\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report: HW7

Author: E64061151 林友鈞 <[E64061151@mail.ncku.edu.tw](mailto:wang@xiaoming.tw)>

Class: 資訊系二乙

Description:

這份作業讓我學到該如何parse packet field以及學到該如何讀binary file.

以下將解釋作業中的資料結構以及各個函數的功能：

struct Ans: 記錄packet 中各個protocol的數量

unsigned TCP\_num 記錄TCP的數量。

unsigned UDP\_num 記錄UDP的數量。

unsigned ICMP\_num 記錄ICMP的數量。

void ANS\_initialize(Ans \* ans)

對Ans物件做初始化。

parameter : Ans物件的指標。

void Skip(FILE \* fp, int num):

在fp中跳過幾個num個bytes。雖然可以用fread去跳過幾個bytes，但考量到fread傳入的參數擁有的空間不一定足夠，因此寫了Skip去跳過Application field的data。

parameter:

FILE \* fp : 需要被跳過幾個bytes的file。

int num : 需要被跳過num個bytes。

void PrintMac(const char \* option, unsigned char \* data)

將MAC印出。

parameter :

const char \* option : 傳入要SRC MAC 或是DST MAC。

unsigned char \* data : MAC 的 data 。

void PrintIP(const char \* option, unsigned char \* data)

將IP印出。

parameter :

const char \* option : 傳入要SRC IP 或是DST IP。

unsigned char \* data : IP 的 data 。

Ans \* READING(FILE \* fp)

傳入檔案，parse field。

parameter :

FILE \* fp : 檔案。

函數中一開始會先計算fp檔案中的bytes數(int length)。counter 記錄著目前已經讀過多少bytes。當counter == length時代表檔案讀取完畢就會跳出迴圈。packet\_pos記錄著目前的讀到packet的哪個位置。當TCP的部分讀完剩下Application的部分時就會呼叫Skip(fp, packet\_length - packet\_pos)將剩下的data跳過。在讀檔的過程中也會順便計算各個protocol的數目。最後回傳Ans \* 物件。

int main(int argh , char const \* argv[])

開啟檔案，並呼叫READING函數做讀檔，最後將Ans裡的資料輸出。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Code:

#include <stdio.h>

#include <stdlib.h>

#include <stdint.h>

typedef struct{

unsigned int TCP\_num;

unsigned int UDP\_num;

unsigned int ICMP\_num;

}Ans;

void ANS\_initialize(Ans \* ans){

ans->TCP\_num = ans->UDP\_num = ans->ICMP\_num = 0;

}

void Skip(FILE \* fp,unsigned int num){

char \* skip = (char \*)malloc(num \* sizeof(char));

fread(skip,sizeof(char),num,fp);

free(skip);

}

void PrintMac(const char \* option,unsigned char \* data){

printf("%s",option);

for(int i = 0; i < 5; i++)

printf("%02x:",data[i]);

printf("%02x\n",data[5]);

}

void PrintIP(const char \* option,unsigned char \* data){

printf("%s",option);

for(int i = 0; i < 3; i++)

printf("%d.",data[i]);

printf("%d\n",data[3]);

}

Ans \* READING(FILE \* fp){

Ans \* ans = (Ans \*)malloc(sizeof(Ans));

ANS\_initialize(ans);

int i = 1,counter = 0,length,packet\_pos;

unsigned char trash[10];

unsigned char protocol;

unsigned char total\_length[2];

unsigned char port[2];

unsigned int packet\_length;

unsigned char data[10];

unsigned char SRC\_IP[4];

unsigned char DST\_IP[4];

fseek(fp,0L,SEEK\_END);

length = ftell(fp);

fseek(fp,0L,SEEK\_SET);

while(counter < length){

packet\_pos = 0;

printf("#%d\n",i);

fread(data,sizeof(char),6,fp);

PrintMac("DST MAC: ",data);

fread(data,sizeof(char),6,fp);

