# ARP Considered Harmful: Manycast Transactions in Ad Hoc Networks

Casey Carter Seung Yi Robin Kravets
Department of Computer Science
University of Illinois, Urbana-Champaign.
{ccarter,seungyi,rhk}@uiuc.edu

***Abstract*—ARP handles neighbor discovery and address resolution in infrastructure networks, but is inadequate for Mobile Ad Hoc Networks (MANETs). Thus, many MANET routing protocols include a neighbor discovery mechanism. This separation of neighbor discovery and address resolution is a fundamental design problem that causes packet loss, particularly when the communication is manycast, a novel variant of multicast communication. Our approach, automatic address resolution, moves address resolution into the routing protocol along with neighbor discovery, correcting these problems.**

## I. Introduction

Mobile Ad Hoc Networks (MANET)s have more dynamic topologies than traditional infrastructure networks. Consequently, MANET routing protocols require that neighbor discovery information be kept as up-to-date as possible. In infrastructure networks, neighbor discovery and address resolution are integrated into a single protocol, the Address Resolution Protocol (ARP) [1], at the interface between the network and link layers. MANET's different requirements have encouraged routing protocols to incorporate a neighbor discovery mechanism stronger than that present in ARP. The integration of neighbor discovery into routing has created a separation between neighbor discovery and address resolution, which may cause the neighbor set and the set of resolved neighbor addresses to lose synchronization. This loss of synchronization can result in unnecessary packet loss and address resolution overhead that can easily be eliminated.

In this paper, we detail the interaction between MANET routing protocols, traditional ARP implementations, and what we term manycast transactions. A *manycast* transaction is one in which a single client sends a request to many servers, all of which respond to that client. Traditional implementations of ARP, although well-suited to infrastructure network operation, cripple manycast transactions in a MANET.

Our contribution is to illustrate the degenerative effects that arise from the separation of neighbor discovery and address resolution. We also present a scheme to address this problem by strongly coupling address resolution and neighbor discovery, regardless of the particular layer in the network stack that performs these functions. This coupled approach avoids the problems that can arise when the two are separate, and specifically addresses manycast.

This work was supported by National Science Foundation grant number ANI–0081308.

In Section II, we present manycast and ARP and discuss how their implementations interact in MANETs. Section III presents our solution to the separation of neighbor discovery and address resolution, automatic address resolution. Simulation results presented in Section IV verify the problem and validate our approach. In Section V, we reiterate the key points of the paper, draw conclusions, and present some directions for future work.

## II. Motivation and Discussion

In this section, we describe manycast and give several examples of manycast communication in ad hoc networks. We also detail the operation of ARP and MANET routing protocols, as well as how they are commonly implemented in isolation. With these concepts in mind, we present the *manycast conflict*: an interaction between manycast, ARP and MANET routing in which multiple responses to a manycast request concentrate into a small region around the originator.

### A. Manycast

The lack of infrastructure and frequency of topology changes in MANETs has given birth to a new style of interaction: manycast. In a manycast transaction, a single source performs an RPC with many destinations in the MANET. Manycast is appropriate for applications in ad hoc networks in which a single client requests service from many servers whose locations are not known *a priori*, and when the service requires only best-effort reliability. The exact set of destinations is identified by a multicast-like address. The routing protocol propagates this request throughout the MANET in a best-effort fashion, and nodes that believe themselves to be in the destination set perform the RPC and reply to the sender. There is no guarantee of reliability, or even that *any* eligible destinations exist in the MANET.

Manycast is used to support a variety of MANET services. For example, the authors' work on MOCA seeks to develop a distributed certificate authority to deploy public key infrastructure in wireless ad hoc networks [2]. The key component of the MOCA framework is a set of distinguished mobile nodes, the MObile Certificate Authorities (MOCA), which collectively act as the certificate authority for the MANET. A mobile node that wishes certification service manycasts its certification request to the MOCAs. The system secret is distributed among

the MOCAs using threshold cryptography, which enables a mobile node to get certification service by contacting any $k$ of the $n$ MOCAs. Manycast has also been used to provide efficient Internet gateway detection for MANETs [3], [4], [5]. We believe that integration of future services into MANET with similar requirements for distribution and robustness against unreliability will only increase the number of users of manycast.

