# Implementation and Validation of a Multicast Routing Protocol in an Ad Hoc Network Testbed

Shiyi Wu, Christian Bonnet

Department of Mobile Communication
Institut Eurécom Sophia-Antipolis, France
{Shiyi.Wu, Christian.Bonnet}@eurecom.fr

Yukihiro Takatani, Masato Hayashi

Corporate Technology Group, Hitachi Europe SAS
Valbonne, France
{yukihiro.takatani, masato.hayashi}@hitachi-eu.com

***Abstract*— We present our experiences in implementing and validating the Multicast Routing protocol with Dynamic Core (MRDC) in a real wireless ad hoc network testbed. The testbed consists of portable PCs and PDAs to reduce hardware cost and facilitate mobility test. The architecture of MRDC has been designed and introduced into the user space of Linux operating system without modifying the kernel. MRDC creates and maintains a group shared tree for each multicast session on demand. Besides the routing functionality, we designed a forwarding module in this implementation to forward all kinds of multicast datagram including MBone traffic. Integrating a simplified IGMP and a multicast tree information collector into the implementation, we evaluated the bandwidth utilization of MRDC in a stationary network scenario and the correctness of MRDC in both topology dynamic and membership dynamic scenarios in testbed experiments.**

***Index Terms*— Mobile ad hoc networks, multicast routing, MRDC, testbed.**

## I. Introduction

A Mobile Ad-hoc NETwork (MANET) is a collection of wireless mobile nodes forming a dynamical temporary network without the use of any existing network infrastructure or centralized administration. Each node in the network acts as a router and forwards packets on behalf of other nodes, which permits nodes communicate with each other beyond direct wireless coverage range. In MANET, multicast communication is an important issue since it can efficiently support a variety of group-oriented applications such as virtual classroom, remote conference and distributed games.

The characteristics of MANETs such as frequent topology change, limited bandwidth and node resources make conventional multicast routing protocol unsuitable. That is why many protocols have been proposed during last few years to support multicast routing in MANETs (MAODV [1], AMRIS [2], ODMRP [3], etc.). Most of these multicast protocols are only studied with software simulations. Software simulation is an excellent choice for initial design and an estimation of results. But the limitations of software simulator, such as lack of realistically duplication of physical layer, make hardware testing essential [4]. Due to high cost of hardware and difficulty of development, there exist few implementations of multicast routing protocol for ad hoc network exist for the real environment [5], [6].

We scope on Multicast Routing protocol with Dynamic Core (MRDC), which is one of on-demand tree-based multicast routing protocols proposed by Shiyi Wu, et al [7]. Implementation and evaluation of MRDC in the network simulator have been performed in their work, but the implementation in the real environment has not been done yet. One advantage of MRDC is its independence of any unicast routing protocol. Implement a unicast routing protocol for MANET in testbed requires extra effort and complicates result analysis.

In this paper we present the implementation of MRDC in a heterogeneous testbed which comprises portable personal computers (portable PC) and PDAs. Because PDAs are cheaper and lighter than portable PCs, this network configuration reduces hardware cost and at the same time facilitates mobility testing. Diverse components introduce extra difficulty in program developing, installation and validation. Thus, unlike [5] and [6] which modified kernel, the MRDC implementation is supposed to run in the user space and a multicast forwarding module is developed to realize on-demand fashion and multicast datagram forwarding in user space. With additional function modules such as a simplified

EURECOM research is partially supported by its industrial partners: ASCOM, Swisscom, FranceTelecom, La Fondation Cegetel, BouyguesTelecom, Thales, STMicroeletronics, Hitachi Europe and Texas Instruments.

IGMP [8] module and a tree information collection module, we evaluated the bandwidth utilization of MRDC and tested the correctness of the implementation in both topology dynamic scenarios and membership dynamic scenarios. We will describe these work in more detail.

The rest of the paper is organized as follows, MRDC is briefly overview in Section II followed by the protocol implementation description in Section III. Section IV introduce the performance evaluation of the implementation in a stationary network scenario and the correctness of MRDC in both topology dynamic and membership dynamic scenarios. And concluding remarks are made in Section V.

## II. MRDC Overview

This section gives a short overview of the Multicast Routing protocol with Dynamic Core (MRDC). For a detailed operation of the protocol, readers are referred to [7].

