhood, many link failures can be instantly bypassed. As long
as the former neighbor remains within the routing zone, incoming data
packets can be redirected around a broken link, through an active
multi-hop path to the former neighbor.

Just as a route's nodes may move apart, they may also move closer
together. This provides an opportunity for routes to be shortened.
When the identity of a route's downstream nodes is known, as in the
case of source routes, a node can check if any "shortcuts" exist
through its routing zone. Some degree of proactive route shortening
can be achieved simply by tracking neighbor connectivity (routing
zone of radius 1 hop). By examining the packet's source route, a
relay can determine the closest node to the destination that is also
a neighbor. In some cases, a multiple hop segment of the route can
be replaced by a single hop. Route shortening can be extended to
larger routing zones, providing earlier opportunities to redirect
traffic along a wider array of shorter alternate segments. Relaying
nodes make locally optimal decisions, selecting the path that reaches
the destination in the fewest hops.

4. IERP Conversion Guidelines

The following guidelines can be used to convert an existing
reactive routing protocol to an IERP, without compromising the
defining features of the base routing protocol.

- Any local proactive route updates and neighbor advertisements
  should be disabled, since this functionality is provided by IARP.

- IERP must support the importation of IARP routes into its own
         routing table and support route lookups into the IARP routing
         table.

       - Advanced route maintenance techniques such as on-line route
         repair and route shortening may be employed.

       - The broadcast() of ROUTE REQUEST packets should be replaced with
         with a call to the bordercast() service, as provided by the BRP.

       - Redundant query termination (i.e. flood control) mechanisms must
         be disabled. Redundant query control is handled by BRP.
         However, IERP may discard ROUTE REQUESTS based on other criteria,
         such as successful route discovery, exceeded QoS metrics,
         expired TTL (limited depth search), etc.

       - ROUTE REQUEST broadcast jitter must be disabled. This service
         is provided by BRP.

5. Implementation Example - Reactive Source Routing

   This example IERP demonstrates the integration of bordercast route
   discovery and routing zone based route maintenance in a source route
   reactive routing protocol. To highlight the role of the routing zone
   services, the implementation of the underlying reactive routing has
   been kept simple. More advanced features, such as diversity
   injection [13], expanding ring search, route metric collection, etc.
   are compatible with the routing zone services and may be added as
   desired.

   When a node has no valid route to forward a data packet, it
   launches a route discovery, probing the network via bordercast
   ROUTE_REQUST packets. When a node receives a ROUTE_REQUEST packet,
   it appends its IP address along with metrics for the link on which
   the packet was received. It then checks its Routing Tables for
   a valid route to the query destination. If no valid route is
   found, the node relays the ROUTE_REQUEST to the "downstream"
   neighbors identified by the bordercast service (provided by the
   Bordercast Resolution Protocol (BRP)). If a valid route to the
   query destination is known, then the route is appended to the

ROUTE_REQUEST's accumulated route. The complete route is copied
to a ROUTE_REPLY packet. The ROUTE_REPLY is forwarded back to
the query source, by IERP, along the reversed accumulated route.

IERP also leverages the known routing zone topology to support
local proactive route maintenance. When a node's IARP detects a
change in its routing zone connectivity, the IERP is notified and
proceeds to review the status of its routes. For each IERP route,
the node identifies an alternate path through its routing zone
that minimizes the distance to the destination. This serves to
bypass failed links and sub-optimal route segments. The updated
routes are then saved in the IERP routing table.

A. Packet Format
```
                         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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Type      |     Length    |   Node Ptr   |    RESERVED    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |           Query ID            |        R E S E R V E D        |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                  Query/Route Source Address                  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ -+-
 |                 Intermediate Node (1) Address                |  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
 |                 Intermediate Node (2) Address                |  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
                            | |                                    |
                            | |                               route
                           \| |/                                  |
```

```
                               \ /                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-| |
    |                Intermediate Node (N) Address            | |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
    |                Query/Route Destination Address          | |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ -+-
```

Field Description:

* Type   (char)   (8 bits)
        Identifies the type of IERP packet. The current version
        of IERP contains two packet types:

        ROUTE_REQUEST:
                Request for a route to the Query Destination.
                The ROUTE_REQUEST records the path that it
                has traveled from the Query Source.