### B. ARP Design

Although its name is Address Resolution Protocol, ARP is effectively a single protocol that simultaneously performs address resolution and neighbor discovery in multiple-access networks. The brevity of the ARP protocol specification [1] (10 pages) is a testament to the elegant simplicity of ARP's design. The longevity of the specification (20 years) attests to ARP's suitability to the address resolution problem. ARP incorporates two distinct goals into a single protocol: *neighbor discovery*, determining that a given network peer is (is not) a neighbor, and *address resolution*, establishing the mapping between link layer addresses and network layer addresses.

*1) Neighbor Discovery:* Neighbor discovery (ND) is the primitive from which multihop routing is built. To route data from one end of a network to the other, it is necessary that each node along the route is aware of the existence of the next hop. We separately characterize two flavors of ND information: positive and negative. Positive ND provides indications that a neighbor is present, or that a new neighbor has come within range of communication. Negative ND indicates that a previous neighbor has left the neighborhood, e.g., by leaving the communication range or crashing. Different mechanisms provide varying flavors of ND: for example, overhearing any transmission from a neighbor is a positive ND indication, but the absence of such transmissions is not necessarily a negative ND indication.

*2) Address Resolution:* As an *inter*-networking protocol, IP must be independent of any specific choice of link layer technology [6]. Consequently, a mechanism that maps IP addresses to link-layer addresses is necessary to transmit datagrams between IP peers attached to the same multiple-access link (neighbors). We refer to this problem as address resolution (AR).

### C. ARP Operation

From a pure protocol view, ARP is simple. Consider a node A (with link-layer address A') which has a datagram it needs to communicate to node B (with link-layer address B'). Node A broadcasts a query packet on the link that contains its own address mapping and requests the link-layer address of node B:

| Request | | | |
|---|---|---|---|
| SPA | SHA | DPA | DHA |
| A | A' | B | ? |

All nodes on the link receive this request, but only node B – whose address is in the Destination Protocol Address field – sends a unicast response packet with its link-layer address to node A:

| Reply | | | |
|---|---|---|---|
| SPA | SHA | DPA | DHA |
| B | B' | A | A' |

In this way, the ARP protocol provides for AR and positive ND. With this single exchange of a pair of packets, both nodes learn that the other node is a neighbor, as well as discovering each other's link-layer addresses. The crux of the problem lies in how this pure protocol specification is integrated into a real network stack.

The system's perspective on ARP is much messier for two reasons. First, systems often desire negative ND as well as positive. Negative ND is useful in many situations:

- Mobile nodes in wireless networks may move out of transmission range.
- IP-to-link-layer mappings may change over time, e.g., when an network interface card is changed.
- A neighbor can sustain a sudden failure.

IETF host requirements specify that ARP implementations must provide a mechanism to flush out-of-date cache entries [7]. This mechanism is usually a timer causing entries to age and be removed from the ARP cache, necessitating a fresh resolution. In this way, ARP implementations usually provide a limited form of negative ND.

Second, ARP resolution is usually performed on-demand, as packets for a neighbor arrive from the application layer. Many ARP implementations derived from the original BSD Unix ARP buffer only a single packet for each neighbor while AR is ongoing. Additional packets that arrive for this same neighbor will replace the buffer contents until the AR process completes. Collisions in this single-packet buffer becomes a problem in ad hoc networks, since ad hoc nodes tend to communicate through a greater number of neighbors than infrastructure network nodes do.

### D. MANET Routing

The goal of MANET routing protocols is to determine a multihop route through an ad hoc network between a source and destination that will live long enough to deliver data packets between the two. These protocols can establish routes either proactively, maintaining routes to all possible destinations, or reactively, establishing routes only in response to data demand.