MRDC is an on-demand, group-shared, tree-based multicast routing protocol. It is independent of any unicast routing protocol. The first source of a multicast session becomes the core of this group, and triggers multicast tree creation. This strategy reduces routing overhead for multicast tree maintenance, improves data transmission efficiency, and is appreciated by MANETs. Tree structure is complex and difficult to be maintained in a frequent topology change network. MRDC solves this problem through local tree recovery and periodical multicast tree refreshing. Local tree recovery tries to repair broken tree branch in several hop away. On the other hand, periodical multicast tree refreshing destroys old tree and constructs a new tree to remove accumulated errors and adapt to topology changes. If core node has no further multicast traffic, it stops periodical tree refreshing procedure. The multicast tree is automatically erased after a timeout. The source, which wins the core competition, becomes core and is engaged in multicast tree creation and period refreshment. In this way, core immigrates to another sources.

The control part of MRDC contains two phases: Tree construction and Tree maintenance. Tree construction phase begins when a core is selected. Core is the first active multicast tree member and broadcasts a Core Advertisement message (CA message) to the rest of the network. Upon receiving a CA message, node becomes potential tree member and registers the node from which comes the message to construct a reverse path to the core. Group receivers initiate RAR/RAA procedure to join the multicast tree when receiving CA message. In this procedure, a Route Active Request message (RAR message) is unicast along the reverse path toward core. The first active tree members which receives the RAR message replies a Route Active Acknowledgment message (RAA message). RAA message activates the tree membership of nodes along the path to the original of the RAR message. Thus, a branch is added into multicast tree. Tree maintenance phase repairs tree branch broken due to topology changes and maintains a correct tree structure. This phase has two procedures: local tree recovery and periodical multicast tree refreshing. In local tree recovery procedure, the upstream node monitors the connectivity of a tree branch. Once a broken branch is detected, the upstream node broadcasts a Join Invite message (JI message) to inform the correspond downstream node and establish a reverse path. Then the downstream node starts a process similar to RAR/RAA procedure to re-join the tree. In periodical multicast tree refreshing procedure, core periodically broadcasts CA message to delete old tree and invite the other group members join the new multicast tree.

## III. Implementation

### A. Implementation Platform

*1) Hardware:* Ad hoc network nodes are Intel Pentium III based Dell C600 laptops and Intel StrongARM based compaq iPAQ H3850s equipped with IEEE802.11 wireless network card.

*2) Operating System:* MRDC was developed on Linux kernel version 2.4.18 provided by Red Hat 7.3. All tools and software packages that we used in our development originate from software bundle incorporated within the Red Hat Linux version 7.3 operating system package. We chose the Linux operating system for its availability and familiarity.

The PDAs used Familiar Linux v0.7 package with kernel version 2.4.19-rmk6-pxal-hh13 as operating system. It is necessary to install packet capture and packet filtering modules on PDA since these packages are not available in the installation package bundle. We used arm-linux-gcc from tool-chain to make cross platform compilation on red hat 7.3 platform

### B. Software Architecture

In this step, MRDC is designed to run in user space to simplify installation and test in both portable PCs and PDAs. The implementation architecture of MRDC is shown in Figure 1. Besides a multicast routing table, MRDC contains a group list which records all multicast

groups of which this node is a receiver and/or source. Because this implementation is aimed to demonstrate how to support multicast applications, other than MRDC core module, IGMP module and tree monitoring module are introduced. MRDC core module is further divided into two parts: Routing Part (RP) which, as describe in [7], constructs and maintains multicast tree on demand and updates multicast routing table, and Multicast Forwarding Part (MFP) which forwards multicast datagram according to the multicast routing table. IGMP module performances as simplified router to host part of Internet Group Management Protocol (IGMP) version 2 [8]. It detect membership changes on the local host and modifies group list. Tree monitoring module is an additional functionality. It collects multicast routing information of a pre-defined group from multicast routing table in each multicast tree member. Then a tree monitor written in JAVA runs in a special machine replays the multicast tree structure based on these informations. MRDC opens three sockets for inside and outside communication. IGMP module owns an IGMP socket (named igmp_socket) to send and receive igmp packets. MRDC core module opens a UDP socket (called udp_socket) for inter-node message exchange and a raw socket (denoted as raw_socket) to deliver multicast datagram to local host. Tree monitoring module shares udp_socket for multicast routing information collection. This architecture forms a complete multicast detection, creation, delivery and tree monitoring flow for demonstration. The rest of this section explains these modules in detail.

