Implementing Scalable Position-Based Multicast for the Linux Kernel*

Matthias Transier, Holger Füßler, Thomas Butter, and Wolfgang Effelsberg

{transier,fuessler,effelsberg}@informatik.uni-mannheim.de, tbutter@tbutter.de

1 Introduction

Although interest in mobile ad-hoc networks has vigorously grown over the last years, research in this area still strongly relies on discrete event simulation. However, recently published work challenges the value of simulations [PJL02] and strongly argues for real-world experiments [TLN03]. In this context, the number of real-world implementations for ad-hoc routing protocols continues to grow.

The improving miniaturization and availability of positioning systems allows the deployment of position-based routing algorithms [MWH01] in mobile ad-hoc networks.

This paper presents a real-world implementation of a position-based multicast routing protocol called SPBM, or "Scalable Position-Based Multicast" [TFW+04]. This is—to the best of our knowledge—the first implementation of an algorithm of this class. The long list of projects implementing protocols of different classes is omitted here due to space restrictions.

The contribution of this work is a brief description of the SPBM Linux kernel module and the challenges that had to be faced during the realization. The sources of our implementation are available on our web site [SPB]. In the following we give an overview of the protocol (Section 2) and report on its implementation as a Linux kernel module (Section 3). We further demonstrate the necessary integration with a mobile GPS system and give an example evaluation scenario using Linux-based HP iPaq hand-held devices (Section 4).

2 The Protocol

In the following, we will briefly introduce the "Scalable Position-Based Multicast" routing protocol. A more detailed description can be found in [TFW+04].

*This work was partly supported by the Deutsche Forschungsgesellschaft (DFG: German Research Foundation) as part of the priority program (SPP) 1140 ("Basissoftware für selbstorganisierende Infrastrukturen für vernetzte mobile Systeme").

The protocol can be divided into two main parts: *group membership management* and *multicast forwarding*. Each of these is based on a quad-tree structure, which is used to aggregate membership information by region, thereby allowing multicast forwarding to proceed in a hierarchical fashion. To use the protocol in a mobile ad-hoc network, each of the participating nodes has to be aware of the geographic dimensions of the network area. The quad-tree is then organized as follows: The entire network is divided into four equally-sized squares, each of which is again divided in turn into four sub-squares. This process continues until the diameter of the smallest squares is small enough to enable all nodes within the same square to communicate with one another directly. The subsequent splittings generate different "grid levels" and the number of splittings is referred to as "grid depth". Figure 1 shows an example for a quad-tree with a grid depth of 3.

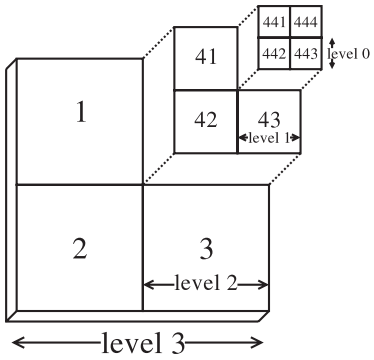

Figure 1: Quad-tree structure of the network

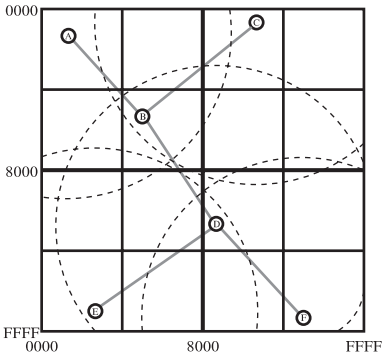

Figure 2: Evaluation setup with 6 devices

Multicast group membership information, i.e., the set of multicast groups in which a node is a member, is managed by means of bit vectors containing one bit per group (1 = subscribed, 0 = not subscribed). Each node periodically sends its vector to the nodes within its own lowest-level square. The disjunction of all vectors of one square constitutes the combined membership information for that square. It is propagated to the next higher level by sending one packet to all the nodes within the square on that level. The frequency of these propagation messages decreases with the level of aggregation, resulting in a very good scalability w.r.t. the size of the network.

Multicast forwarding is then done using the information collected by group management packets overheard by each node. When forwarding a packet, the forwarder stores the information about the remaining receiver nodes and/or squares inside the packet header. At the source, this is the information about all known group members, and at every consecutive hop, the information about the set of remaining receivers. For the exact next-hop selection mechanism, please refer to [TFW+04]. Roughly speaking, the forwarding and splitting process uses position-based routing towards node and square positions, relying on the fact that information grows more detailed as a packet approaches the final receivers.