Although infrastructure-based routing protocols traditionally delegate ND to ARP, the dynamicity of MANETs requires ad hoc protocols to maintain finer-grained control of ND. For our purposes, we characterize protocols by the proactivity/reactivity with which they perform this discovery. Protocols can perform ND proactively by continuously maintaining awareness of the neighborhood (ABR [8], CEDAR [9], NSR [10], OLSR [11], TBRPF [12], TORA [13], ZRP [14]) or reactively by discovering neighbors as a side effect of on-demand route discovery (AODV [15], DSR [16]). Some protocols (NSR, ZRP) perform proactive ND, while also discovering routes reactively. Since the two processes have differing goals, proactive ND can complement reactive routing.

We believe that the source of the conflict between ARP and MANET routing is rooted in the fact that both ARP and MANET routing protocols perform ND simultaneously, but independently. Many routing protocols incorporate the concept of link-layer notification made popular by the IEEE 802.11 specification [17]: the link layer can notify the protocol of failure to

deliver a packet to an assumed neighbor. By our classification, this link-layer notification is a negative ND indication. Unfortunately, this application of link-layer notification constitutes a violation of proper layering: the network-layer routing protocol is aware of a ND indication while link-layer ARP is not.

Even in the absence of this layering/design violation, link-layer ND performed by ARP is simply redundant in MANET. As specified, most MANET routing protocols are already performing ND, and doing a better job of it than ARP. Even a purely reactive routing protocol like DSR discovers neighbors via a RREQ/RREP sequence before attempting to unicast packets to/via that neighbor. If MANET routing protocols can cheaply provide AR service, then the service provided by ARP can be eliminated from the system entirely.

### *E. Manycast Conflict*

The ideal implementation of manycast in MANET routing protocols of course differs between proactive and reactive routing protocols. Under a proactive protocol, manycast is routed just as multicast traffic: the manycast set membership is disseminated proactively along with the regular routing updates. Reactive routing results in much more interesting implementation detail. The naive approach floods the RPC request throughout the network, efficiently locating the destination set, but devastates the network by performing a flood to discover the return route for each reply. Efficient manycast under reactive protocols requires establishing the return routes as the request is propagated, so that a manycast RPC results in at most one flood of the MANET. Replies can then be sent unicast on the return routes so created.

The manycast conflict occurs when replies are propagated back to the manycast origin. Requests are sent using broadcast flooding (for reactive protocols), or at best multicast (for proactive protocols), but in neither case are ARP mappings established. The replies are likely to be synchronized, resulting in a reverse broadcast storm [18] as replies and ARP broadcasts "implode" into the origin. There are two separate effects that contribute to the manycast conflict: the queuing effect, and the reverse broadcast storm.

The queuing effect occurs when replies collide in the single-packet ARP queue. As manycast replies approach the origin, they are concentrated into an ever smaller region around the destination. Since each return hop may require ARP to resolve the next hop's address, it is likely that multiple responses will collide in the ARP buffer while awaiting AR (see Fig. 1). In this way, ARP simply discards many of the reply packets out of hand.

The reverse broadcast storm is due to a "storm front" of synchronized broadcast ARP requests converging on the origin (as in Fig. 2). In wireless links – at least in 802.11 – broadcasts are more likely to collide than unicasts. As the storm front converges on the origin, the medium may become congested; contention for medium access with the reply packets themselves further contributes to congestion. The medium may become so congested that ARP request packets collide or are dropped due to timeouts waiting for medium access, resulting in data packets that lie idly awaiting an AR that will never complete.

When these two effects combine, we have the manycast conflict: many replies are propagated through the network, which concentrate into a small region near the origin where high MAC contention results in loss. Consequently, a small proportion of manycast replies will be delivered to the origin, while resulting in a disproportionately large network load: replies are dropped only after crossing much of the network.