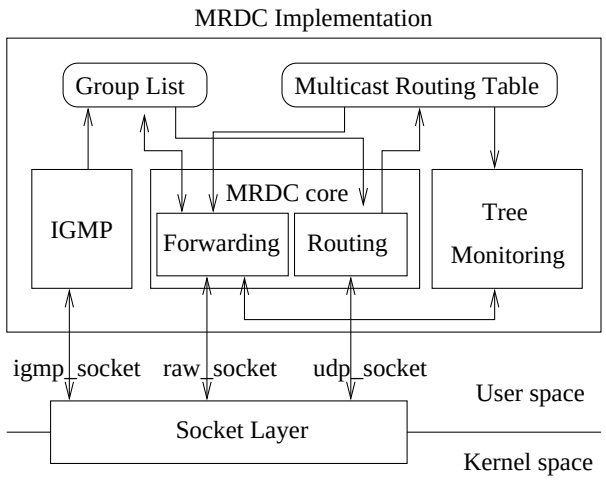


Fig. 1
MRDC IMPLEMENTATION STRUCTURE

*1) Routing Part:* A multicast tree creation starts when multicast forwarding part sniffs a packet addressed to a multicast group which does not exist in multicast routing table.

In Routing Part (RP), MRDC creates and maintains multicast trees on demand. A multicast tree is created when the first sender sends the first multicast packet. (We elaborate on how to detect the first sender when we explain multicast forwarding part.) This node becomes the core, and routing part starts to create a multicast tree. The routing information is stored in multicast routing table. After tree construction, multicast forwarding part begins to deliver datagram. Nodes can join or leave the multicast group at any time during the session. The core maintains the multicast session by refreshing the tree periodically, but if the source status is timeout which means no multicast datagram is sent during a period of time, the core stops this maintenance and a multicast tree is automatically erased by deleting routing information from multicast routing table. In addition, this implementation supports core immigration. Each multicast source supposes itself is core and begins to broadcast its own CA message when multicast tree is erased. When receiving multiple CA messages addressing same multicast group, nodes compare core IP address and choose the biggest one. The source which has the biggest IP address thus becomes core comptition winner and continue to send its CA message periodically. While the other sources stop sending their CA message and join multicast tree as a normal group member.

*2) Multicast Forwarding Part:* We met several difficulties during MRDC implementation. i) How to detect multicast datagram. MRDC cannot get multicast datagram directly since it runs in user space. ii) How to forward multicast datagram. In table-driven unicast case, routing agent can run in user space and modifies routing table in kernel. IP forwarding module in kernel forwards unicast datagram according to the routing table. MRDC cannot use the same approach because multicasting in MANETs and multicasting in fixed Internet is different. The routing table's multicast tree states consist of local interfaces instead of neighbor identities and , the verification for incoming data is also done on incoming interface rather the sender. However, one MANET node can use the same interface talking to any neighbor on the same wireless channel. The incoming physical interface verification done by the IP kernel is no longer applicable. iii) How to detect duplications. Duplication cannot be avoided since nodes use the same wireless channel to communicate.

In Multicast Forwarding Part (MFP), we introduce the combination of packet capture, packet encapsula-