Consequently, an implementation of SPBM has to comprise the following elements: First, it has to receive and send group membership update packets while managing the information about current memberships of adjacent nodes and squares; second, it must provide

interfaces for incoming and outgoing data packets, as well as for data packets which have to be relayed; third, the protocol has to be controllable by the applications in regard to multicast group memberships; and finally, it has to implement a facility for obtaining the node's own position from a positioning device.

3 Implementation

3.1 Fundamental Design Decisions

When implementing a routing protocol in Linux, the first main decision is whether to implement the protocol in kernel or in user space. While user space programs are much easier to debug, every routed network packet has to be transferred out of the kernel into user-accessible memory and back again, leading to higher additional latencies and thereby decreasing the usability of the protocol. For this reason, the SPBM protocol was implemented as a Linux kernel module, acknowledging the potential difficulties in the development process.

The goal of our implementation is to enable UDP communication between a sender and a multicast group. In terms of network layers, the module resides on layer 2.5, i.e., above the MAC protocol but below the IP layer. Consequently, the SPBM packet header is located after the MAC and before the IP header.

An important requirement for the easy deployment of a kernel module is its compatibility with a non-modified standard kernel. In our case, this has implications for the use of IP addresses. Standard IP multicast addresses (ranged from 224.0.0.0 through 239.255.255.255) are handled by the multicast routing implementation already included in the kernel. This cannot be circumvented without modifying the original kernel sources. Thus, we defined the valid address range for SPBM multicast groups as 10.255.0.0 through 10.255.255.255. A drawback to this approach is that applications on the host must use unicast rather than multicast sockets [SFR04] to send and receive multicast traffic. This forces one to employ a different means of initiating multicast communication that had been decided upon as the `/proc` interface, being a standard method of the kernel to provide easy-to-use kernel-to-user-space communication.

3.2 Packet Handling

The basic mode of operation is as follows: upon initialization the kernel module initialized, it registers a new MAC packet type number for the SPBM protocol. Incoming packets comprising this type number are then automatically delivered to a handler provided by the module. This handler function is responsible for the further processing of the incoming packets by classifying them into control and data packets, changing the local state appropriately and forwarding the packet if required.

Packets generated at the local host are filtered via the netfilter interface at the hook NF_IP_LOCAL_OUT (see [CP02]) and—if they are addressed to an SPBM address—captured by the routing module.

Outgoing packets are directly sent to the wireless network device. In preparation, a buffer space is filled with an Ethernet header and the packet data. The Ethernet header contains the protocol type number of SPBM, the node's own MAC address as the source and the next hop's MAC address as the destination.

While these techniques are common and likely to be used in implementations of new protocols for the Linux kernel, packets coming from the network and addressed to the local host are handled in a way unique to SPBM. This is because of the fact that we want to provide multicast communication via unicast sockets. The module deals with this issue by converting the received packets into standard unicast packets and handing them over to the IP stack. For this purpose, a new buffer space is allocated, MAC and IP headers are created and the actual packet is appended. The IP stack itself is then able to deliver the packet to the application.

The group management part of the protocol requires the sending of packets at a certain time. Thus, a time-based event scheduler is necessary, which consists of a priority queue that holds the events and a kernel thread that processes them at the defined time. Separate threads yield two major advantages: First, an implementation bug occurring in an own kernel thread does not bring down the whole system. Second, it enables the routing module to serialize all function calls through the scheduler. Incoming packets and packets waiting to be sent generate an event rather than being processed immediately. This provides a simple locking mechanism, allowing for conflict-free reading and writing of the SPBM data structures. Basically, the event list contains entries that consist of a time denoting the arrival of an event, a pointer to a function to be executed at event arrival, and a pointer to a socket buffer that possibly contains the packet to be processed.

3.3 Configuration Interface

As described above, the /proc interface is used to configure the SPBM module. To enable applications to communicate with the kernel, /proc basically provides a virtual file system. Applications can read or write files in this part of the file system tree to communicate with the kernel.

When an application wants to initiate multicast communication it writes the desired group ID to /proc/spbm/join telling the SPBM kernel module to initiate a group join. A corresponding call can be made to /proc/spbm/leave. Reading from /proc/spbm/join gives a list of the currently subscribed groups. When communication is initiated by joining a group, a datagram socket to the corresponding IP address henceforth supplies the application with the multicast packets received from the group. Sending to this socket implies sending to the multicast group.