        ROUTE_REPLY:
                Response to a ROUTE_REQUEST packet, issued by
                the node that discovers a route to the Query
                Destination, and sent back to the Query Source.

* Length   (char)   (8 bits)
        Length of the packet, in multiples of 32 bit words.

* Node Pointer   (char)   (8 bits)
        Index into the route (see below) corresponding to the
        node that has just received, or is next to receive,
        this packet.

* Query ID   (unsigned int)   (16 bits)
        Sequence number which, along with the Query Source Address
        (see below) uniquely identifies any ROUTE_REQUEST in the
        network.

* Query/Route Source Address   (node_id)   (32 bits)
        IP address of the node that initiates the ROUTE_REQUEST.
        In subsequent stages, this corresponds to the IP address
        of the discovered route's source node.

* Query/Route Destination Address   (node_id)   (32 bits)
        IP address to be located during the ROUTE_REQUEST

phase. In subsequent stages, this field contains the IP
address of the discovered route's destination node.

* Route   (node_id)   (N * 32 bits)
Variable length field that contains the recorded IP
addresses of nodes along the path traveled by this
ROUTE_REQUEST packet from the Query Source. After a
route to the Query Destination has been discovered,
this set of IP addresses provides a specification of
the route between the Route Source and Route Destination.

B.   Data Structures

B.1 IARP Routing Table   (See IARP Description [3])

B.2 IERP Routing Table

```
+-----------------------|-------------------------------+
|   Dest    |  Subnet   |      Route      |     Route     |
|   Addr    |   Mask    |                 |    Metrics    |
| (node_id) | (node_id) | (node_id list) | (metric list) |
|-----------+-----------|----------------+---------------|
|           |           |                 |               |
|-----------+-----------|----------------+---------------|
|           |           |                 |               |
|-----------+-----------|----------------+---------------|
|           |           |                 |               |
+-----------------------|-------------------------------+
```

C.   Interfaces

C.1 Higher Layer (N/A)