PrintMac("SRC MAC: ",data);

fread(trash,sizeof(char),4,fp);

fread(total\_length,sizeof(char),2,fp);

fread(trash,sizeof(char),5,fp);

fread(&protocol,sizeof(char),1,fp);

fread(trash,sizeof(char),2,fp);

fread(SRC\_IP,sizeof(char),4,fp);

fread(DST\_IP,sizeof(char),4,fp);

packet\_pos += 34;

packet\_length = 256 \* total\_length[0] + total\_length[1] + 14;

if(protocol == 0x06){

printf("Protocol: TCP\n");

PrintIP("SRC IP: ",SRC\_IP);

PrintIP("DST IP: ",DST\_IP);

fread(port,sizeof(char),2,fp);

printf("SRC Port: %d\n",256\*port[0]+port[1]);

fread(port,sizeof(char),2,fp);

printf("DST Port: %d\n",256\*port[0]+port[1]);

packet\_pos += 4;

ans->TCP\_num += 1;

}

else if(protocol == 0x01){

printf("Protocol: ICMP\n");

PrintIP("SRC IP: ",SRC\_IP);

PrintIP("DST IP: ",DST\_IP);

ans->ICMP\_num += 1;

}

else if(protocol == 0x11) {

printf("Protocol: UDP\n");

PrintIP("SRC IP: ",SRC\_IP);

PrintIP("DST IP: ",DST\_IP);

fread(port,sizeof(char),2,fp);

printf("SRC Port: %d\n",256\*port[0]+port[1]);

fread(port,sizeof(char),2,fp);

printf("DST Port: %d\n",256\*port[0]+port[1]);

packet\_pos += 4;

ans->UDP\_num += 1;

}

Skip(fp,packet\_length - packet\_pos);

printf("Packet Length: %d\n\n",packet\_length);

counter += packet\_length;;

i++;

}

printf("Number of Packet: %d\n",i - 1);

return ans;

}

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

FILE \*fp = NULL;

if(argc != 2) {

printf("usage: ./hw7 filename\n");

exit(0);

}

fp = fopen(argv[1],"rb");

if(!fp){

printf("Can't open file.\n");

return -1;

}

Ans \* ans;

ans = READING(fp);

printf("Number of TCP Packet: %d\n",ans->TCP\_num);

printf("Number of UDP Packet: %d\n",ans->UDP\_num);

printf("Number of ICMP Packet: %d\n",ans->ICMP\_num);

return 0;

}

Compilation:

gcc -std=c99 -o hw7 hw7.c

Execution:

./hw7

Output:

> ./hw7 test\_fix.out

#1

DST MAC: 00:08:9b:c4:61:fc

SRC MAC: 08:62:66:50:b8:4e

Protocol: TCP

SRC IP: 140.116.82.165

DST IP: 140.116.82.171

SRC Port: 63625

DST Port: 80

Packet Length: 66

#2

DST MAC: 08:62:66:50:b8:4e

SRC MAC: 00:08:9b:c4:61:fc

Protocol: TCP

SRC IP: 140.116.82.171

DST IP: 140.116.82.165

SRC Port: 80

DST Port: 63625

Packet Length: 66

#3

DST MAC: 00:08:9b:c4:61:fc

SRC MAC: 08:62:66:50:b8:4e

Protocol: TCP

SRC IP: 140.116.82.165

DST IP: 140.116.82.171

SRC Port: 63625

DST Port: 80

Packet Length: 54

…………(中間為了省紙做環保所以省略)

#34

DST MAC: 08:62:66:50:b8:4e

SRC MAC: 00:08:9b:c4:61:fc

Protocol: ICMP

SRC IP: 140.116.82.171

DST IP: 140.116.82.165

Packet Length: 74

#35

DST MAC: 00:08:9b:c4:61:fc

SRC MAC: 08:62:66:50:b8:4e

Protocol: UDP

SRC IP: 140.116.82.165

DST IP: 140.116.82.171

SRC Port: 57229

DST Port: 1013

Packet Length: 61

Number of Packet: 35

Number of TCP Packet: 25

Number of UDP Packet: 6

Number of ICMP Packet: 4