Another task of the configuration interface is to provide an easy way to inform the kernel about the current geographical position. To accomplish this, writing to

`/proc/spbm/position` communicates the current position to the kernel, while reading from this file returns the current position of the node as perceived by the kernel. Since the kernel itself does not provide trigonometric functions, the position format used on this interface (and in all protocol operations) complies to a two-dimensional Cartesian coordinate system using 16 Bit per dimension as opposed to the floating-point geocentric angular coordinate format provided by most of the currently available GPS systems. The process of reading coordinates from a positioning service, converting them to protocol coordinates and feeding them to the module has to be accomplished by a daemon specific to the positioning service used. It is evident that each participating node has to be configured with the geographic dimensions of the network in order to participate in multicast communication.

The usage of 16-bit integer coordinates introduces some position inaccuracy. Depending on the size of the network that is mapped to these coordinates, the maximum positioning error is $\sqrt{2} \cdot \frac{s}{2^{16} \cdot 2}$, which is about four centimeters for a network size of 2000m by 2000m, and thus, far smaller than the inaccuracy of GPS. Of course, handling networks of a much greater size could require position encoding with greater precision.

4 Deployment Example

To validate the correct operation of our implementation, we carried out simple tests with with a simple setup of six nodes. In the experiment the nodes were "virtually" located as depicted in Figure 2. In order to enable reproducible experiments, the nodes were physically positioned directly next to each other, with the topology being enforced by filtering packets from nodes with a virtual position beyond the transmission range, as depicted by circles in Figure 2. This set-up causes an increase in the congestion level of the network since all nodes are in one another's interference range.

During each experiment we transmitted packets from node *A* to a multicast group that was joined by all other nodes. The sending rate of node *A* was limited only by the rate accepted by the MAC of node *A*, the size of the data payload was set to 1000 bytes, IEEE802.11 was set to 11 MBit/s, thus about 2.2 MBit/s gross for each link in Figure 2. We carried out the experiment 10 times. As a result, all nodes *B* through *F*, which were iPaq 3660 devices, received on average data at the rate of 408 kBit/s, while no packet loss occurred. The latter was to be expected since there was no node mobility and all transmissions of data packets were performed using unicast and MAC-level retransmissions. It is assumed that the bottleneck in these experiments is the CPU power of the iPaq hand-held devices. This assumption has to be further investigated by means of extensive performance analyses which are planned for the near future.

5 Conclusions and Future Work

In this paper we have presented the first Linux kernel implementation of SPBM, a position-based multicast routing protocol. We have outlined how the protocol implementation module fits into the kernel architecture and how it communicates with the user space and the standard networking stack. Furthermore, we have shown basic protocol design compo-

nents like the selected addressing / geographical positioning scheme and provided a deployment example. For the future, we plan extensive measurements and field-tests with multicast applications.

References

[CP02] Crowcroft, J. und Phillips, I.: *TCP/IP and Linux Protocol Implementation*. John Wiley & Sons. 2002.

[MWH01] Mauve, M., Widmer, J., und Hartenstein, H.: A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network*. 15(6):30–39. November/December 2001.

[PJL02] Pawlikowski, K., Jeong, H.-D. J., und Lee, J.-S. R.: On Credibility of Simulation Studies of Telecommunications Networks. *IEEE Communications Magazine*. 40(1):132–139. January 2002.

[SFR04] Stevens, W. R., Fenner, B., und Rudoff, A. M.: *UNIX Network Programming*. volume 1. Addison-Wesley. 3rd. 2004.

[SPB] SPBM Implementation Home Page. http://www.informatik.uni-mannheim.de/informatik/pi4/projects/pbm/kernel.html.

[TFW+04] Transier, M., Füßler, H., Widmer, J., Mauve, M., und Effelsberg, W.: A Hierarchical Approach to Position-Based Multicast for Mobile Ad-hoc Networks. Technical Report TR-04-002. Department for Mathematics and Computer Science, University of Mannheim. 2004.

[TLN03] Tschudin, C., Lundgren, H., und Nordström, E.: Embedding MANETs in the Real World. In: *Proc. of PWC '03*. S. 578–589. Venice, Italy. September 2003.