## III. Automatic Address Resolution

The logical solution to the separation of ND and AR is to ensure that the two remain tightly coupled in the network stack. This approach will eliminate redundant ND, as well as the manycast conflict. By the very nature of MANET routing, packets cannot be routed through a neighbor before an exchange of control packets allows that neighbor to be discovered. If AR is tied to that discovery, then the manycast conflict is easily sidestepped: no ARP-broadcast-implosion can occur, and manycast responses will never see the ARP packet queue, let alone collide within it. We propose to achieve this reintegration by incorporating AR into the MANET routing protocol.

Although previous work on Topology Broadcast based on Reverse Path Forwarding (TBRPF) has provided mechanisms to support automatic AR (Ref. [12], Section 10.3 "Optional Automatic Address Resolution"), we advocate that such support is not only possible but in fact necessary. We also extend AR support to other protocols, but retain TBRPF's term *Automatic Address Resolution*. In proactive protocols, like TBRPF, it is both straightforward and efficient to perform AR and ND simultaneously with proactive information exchange. As a routing

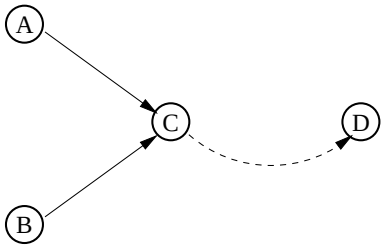


Fig. 1. Queuing Effect: manycast replies returning from A and B to D collide in the queue at C while ARP is resolving D's address; one reply is dropped.

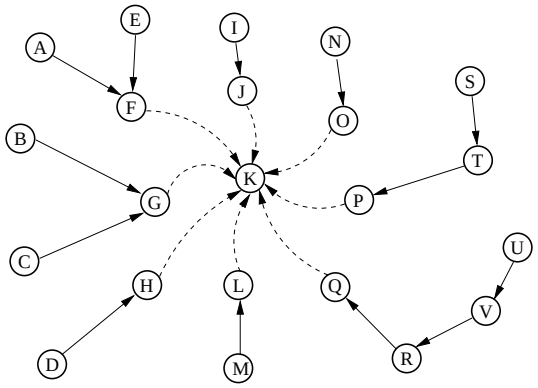


Fig. 2. Reverse Broadcast Storm Effect: many replies converge around the origin K, resulting in high medium contention and possible packet loss.

protocol receives proactive updates from its neighbors, it can easily maintain ARP cache entries for those neighbors simply by recording the originating link-layer address for each update packet. With reactive protocols, the approach is slightly more complicated.

Reactive protocols perform ND on-demand as route request/reply packets are propagated through the network, or (negatively) on a timeout or in response to link-layer notification of packet delivery failure. Reactive protocols can also perform automatic AR provided that, whenever a route request/reply packet is received, it is possible to determine the network-layer address of the previous hop of that packet. We assume that the link-layer source address of received packets is available to the routing protocol.

In a source-routed protocol such as DSR, automatic AR is trivial: the previous hop's network address is clearly indicated in the source route attached to a request/reply. We claim that any on-demand routing protocol that propagates requests hop-by-hop and propagates replies backward along the discovered route can also perform automatic AR. Note that a) hop-by-hop propagation implies that any received route request packet has the previous hop's network-layer address in its IP source address field, and b) a route reply must, at minimum, indicate the IP address of the next hop along the route, which is necessarily the previous hop transmitter of the reply packet.

Provided the network and link layer addresses of the neighbor relaying a route reply, the routing protocol can maintain ARP cache entries in parallel with route discovery. Whenever the routing protocol receives a request or reply it updates the ARP cache entry for that neighbor, ensuring that such an entry is present *before* the discovered route (and consequently, the neighbor) is used to forward packets.

Whether proactive or reactive, handling negative ND – such as when an update timer expires, or when some threshold number of periodic updates are missed, or link-layer notification provides an indication – is similar. When the routing protocol decides that a previous neighbor has moved away, it can delete that neighbor's ARP cache entry to maintain freshness of the cache.

The key concept of this approach is to ensure that the neighbor knowledge (represented by the set of routes with a particular neighbor as next hop) and AR information share fate. This ensures that a neighbor's link-layer address is already known when unicast communication with that neighbor becomes necessary.

