

```

/* -----
(c) Riduan M. ABID - eMail: R.Abid@aui.ma, abidmoh@auburn.edu,
This Work is relevant to a Ph.D Dissertation,
Supervisor: Dr. Biaz - eMail: biazsaa@auburn.edu,
Shelby Center for Engineering Technology- Auburn Unveristy - 2009,
-----*/
#include <sys/socket.h>
#include <linux/if_packet.h>
#include <linux/if_ether.h>
#include <linux/if.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/ioctl.h>
#include <asm/types.h>
#include <sys/time.h>

#define BROADCAST_ADDR {0xff, 0xff, 0xff, 0xff, 0xff, 0xff}

struct PREQ {
    __u8 eltID;
    __u8 hop_count;
    __u8 MPP_mac_addr[6];
    __u32 MPP_seq_no;
    double metric;
};

int main(int argc, char* argv[]) {

    int raw_sock, fd, i, PREQs_BROADCAST_PERIOD, PREQ_size = sizeof(struct PREQ);
    __u8 dest_mac_addr[6] = BROADCAST_ADDR, MPP_mac_addr[6], buffer[1024];
    struct sockaddr_ll sll;
    struct ifreq ifr;
    struct PREQ PREQ;

    // Checking for Arguments,
    if (argc < 3) {
        perror("\n Inussufficient Arguments ");
        perror("\n Usage: <executable> <Interface_Name>
<PREQ_BROADCAST_PERIOD>");
        exit(-1);
    }
    PREQs_BROADCAST_PERIOD = atoi(argv[2]); // in Seconds,

    // Getting Local MAC Address,
    fd = socket(AF_INET, SOCK_DGRAM, 0);
    ifr.ifr_addr.sa_family = AF_INET;
    strncpy(ifr.ifr_name, argv[1], IFNAMSIZ-1);
    ioctl(fd, SIOCGIFHWADDR, &ifr);
    close(fd);
    memcpy(MPP_mac_addr, ifr.ifr_hwaddr.sa_data, 6);

```

```

// Finding Local Wireless Interface Index,
if ((raw_sock = socket(PF_PACKET, SOCK_RAW, htons(ETH_P_ALL))) < 0 ) {
    perror("socket");
    exit(-1);
}
memset(&ifr, 0, sizeof(ifr));
strcpy(ifr.ifr_name, argv[1]);
if ( ioctl(raw_sock, SIOCGIFINDEX, &ifr) < 0) {
    perror("ioctl (SIOCGIFINDEX) ");
    exit(-1);
}

// Bind the socket to the interface,
memset(&sll, 0, sizeof(sll));
sll.sll_family = AF_PACKET;
sll.sll_protocol = ETH_P_ALL;
sll.sll_ifindex = ifr.ifr_ifindex;
if ( bind(raw_sock, (struct sockaddr *) &sll, sizeof(sll)) < 0) {
    perror("Error Binding to ETH_P_ALL");
    exit(-1);
}

// Set the MAC address of source and destination
memcpy(sll.sll_addr, dest_mac_addr, 6);
sll.sll_addr[6] = 0x00;
sll.sll_addr[7] = 0x00;
memcpy(buffer, dest_mac_addr, 6);
memcpy(buffer+6, MPP_mac_addr, 6);

// Filling-Initial PREQ,
PREQ.eltID = 13;
PREQ.hop_count = 0;
memcpy(PREQ.MPP_mac_addr, MPP_mac_addr, 6);
PREQ.MPP_seq_no = 0;
PREQ.metric = 0;
while (1) {
    PREQ.MPP_seq_no++;
    memcpy(buffer+12, &PREQ, PREQ_size);
    if ((i = sendto(raw_sock, buffer, 12+PREQ_size, 0x00, (struct sockaddr
*)&sll, sizeof(sll))) < 0 ) {
        perror("sendto");
        exit(-1);
    }
    // Feedback,
    printf("\n --- Sent %d Bytes - Seq_no: %d, \n", i, PREQ.MPP_seq_no);
    sleep(PREQs_BROADCAST_PERIOD);
}
return 0;
}

```