C.2 Lower Layer (BRP)
   C.2.1   Deliver(packet,BRP_cache_ID)
        Used by BRP to deliver packet to IERP.

```
        C.3  Lower Layer (IP)
           C.3.1   Deliver(packet)
                Used by IP to deliver packet to IERP.

        C.4  IARP
           C.4.1  IARP_updated()
                Notifies IERP that the routing zone connectivity
                has changed.

        C.5  ICMP
           C.5.1  Initiate_Route_Discovery(dest)
                Initiates route discovery for "dest" when no route
                to "dest" is available in the routing table.


   D.  State Machine

       This implementation of IERP consists of only one state (IDLE).
       Therefore, no state transitions need to be specified.  IERP
       immediately acts upon an event and then returns back to the
       IDLE state.

       Note:  1) X is used as a label for the node running this state
                 machine.

       D.1
           Event:   IARP reports an update to routing zone connectivity.

           Action:  For each route in X's IERP routing table, compute a
                    path to each downstream node (based on the IARP
                    routing table.)  Identify the computed path that
                    minimizes the route length, and update the IERP route
                    with this path.
```

```
       D.2
           Event:   A ROUTE_REQUEST packet is received with a destination
                    that appears within X's routing zone.
```

```
        Action:   The packet's accumulated route information (for the
                  source) is recorded in X's Routing Table and Temporary
                  Query Cache.  The accumulated route is replaced by
                  X's address and any accumulated route metrics are
                  updated and compressed.

                  X copies the ROUTE_REQUEST packet's contents to a
                  ROUTE_REPLY packet.  The ROUTE_REPLY packet is
                  returned to the query source, along the reversed
                  accumulated route.

     D.3
        Event:    A ROUTE_REQUEST packet is received with a destination
                  that DOES NOT appear within X's routing zone.

        Action:   X adds its address to the accumulated route and the
                  ROUTE_REQUEST packet is bordercast().

     D.4
        Event:    A ROUTE_REPLY packet is received.

        Action:   The packet's accumulated route information (for the
                  destination) is recorded in X's Routing Table.  X
                  adds its address to the accumulated route and adds
                  metrics for the downstream (toward the destination)
                  link to the accumulated metrics.  If X is not the
                  query source, then X forwards the message toward the
                  source (directly through IP), along a path selected
                  from the Temporary Query Cache (i.e. based on
                  Diversity Injection).

  E.  Pseudocode Implementation

     We define two complimentary operations on packets:
     extract(packet) and load(packet)

        extract(packet)
           extracts the fields of the IERP packet to the following
           variables: {type, length, node_ptr, query_id,
                       source, route, dest}

        load(packet)
           loads the values of the aforementioned variables into
           the fields of the IERP packet.
           load(packet) automatically computes the packet length
```

```
E.1 IARP_updated()

    // Perform route maintenance on each IERP route
    for each dest (BELONGING TO) IERP_Routing Table
    {
        for each route (BELONGING TO) IERP_Routing_Table[dest]
        {
            L         = length(route);
            min_dist  = INF;
            new_route = NULL;

            // Lookup the IARP path to each node in route.
            // Update the IERP route using the IARP path
            // that minimizes the distance to the
            // IERP route's destination.
            for j = 1:L
            {
                node = route(j);
                route_tail = route(j+1:L);
                IARP_path = IARP_Routing_Table(node).route;
                new_route = IARP_path + route_tail;
                if(hop_count(IARP_path + route_tail) < min_dist)
                {
                    new_route = IARP_path + route_tail;
                    min_dist  = hop_count(new_route);
                }
            }
        }
    }

    // Import IARP routes to IERP, so that all known routes are
    // centrally available
    import(IERP_Routing_Table, IARP_Routing_Table);


E.2  Initiate_Route_Discovery(dest)

     // ROUTE_REQUEST is initialized and sent down to BRP
     // to launch bordercast
     source    = MY_ID
     query_id  = MY_QUERY_ID++;
     type      = ROUTE_REQUEST;
     route     = NULL;
     node_ptr  = 0;

     load (packet);

     // Replaces traditional "broadcast()"
     bordercast(packet, NULL);
```

```
      E.3  Deliver(packet, BRP_cache_ID)

           extract(packet);
           switch(type)
           {
              case:  ROUTE_REQUEST
                  if ((EXISTS) IERP_Routing_Table[dest].route)
                  {
                      // Append discovered route to accumulated route
                      // and send reply back to the source.
                      route += IERP_Routing_Table[dest].route
                      type  = ROUTE_REPLY;

                      load(packet);
                      send(packet,next_hop,IP);
                  }
                  else
                  {
                      // If route to destination is not available, then
                      // append MY_ID to accumulated route and continue
                      // to forward ROUTE_REQUEST packet.
                      route += MY_ID;
                      node_ptr++;

                      load(packet);

                      // Replaces traditional "broadcast()"
                      bordercast(packet, BRP_cache_ID);
                  }
              break;

              case:   ROUTE_REPLY

                  // Extract route from packet and record it in the
                  // IERP Routing Table
                  route_tail = route(current_hop_ptr : length(route)):
                  add(IERP_Routing_Table[dest], route_tail);

                  // Forward ROUTE_REPLY until it reaches source
                  if(MY_ID != source)
                  {
                      node_ptr--;
                      next_hop = route(node_ptr);

                      load(packet);
                      send((packet,next_hop),IP);
```

```
                    }
                break;
            }
```

5. References

[1]   Garcia-Luna-Aceves, J.J. and Spohn, M., "Efficient Routing in
           Packet-Radio Networks Using Link-State Information,"
           WCNC '99, September 1999.

[2]   Haas, Z.J., Pearlman, M.R. and Samar, P., "Bordercast Resolution
           Protocol (BRP)," IETF Internet Draft,
           draft-ietf-manet-brp-02.txt, July 2002.

