Practical 3

# Aim:

Pcap Program and Filters

# Code

#include <pcap.h>

#include <stdio.h>

#include <stdlib.h>

#include <errno.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <netinet/if\_ether.h>

void my\_callback**(**u\_char **\***args**,** const struct pcap\_pkthdr**\*** pkthdr**,** const u\_char**\***

packet**)**

**{**

static int count **=** 1**;**

fprintf**(**stdout**,** "%3d, "**,** count**);**

fflush**(**stdout**);**

count**++;**

**}**

void another\_callback**(**u\_char **\***arg**,** const struct pcap\_pkthdr**\*** pkthdr**,**

const u\_char**\*** packet**)**

**{**

int i**=**0**;**

static int count**=**0**;**

printf**(**"Packet Count: %d\n"**,** **++**count**);** /\* Number of Packets \*/

printf**(**"Recieved Packet Size: %d\n"**,** pkthdr**->**len**);** /\* Length of header \*/

printf**(**"Payload:\n"**);** /\* And now the data \*/

**for(**i**=**0**;**i**<**pkthdr**->**len**;**i**++)** **{**

**if(**isprint**(**packet**[**i**]))** /\* Check if the packet data is printable \*/

printf("%c ",packet[i]); /\* Print it \*/

else

printf(" . ",packet[i]); /\* If not print a . \*/

if((i%16==0 && i!=0) || i==pkthdr->len-1)

printf("\n");

}

}

int main(int argc,char \*\*argv)

{

int i;

char \*dev;

char errbuf[PCAP\_ERRBUF\_SIZE];

pcap\_t\* descr;

const u\_char \*packet;

struct pcap\_pkthdr hdr;

struct ether\_header \*eptr; /\* net/ethernet.h \*/

struct bpf\_program fp; /\* hold compiled program \*/

bpf\_u\_int32 maskp; /\* subnet mask \*/

bpf\_u\_int32 netp; /\* ip \*/

if(argc != 2){

fprintf(stdout, "Usage: %s \"expression\"\n"

,argv[0]);

return 0;

}

/\* Now get a device \*/

dev = pcap\_lookupdev(errbuf);

if(dev == NULL) {

fprintf(stderr, "%s\n", errbuf);

exit(1);

}

/\* Get the network address and mask \*/

pcap\_lookupnet(dev, &netp, &maskp, errbuf);

/\* open device for reading in promiscuous mode \*/

descr = pcap\_open\_live(dev, BUFSIZ, 1,-1, errbuf);

if(descr == NULL) {

printf("pcap\_open\_live(): %s\n", errbuf);

exit(1);

}

/\* Now we'll compile the filter expression\*/

if(pcap\_compile(descr, &fp, argv[1], 0, netp) == -1) {

fprintf(stderr, "Error calling pcap\_compile\n");

exit(1);

}

/\* set the filter \*/

if(pcap\_setfilter(descr, &fp) == -1) {

fprintf(stderr, "Error setting filter\n");

exit(1);

}

/\* loop for callback function \*/

pcap\_loop(descr, -1, another\_callback, NULL);

return 0;

}

# Screenshots

