Sending raw Ethernet packets from a specific interface in C on Linux

Posted on September 14, 2011 by austinmarton

Lately I've been writing some code to send packets to a specific MAC address from a specific interface. I'm sure this will come in handy again so here is how it goes:

Includes:
(might not need all of these)

```
#include <netinet/in.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <net/if.h>
#include <netinet/ip.h>
#include <netinet/udp.h>
#include <netinet/ether.h>
#include <linux/if_packet.h>
```

Open the raw socket:

```
int sockfd;
...
/* Open RAW socket to send on */
if ((sockfd = socket(AF_PACKET, SOCK_RAW, I
    perror("socket");
}
```

Get the index of the interface to send on:

```
struct ifreq if_idx;
...
memset(&if_idx, 0, sizeof(struct ifreq));
strncpy(if_idx.ifr_name, "eth0", IFNAMSIZ-1
if (ioctl(sock, SIOCGIFINDEX, &if_idx) < 0)
    perror("SIOCGIFINDEX");
```

Get the MAC address of the interface to send on:

```
struct ifreq if_mac;
...
memset(&if_mac, 0, sizeof(struct ifreq));
strncpy(if_mac.ifr_name, "eth0", IFNAMSIZ-1
if (ioctl(sock, SIOCGIFHWADDR, &if_mac) < 0
    perror("SIOCGIFHWADDR");
```

Get the IP address of the interface to send on:

```
struct ifreq if_ip;
...
memset(&if_ip, 0, sizeof(struct ifreq));
strncpy(if_ip.ifr_name, "eth0", IFNAMSIZ-1)
if (ioctl(sock, SIOCGIFADDR, &if_ip) < 0)
    perror("SIOCGIFADDR");
```

Construct the Ethernet header:

```
int tx_len = 0;
char sendbuf[1024];
struct ether_header *eh = (struct ether_hea
...
memset(sendbuf, 0, 1024);
/* Ethernet header */
eh->ether_shost[0] = ((uint8_t *)&if_mac.if
eh->ether_shost[1] = ((uint8_t *)&if_mac.if
eh->ether_shost[2] = ((uint8_t *)&if_mac.if
eh->ether_shost[3] = ((uint8_t *)&if_mac.if
eh->ether_shost[4] = ((uint8_t *)&if_mac.if
eh->ether_shost[5] = ((uint8_t *)&if_mac.if
eh->ether_dhost[0] = MY_DEST_MAC0;
eh->ether_dhost[1] = MY_DEST_MAC1;
```

```
eh->ether_dhost[2] = MY_DEST_MAC2;
eh->ether_dhost[3] = MY_DEST_MAC3;
eh->ether_dhost[4] = MY_DEST_MAC4;
eh->ether_dhost[5] = MY_DEST_MAC5;
eh->ether_type = htons(ETH_P_IP);
tx_len += sizeof(struct ether_header);
```

Construct the IP header:

```
struct iphdr *iph = (struct iphdr *) (sendb
...
/* IP Header */
iph->ihl = 5;
iph->version = 4;
iph->tos = 16; // Low delay
iph->id = htons(54321);
iph->ttl = ttl; // hops
iph->protocol = 17; // UDP
/* Source IP address, can be spoofed */
iph->saddr = inet_addr(inet_ntoa(((struct s
// iph->saddr = inet_addr("192.168.0.112");
/* Destination IP address */
iph->daddr = inet_addr("192.168.0.111");
tx_len += sizeof(struct iphdr);
```

Construct the UDP header:

```
struct udphdr *udph = (struct udphdr *) (se
...
/* UDP Header */
udph->source = htons(3423);
udph->dest = htons(5342);
udph->check = 0; // skip
tx_len += sizeof(struct udphdr);
```

Fill in UDP payload:

```
/* Packet data */
sendbuf[tx_len++] = 0xde;
sendbuf[tx_len++] = 0xad;
sendbuf[tx_len++] = 0xbe;
sendbuf[tx_len++] = 0xef;
```

Fill in remaining header info:

```
unsigned short csum(unsigned short *buf, in
```

```
{
    unsigned long sum;
    for(sum=0; nwords>0; nwords--)
        sum += *buf++;
    sum = (sum >> 16) + (sum &0xffff);
    sum += (sum >> 16);
    return (unsigned short)(~sum);
}
...
/* Length of UDP payload and header */
udph->len = htons(tx_len - sizeof(struct et
/* Length of IP payload and header */
iph->tot_len = htons(tx_len - sizeof(struct
/* Calculate IP checksum on completed heade
iph->check = csum((unsigned short *)(sendbu
```

Send the raw Ethernet packet:

```
/* Destination address */
struct sockaddr_ll socket_address;
...
/* Index of the network device */
socket_address.sll_ifindex = if_idx.ifr_ifi
/* Address length*/
```

```
socket_address.sll_halen = ETH_ALEN;
/* Destination MAC */
socket_address.sll_addr[0] = MY_DEST_MAC0;
socket_address.sll_addr[1] = MY_DEST_MAC1;
socket_address.sll_addr[2] = MY_DEST_MAC2;
socket_address.sll_addr[3] = MY_DEST_MAC3;
socket_address.sll_addr[4] = MY_DEST_MAC4;
socket_address.sll_addr[5] = MY_DEST_MAC5;
/* Send packet */
if (sendto(sock, sendbuf, tx_len, 0, (struc
    printf("Send failed\n");
```

Update:
As in the comments, I've written a working example that can be found here: https://gist.github.com/1922600

Change the destination MAC address (e.g. 00:11:22:33:44:55) and compile:

```
gcc sendRawEth.c -o sendRawEth
```

In one terminal run tcpdump to observe the packets:

```
sudo tcpdump -nettti eth0 '(ether dst host
```

And in another run the program as root:

```
sudo ./sendRawEth eth0
```

References:

http://aschauf.landshut.org/fh/linux/udp_vs_raw/ch01s03.html
http://www.tenouk.com/Module43a.html
http://linux.die.net/man/3/sendto