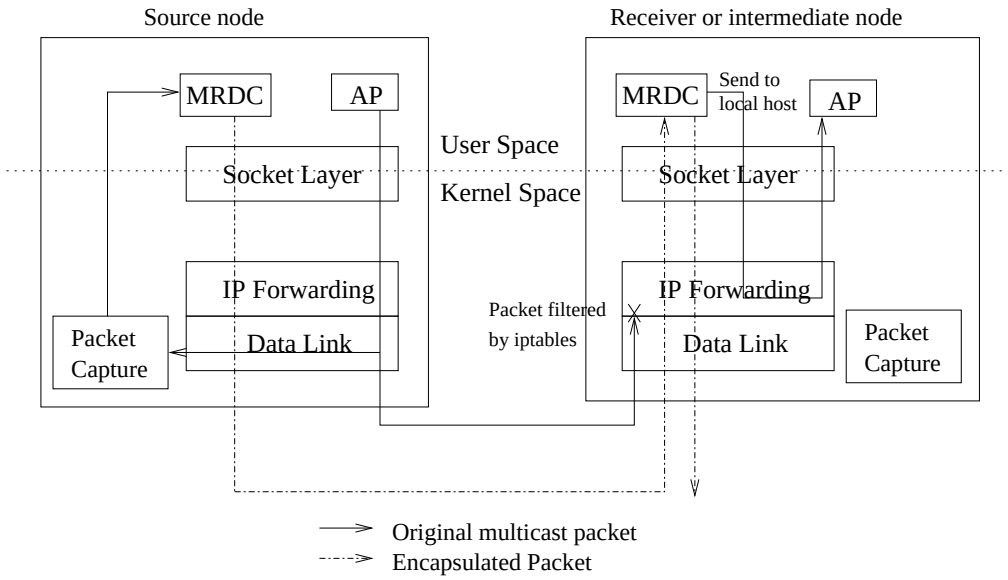


Fig. 2
MULTICAST PACKET FORWARDING IN MRDC IMPLEMENTATION

tion/decapsulation and packet filtering, to solve the above problems. Figure 2 illustrates this procedure. A multicast datagram sent by an application is sniffed by packet capture at data link layer. Packet capture passes the packet to MRDC. MRDC encapsulates the captured packet into a MRDC data packet and then broadcasts it to neighbors through upd_socket. Nodes relay MRDC data packet till group receivers according to MRDC's multicast routing table. At last, MRDC in the side of receiver extracts the datagram from MRDC data packet and sends it to local application through raw_socket. To detect duplication, MRDC uses the informations stored in MRDC data packet header. On the other hand, if a group receiver is in coverage region of a group source, it receives multicast datagram directly from source. In order to avoid this phenomenon, the packet filtering (iptables installed in Linux) is applied to each node. We will explain in details how they work.

The basic technique used to capture multicast traffic packets in user space is the same as Unix tcpdump [9]. Since all machines use Linux as operating system, we use libpcap facility. It listens traffic at data link layer and sniff packets which are in accordance with a pre-defined rule set. For example, in this implementation, we want to capture all multicast datagram sent by a node itself. If the ip address of node A is 192.168.25.197, the following rule is set:
*(tcp or udp) and ip multicast and src host 192.168.25.197*
This rule captures all tcp and upd multicast packets sent by node A. Then libpcap facility passes the captured packets to MFP as a raw packet. MFP checks the destination address in the ip header of captured packet by consulting multicast routing table. If the destination address concerns a new multicast group, MFP sets up group source state in group list, caches the packet and informs routing part to create multicast tree. Otherwise, if the destination address concerns an existing group, MFP refreshes group source state and encapsulates the captured multicast datagram into a MRDC data packet and broadcasts it through udp_socket. A MRDC data packet header, as show in Figure 3, has following five fields. A sequence number is assigned by the source in order to detect packet duplication.

When a multicast tree member receives a MRDC data packet, MFP first detects whether the same packet is received before and drops all duplication. If it is not the case, MFP decreases ttl by one and broadcasts the MRDC data packet to its neighbors. At the same time if the group list indicates that local host is a receiver of the corresponding multicast group, MFP decapsulates the MRDC data packet and transfers the multicast datagram through raw_socket.

An important point that we should take into account is the broadcast characteristic of wireless link. An application may receive the same multicast datagram

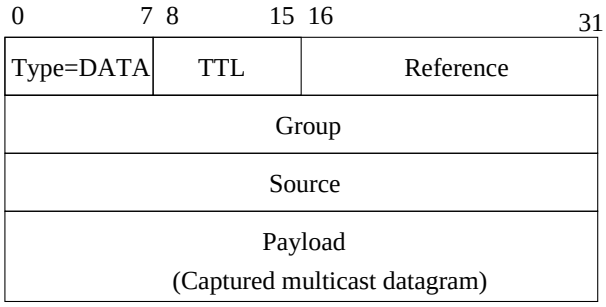


Fig. 3
MRDC PACKET STRUCTURE

which has been already received if the node is in the coverage range of the source. Therefore it is necessary to prepare the packet duplication avoidance mechanism in the implementation. To filter the packet from source,we use the Netfilter/iptables facility [10]. It sits in-between the kernel IP stack and network device drivers and manipulates every packet in or out of this host according to pre-defined rules. Rules can be set or changed at any time through a command interface. For example, if the wireless interface is eth0, the following rule is set:
*iptables -A INPUT -d 224.0.0.0/16 -p udp -i eth0*
This rule drops all udp multicast packets coming from eth0.