We note that no additional communication resources are used for automatic AR: all of the necessary AR information is already available to the routing protocol, we simply advocate that the routing protocol should take maximum advantage of that data. Not only does automatic AR require no additional overhead, it actually reduces overhead by eliminating the exchange of ARP packets.

In general, automatic AR is applicable to any routing protocol that discovers the existence of neighbors (and their IP addresses) by receiving packets from those neighbors. Although this rules out anything as simple as statically configured routes in infrastructure networks, it does include the vast majority of dynamic routing protocols, particularly those MANET routing protocols considered to be proactive, reactive, and hybrids thereof. Any such protocol agent is easily augmented to perform automatic AR, simply by recording the link-layer address when a neighbor discovery occurs.

## IV. Evaluation

We present the results of several sets of simulation runs that illustrate the problems of separate ND/AR, and show how automatic AR resolves these issues. Specifically, we show how manycast transactions in MOCA are affected by the incorporation of automatic AR into an on-demand routing protocol, as well as briefly discussing the reduction in overhead achieved thereby. Other results, not presented here for lack of space, show how the manycast conflict impacts connectivity in a scheme that extends Mobile IP support into MANETs. Automatic AR is effective in preventing ARP conflicts for that application.

All simulations use the NS-2 simulator version 2.1b8a [19] to simulate a MANET with 150 mobile nodes, 30 of which are MOCAs, in a 1000m square area. We chose AODV as representative of the on-demand routing protocols. We evaluate three different forms of AR:

1) (ARP) AODV and the original ARP code included in NS-2, which is an approximation to BSD ARP.
2) (ARP-q) AODV and an ARP with greater queuing capacity (90 packets) [20]. The scalability of AODV-q is sensitive to the choice of queue length. We have deliberately chosen an unrealistically long queue length to illustrate that queuing alone is insufficient to address the entire problem.
3) (AODV-AR) AODV with automatic AR and no ARP.

### A. Long-Term Experiment

The long-term experiment is a ten minute simulation, during which 100 non-MOCAs each make one certification request per minute, for a total of 1000 certification requests. In an ideal setting with a fully connected network, each certification request should be answered with exactly 30 replies, totaling 30000 replies for the entire run. The results achieved for each AR type are reported in Table I.

With normal ARP, we note that only 72.6% of the manycast replies managed to reach the origin. The improvement of 19.1% for ARP-q, that comes from merely extending the queue, illustrates the impact of the queuing effect on manycast. This simple solution would seem effective, since the additional impact of the reverse broadcast storm effect is very small in the long term: only 0.5% here. We remind the reader that ARP-q achieves these gains by dedicating 90 times the buffer space of

TABLE I
Long-Term Simulation Results: replies received

| AR Type | ARP | ARP-q | AODV-AR |
|---|---|---|---|
| # of Replies | 21783 | 27498 | 27660 |
| % of Replies | 72.6% | 91.7% | 92.2% |

traditional ARP, while AODV-AR uses no ARP buffer space at all.

Intuitively, most of the ARP traffic in a simulation run happens at the beginning, when no node has neighbor knowledge. During this period, the reverse broadcast storm effect is noticeable. After the initial period of activity, caching ensures that the only additional ARP traffic is due to mobile nodes occasionally moving near new neighbors. Since this type of activity is non-localized, extending the ARP queue should suffice to address this level of activity. Long-term experiments should thus be queue-effect dominated, and we must inspect smaller timescales to see the true impact of the reverse broadcast storm effect.

### *B. Short-Term Experiment*

To see the detailed effect of the reverse broadcast storm effect on manycast, we evaluate a single certification request experiment. In a network with layout similar to that of the long-term experiment, a single node manycasts a single certification request. All MOCAs in the network should receive the certification request and send a reply. This scenario is repeated for 50 randomly chosen nodes over 10 different random topologies with each AR type. Fig. 3 shows a histogram detailing, for each number between 0 and 30 replies, the number of runs that result in that number of replies. Note that the endpoints of the data series for ARP and AODV-AR intersect at 27 replies; the reader should be careful not to confuse the two lines for one continuous line crossing the entire graph.