[3]   Haas, Z.J., Pearlman, M.R. and Samar, P., "Intrazone Routing
           Protocol (IARP)," IETF Internet Draft,
           draft-ietf-manet-iarp-02.txt, July 2002.

[4]   Haas, Z.J., Pearlman, M.R. and Samar, P., "Zone Routing Protocol
           (ZRP)," IETF Internet Draft, draft-ietf-manet-zrp-04.txt,
            July 2002.

[5]   Jacquet, P., Muhlethaler, P., Qayyum A., Laouiti A., Viennot L.,
           and Clausen T., "Optimized Link State Routing Protocol,"
           IETF Internet Draft, draft-ietf-manet-olsr-03.txt,
           November 2000.

[6]   Johnson, D.B. and Maltz, D.A., "Dynamic Source Routing in Ad-Hoc
           Wireless Networks," in Mobile Computing, edited by T.
           Imielinski and H. Korth, chapter 5, pp. 153-181,
           Kluwer, 1996.

[7]   Moy, J., "OSPF Version 2," INTERNET DRAFT STANDARD, RFC 2178,
           July 1997.

[8]   Murthy, S. and Garcia-Luna-Aceves, J.J., "An Efficient Routing
           Protocol for Wireless Networks," MONET, vol. 1, no. 2,
           pp. 183-197, October 1996.

[9]   Ogier, R. "Efficient Routing Protocols for Packet-Radio Networks
           Based on Tree Sharing," MoMUC '99, November 1999.

[10]   Park, V.D. and Corson, M.S., "A Highly Adaptive Distributed
           Routing Algorithm for Mobile Wireless Networks,"
           IEEE INFOCOM'97, Kobe, Japan, 1997.

[11] Pearlman, M.R. and Haas, Z.J., "Determining the Optimal
            Configuration for the Zone Routing Protocol," IEEE JSAC,
            August, 1999.

[12] Pearlman, M.R., Haas, Z.J. and S.I. Mir, "Using Routing Zones to
            Support Route Maintenance in Ad Hoc Networks,"
            IEEE WCNC 2000, Chicago, IL, Sept. 2000.

[13] Pearlman, M.R., Haas, Z.J., "Improving the Performance of Query-
            Based Routing Protocols Through Diversity Injection,"
            IEEE WCNC 1999, New Orleans, LA, Sept. 1999.

[14] Perkins, C.E. and Royer, E.M., "Ad Hoc On-Demand Distance
            Vector Routing," IEEE WMCSA'99, New Orleans, LA, Feb. 1999

[15] Perkins, C.E. and Bhagwat, P., "Highly Dynamic
            Destination-Sequenced Distance-Vector Routing (DSDV) for
            Mobile Computers," ACM SIGCOMM, vol.24, no.4, October 1994.

[16] Postel, J., "Internet Protocol," RFC 791, Sept. 1981.

Authors' Information

   Zygmunt J. Haas
   Wireless Networks Laboratory
   323 Frank Rhodes Hall
   School of Electrical & Computer Engineering
   Cornell University
   Ithaca, NY 14853
   United States of America   tel: (607) 255-3454, fax: (607) 255-9072
   Email: haas@ece.cornell.edu

   Marc R. Pearlman
   389 Frank Rhodes Hall
   School of Electrical & Computer Engineering
   Cornell University
   Ithaca, NY 14853
   United States of America
   tel: (607) 255-0236, fax: (607) 255-9072
   Email: pearlman@ece.cornell.edu

   Prince Samar
   374 Frank Rhodes Hall
   School of Electrical & Computer Engineering
   Cornell University
   Ithaca, NY 14853
   United States of America
   tel: (607) 255-9068, fax: (607) 255-9072
   Email: samar@ece.cornell.edu

 The MANET Working Group contacted through its chairs:

   M. Scott Corson
   Institute for Systems Research

University of Maryland   College Park, MD 20742
(301) 405-6630
corson@isr.umd.edu

Joseph Macker
Information Technology Division
Naval Research Laboratory
Washington, DC 20375
(202) 767-2001
macker@itd.nrl.navy.mil