According to this architecture, multicast packets are delivered properly.

*3) IGMP Module:* IGMP Module periodically sends igmp membership query message through igmp_socket and listens at igmp_socket. It sets up group receiver state in group list if receiving a membership report message. On the contrary, it unsets group receiver state if capturing a igmp leave group message.

On the other hand, group list periodically unsets the states which are not updated during last period.

*4) Tree Monitoring Module:* The node whose IP address is coincidence with the predefined monitor address is topology monitor. Monitor consulting the multicast routing table to check whether a multicast tree exists for the pre-defined group. If it is the case, monitor broadcasts a message to the rest of the network. This message is made up of monitored group, monitor address, sequence number and last hop fields, When this message propagates in the network, a reverse path to the monitor is constructed. All members of the corresponding multicast tree sends the information including the IP address of their upstream and downstreams to the monitor through this reverse path. This procedure is executed periodically.

*5) Timers:* We selected five seconds for multicast tree refresh interval. IGMP queries membership every eight seconds query and membership timeout was set to eighteen seconds. Tree monitoring collects tree information every second.

## IV. PERFORMANCE EVALUATION

We created a six node testbed for our multicast experiments. We study the bandwidth utilization of MRDC in a stationary network scenario and verify the correctness of the MRDC in topology and membership dynamic scenarios. During the evaluation, all WaveLan devices operated on 2.4 GHz bandwidth and communicated at the capacity of 2 Mb/s with transmission power of 1mW. The WaveLan devices were operated in an ad hoc mode.

### *A. Stationary Network Scenario and Results*

The experimental network setting is shown in Figure 4. This topology is similar to [6]. Our network consisted of six nodes among which three are portable PCs (nodes A, B and C) and other three are PDAs (nodes D, E and F). All nodes can hear each other in the MAC layer. A virtual wall is constructed in the network layer via iptables. For example, a filter is set in node A to drop all packets coming from nodes D, E and F. A topology monitoring program developed by Hitachi ran in Monitor to display the multicast tree based on the informations collected by tree monitoring module. In this experiment, a file is multicast from node A to the receivers E and F. Figure 5 illustrates two multicast tree structures which were shown by topology monitor during the experiment. That is because MRDC chooses the first discovered path. If node F receives first new CA message from node E, a one branch multicast tree, tree_1, is constructed. Otherwise, a two branches tree, tree_2, is formed. Table I shows the measurement results. The total throughput is far below the full WaveLan data rate of 2 M/s. There are three reasons. The first, it is due to network layer multi-hop forwarding while nodes were physically placed together. Multicast source and relayor shared the same wireless channel. The second reason is we did not prevent the original multicast traffic to be injected into the wireless channel. The third one is that two alternative multicast tree contain different number of interior nodes. Tree_1 has three interior nodes while Tree_2 contains four interior nodes. In the former tree, the bandwidth is divided by four (one original traffic and three forward traffic) and in the later tree, the bandwidth is divided by five. In this MRDC implementation, the MRDC overhead comes mostly from multicast packet encapsulation while

routing messages such as CA, RAR and RAA messages can be ignored. IGMP control overhead can be ignored. However, tree monitoring overhead is high because we chose a small tree information collection interval. This small interval permit us to observe tree changes in next experiments.

In this implementation, MRDC operates in user space. If we succeed in moving MRDC to kernel, high costly kernel-to-user crossing for store-and-forward packets can be avoided, the overhead caused by encapsulation can be greatly reduced and original traffic will not be injected into network. We believe consequently the data throughput can be significantly improved.