These results show that both the reverse broadcast storm and the queuing effect equally affect short-term behavior. The ARP results are widely spread around the mode of 20, indicating that around 1/3 of the replies are generally lost with wide variance. ARP-q, with its 90-packet queue, manages to improve upon ARP's results with a mode of 26. ARP-q also shows less variance. AODV-AR achieves the best results: its mode is 30, the ideal, with no results below 27.

Although these short term effects are negligible in the long-term experiment, one can postulate realistic scenarios in which short-term behavior dominates. Consider a sensor network consisting of many nodes distributed over a large geographic area. A single control node is mobile in that area, and performs periodic manycast transactions to retrieve data from the sensors with a very long period. If the period of data retrieval is longer than the timeout used to age ARP entries, each manycast will see a fresh network and exhibit exactly the behavior seen in our short-term experiment. If the mobility level of a MANET is great enough that most nodes change neighbors in between manycast transactions, the same effect occurs.

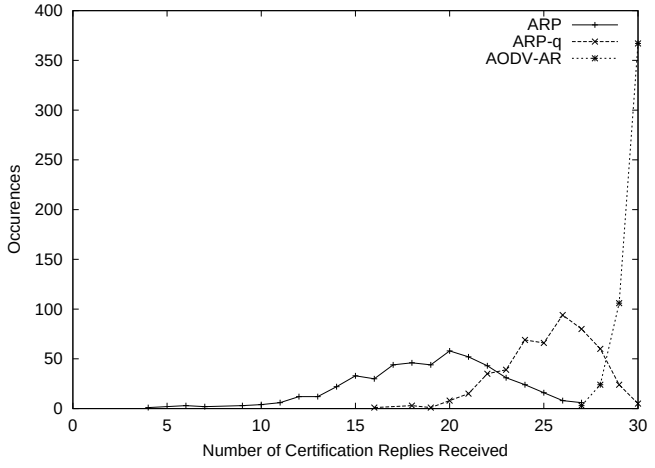


Fig. 3. Short-Term Simulation Results: Number of tests with $n$ replies

## V. Conclusion

The neighbor discovery service provided by ARP is not adequate for MANETs. The approach of current MANET routing protocols to this problem is to incorporate a more advanced ND mechanism, which is separate from address resolution in ARP. Analysis and simulated results support our conjecture that this separation of ND and AR contributes to packet loss in MANETs. Our proposed solution, automatic AR, entails shifting the AR component of ARP into the MANET routing protocol where it can again be tightly coupled to ND.

Although the study of manycast is novel to this paper, we are not the first to encounter problems with ARP in a MANET. Holland and Vaidya, in their study of TCP performance over MANET [21], note great packet loss due to an interaction between stale MANET routes and the ARP queue. The authors of a performance comparison of MANET routing protocols [22] noted that they had to modify the implementations of on-demand routing protocols to avoid the ARP queuing effect; otherwise, on route establishment, the routing protocol might immediately pass several packets to ARP that would instantly collide in the queue. We note that both of these problems, although they could be alleviated somewhat by using a longer queue, are easily solved by automatic AR.

One clear drawback to automatic AR is that it requires the routing protocol to understand the link layer address format, which is also a layering violation. If the link layer is "smart," as in Bluetooth [23], it may already be performing some variety of ND/AR that could be made available to the routing protocol. Future work will address this design issue by developing a stronger link-layer/network-layer interface than that provided by ARP. This stronger interface will support the needs of MANET routing protocols, enabling the removal of ND from routing.

Automatic AR is easily implemented into a MANET routing protocol, simply requiring the protocol to record neighbors' link-layer addresses upon receiving control packets. Automatic AR resolves address in MANETs at no cost in communication overhead over that already present in the routing protocol itself.

