#include "fill\_packet.h"

#include <arpa/inet.h>

#include <unistd.h>

#include <sys/types.h>

void

fill\_iphdr ( struct ip \*ip\_hdr , const char\* dst\_ip)

{

ip\_hdr->ip\_hl = 0x07; /\* header length \*/

ip\_hdr->ip\_v = 0x04; /\* version \*/

ip\_hdr->ip\_len = htons(PACKET\_SIZE); // 92byte /\* total length \*/

ip\_hdr->ip\_id = 0; /\* identification \*/

ip\_hdr->ip\_off = htons(IP\_DF); /\* fragment offset field \*/

ip\_hdr->ip\_ttl = 0x40; /\* time to live \*/

ip\_hdr->ip\_p = 0x01; /\* protocol \*/

inet\_aton(dst\_ip,&(ip\_hdr->ip\_dst)); /\* ip\_dst source and dest address \*/

}

void

fill\_icmphdr (struct icmphdr \*icmp\_hdr)

{

extern pid\_t pid;

icmp\_hdr->type = ICMP\_ECHO; // 8位類型

icmp\_hdr->code = 0; // 8位代碼

icmp\_hdr->checksum = 0; // 16位校驗和,從TYPE開始,直到最後一位用戶數據,如果為字節數為奇數則補充一位

icmp\_hdr->un.echo.id = pid; // 識別號（一般用進程號作為識別號）,用於匹配ECHO和ECHO REPLY包

icmp\_hdr->checksum = fill\_cksum(icmp\_hdr);

}

u16

fill\_cksum(struct icmphdr\* icmp\_hdr) // unsigned short = u16

{

int nleft = ICMP\_PACKET\_SIZE;

int sum = 0;

unsigned short \*w = (unsigned short \*)icmp\_hdr;

unsigned short answer = 0;

while( nleft > 1 ) {

sum += \*w++;

nleft -= 2;

}

if( nleft == 1 ) {

\*(unsigned char \*) (&answer) = \*(unsigned char \*) w;

sum += answer;

}

sum = (sum >> 16) + (sum & 0xFFFF);

sum += (sum >> 16);

answer = ~sum;

return (answer);

}