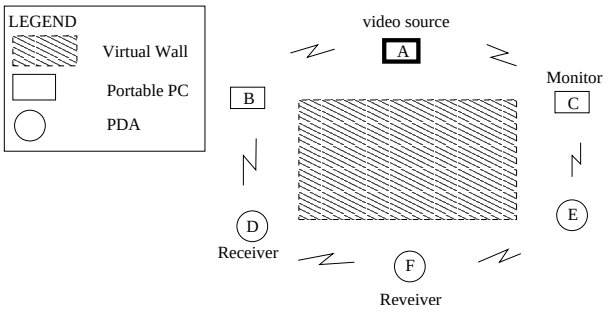


Fig. 4
STATIONARY NETWORK

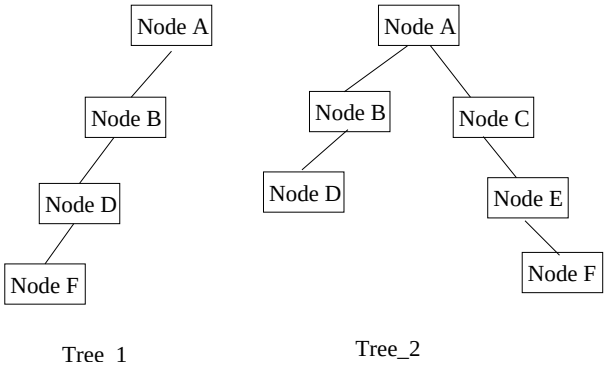


Fig. 5
TREE STRUCTURE IN STATIONARY NETWORK

TABLE I
MRDC IN STATIONARY NETWORK WITH ONE MULTICAST SOURCE

| | Value | % of total |
|---|---|---|
| MRDC control packet O/H | 0.26 kb/s | 0.07% |
| MRDC data packet header | 15.75 kb/s | 4.11% |
| Total MRDC O/H | 16.01 kb/s | 4.18% |
| IGMP O/H | 0.04 kb/s | 0.01% |
| Tree Monitoring O/H | 1.29 kb/s | 0.34% |
| Avg. # of multicast tree branch | 1.4 | N/A |
| Effective data throughput | 365.36 kb/s | 95.47% |
| Total throughput | 382.7 kb/s | 100% |

to the conference group. Because vic sends multicast packets at regular intervals to announce membership to other vics, although there is only one video source at application level, each vic is a multicast source at the point view of MRDC. Thus this conference group is a multiple source scenario for MRDC. We configured IP address of wireless nodes to satisfy $@A < @B < @C < @D < @E < @F$. We ran vic first on node D and then on nodes A, E and F. This running sequence resulted in that node D became core. Figure 7 demonstrates the tree structure.

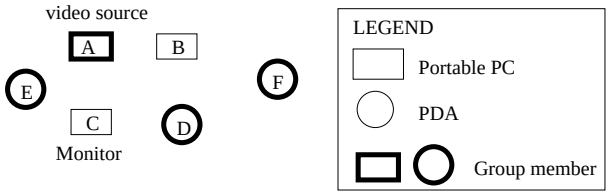


Fig. 6
DYNAMIC NETWORK

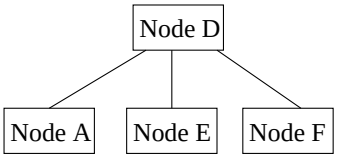


Fig. 7
INITIAL MULTICAST TREE STRUCTURE

### *B. Dynamic Network Scenario and Results*

In this test, we removed the virtual wall constructed by iptables. All nodes were initially in the coverage region of the others as shown in Figure 6. Vic [11] addressed to a multicast address ran on node A, D, E and F to form a video conference group. A web-cam is connected to node A to serve as a multicast source and send video stream

This test contains two parts: topology dynamic part and membership dynamic part to test correctness and efficiency of MRDC implementation. In topology dynamic part, we moved node F outside of the coverage range of node A and D but still in the coverage of node B. During the movement, node F firstly receives

video stream directly from node A. Then multicast tree structure changes (see Figure 8) and node F get video stream through the relay of node B. In membership dynamic part, we kept node F outside of the coverage range of A and D but in the coverage of node B and stopped vic on node D. Tree monitoring showed that tree structure changed, and after a short transient time, F became core and node D disappeared. Then, we re-ran vic on node D. This node joined the tree as a leaf node as shown in Figure 9.