## References

[1] D.C. Plummer, "Ethernet address resolution protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware," Request for Comments (Standard) RFC 826, Internet Engineering Task Force, November 1982.

[2] Seung Yi and Robin Kravets, "Key management for heterogeneous ad hoc wireless networks," Tech. Rep. UIUCDCS-R-2002-2290, UIUC Department of Computer Science, July 2002.

[3] Prashant Ratanchandani and Robin Kravets, "A hybrid approach for internet connectivity for mobile ad hoc networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[4] Yuan Sun, Elizabeth M. Belding-Royer, and Charles E. Perkins, "Internet connectivity for ad hoc mobile networks," *International Journal of Wireless Information Networks, Special Issue on Mobile Ad hoc Networks*, 2002.

[5] Ryuji Wakikawa, Jari T. Malinen, Charles E. Perkins, Anders Nilsson, and Antti J. Tuominen, "Global connectivity for IPv6 mobile ad hoc networks," Internet Draft (Work in Progress) `draft-wakikawa-manet-globalv6-01.txt`, Internet Engineering Task Force, July 2002.

[6] J. Postel, "Internet protocol," Request for Comments (Standard) RFC 791, Internet Engineering Task Force, September 1981.

[7] R. Braden, "Requirements for internet hosts – communication layers," Request for Comments (Standard) RFC 1122, Internet Engineering Task Force, October 1989.

[8] C.-K. Toh, "Associativity-based routing for ad-hoc mobile networks," *Wireless Personal Communications Journal, Special Issue on Mobile Networking and Computing Systems*, vol. 4, 1997.

[9] P. Sinha, R. Sivakumar, and V. Bharghavan, "CEDAR: a core-extraction distributed ad hoc routing algorithm," in *IEEE Infocom '99*, 1999.

[10] Marcelo Spohn and J. J. Garcia-Luna-Aceves, "Neighborhood aware source routing," in *Proc. of the ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHOC '01)*, October 2001.

[11] Thomas Clausen, Phillippe Jacquet, Anis Laouiti, Pascale Minet, Paul Muhlethaler, Amir Qayyum, and Laurent Viennot, "Optimized link state routing protocol," Internet Draft (Work in Progress) `draft-ietf-manet-olsr-05.txt`, Internet Engineering Task Force, October 2001.

[12] Richard G. Ogier, Fred L. Templin, Bhargav Bellur, and Mark G. Lewis, "Topology broadcast based on reverse-path forwarding (TBRPF)," Internet Draft (Work in Progress) `draft-ietf-manet-tbrpf-05.txt`, Internet Engineering Task Force, March 2002.

[13] Vincent D. Park and M. Scott Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *INFOCOM '97*, 1997, vol. 3, pp. 1405–1413.

[14] Zygmunt Haas and Marc Perlman, "The zone routing protocol for ad hoc networks," Internet Draft (Work in Progress) `draft-ietf-manet-zrp-03.txt`, Internet Engineering Task Force, March 2000.

[15] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir R. Das, "Ad hoc on-demand distance vector (AODV) routing," Internet Draft (Work in Progress) `draft-ietf-manet-aodv-11.txt`, Internet Engineering Task Force, June 2002.

[16] David B. Johnson, David A. Maltz, and Yih-Chun Hu, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," Internet Draft (Work in Progress) `draft-ietf-manet-dsr-07.txt`, Internet Engineering Task Force, February 2002.

[17] IEEE Computer Society, "802.11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," June 1997.

[18] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu, "The broadcast problem in a mobile ad hoc network," in *MOBICOM '99*, 1999, pp. 151–162.

[19] "Network simulator 2," http://www.isi.edu/nsnam/ns/.

[20] Marcel Odena, "ARP with queue hack," March 2002, Posting to ns-2 users mailing list (ns-users).

[21] Gavin Holland and Nitin Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *ACM Mobicom '99*, 1999.

[22] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *ACM Mobicom '98*, 1998.

[23] The Bluetooth SIG, "Specification of the bluetooth system, version 1.1," February 2001.