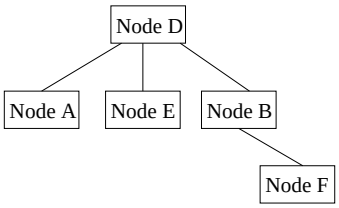


Fig. 8
MULTICAST TREE STRUCTURE AFTER MOVEMENT

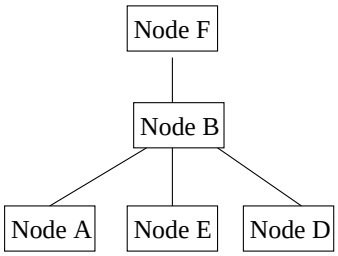


Fig. 9
FINAL MULTICAST TREE STRUCTURE

During this test, the video transmission rate was around 300kb/s. The replayed video was fluent in both portable PCs and PDAs and the error rate shown in vic in most of time was smaller than 10%.

## V. CONCLUSION

We presented our experience on implementation of MRDC (Multicast Routing protocol with Dynamic Core), a tree-based on-demand multicast routing protocol for MANET, in a testbed which consisted of portable PCs and PDAs. MRDC does not depend on any unicast routing protocol. The hybrid network configuration reduce hardware cost of the testbed and facilitates mobility test since PDAs are smaller and cheaper than portable PCs. The main parts of MRDC, including tree construction and maintenance, have been successfully implemented in the user space of Linux operating system. We also designed a forwarding module to solve the problems of multicast datagram forwarding and realize on-demand fashion when program runs in user space. Besides these, IGMP module and tree monitoring module have been designed and integrated in the MRDC implementation to form a complete solution for multicast application supporting and topology monitoring. We evaluated the bandwidth utilization of this implementation in a stationary network scenario and showed that if we do not consider encapsulation overhead, MRDC creates a little control overhead for multicast traffic delivery. Then, we used a MBone traffic - vic to test MRDC with node movement and membership changes. The results prove that MRDC correctly deal with topology dynamic and membership dynamic.

The success of MRDC implementation in user space encourages us to bring these functionalities into kernel and test the scalability of MRDC. We also plan to compare the performance of MRDC with other ad hoc multicast implementation in our testbed.

## REFERENCES

[1] E. M. Royer and C. E. Perkin. Multicast operation of ad-hoc on-demand distance vector routing protocol. In *ACM/IEEE MobiCom 1999*, August 1999.
[2] C. W. Wu, Y. C. Tay, and C. K. Toh. Amris: A multicast protocol for ad hoc wireless networks. In *IEEE MILCOM'99*, Atlantic City, NJ, USA, November 1999.
[3] Sung-Ju. Lee, Mario Gerla, and Ching-Chuan Chiang. On-demand multicast routing protocol. In *IEEE WCNC'99*, pages 1298–1302, New Orleans, LA, USA, September 1999.
[4] Sagar Sanghani, Timothy X Brown, Shweta Bhandare, and Sheetalkumar Doshi. Ewant: The emulated wireless ad hoc network testbed. In *IEEE WCNC2003*, volume 3, pages 1844–1849, March 2003.
[5] Lusheng Ji, Mary Ishibashi, and M. Scott Corson. An approach to mobile ad hoc network protocol kernel design. In *IEEE WCNC'99*, pages 1303–1307, New Orleans, LA, USA, September 1999.
[6] Sang Ho Bae, Sung-Ju Lee, and Mario Gerla. Multicast protocol implementation and validation in an ad hoc network. In *ICC 2001*, volume 10, pages 3196–3200, Helsinki, Finland, June 2001.
[7] Shiyi Wu and Christian Bonnet. Multicast routing protocol with dynamic core. In *IST 2001*, pages 274–280, Tehran, Iran, Sep. 2001.
[8] Fenner and W. Internet group management protocol version 2. RFC 2236, November 1997.
[9] tcpdump. URL: http://www.tcpdump.org/.
[10] R. Russell. Linux 2.4 packet filtering howto. URL: http://netfilter.samba.org/documentation/.
[11] A video conference application. URL: http://www-nrg.ee.lbl.gov/vic.