커널 네트워크 소스 편집& 테스트

컴퓨터 통신 실습

홍석준

I. 커널 소스 편집

네트워크 커널 소스(layer 4) 편집

❑ 아래 그림과 같이 printk 문 추가(udp_rcv)

udp.c
/usr/src/linux-4.15.8/net/ipv4

```
        return 0;
}

int udp_rcv(struct sk_buff *skb)
{
        printk("udp_rcv\n");
        return __udp4_lib_rcv(skb, &udp_table, IPPROTO_UDP);
}

void udp_destroy_sock(struct sock *sk)
{
        struct udp_sock *up = udp_sk(sk);
        bool slow = lock_sock_fast(sk);
        udp_flush_pending_frames(sk);
        unlock_sock_fast(sk, slow);
        if (static_key_false(&udp_encap_needed) && up->encap_type) {
                void (*encap_destroy)(struct sock *sk);
                encap_destroy = READ_ONCE(up->encap_destroy);
                if (encap_destroy)
                        encap_destroy(sk);
        }
}
```

C Tab Width: 8 Ln 2286, Col 32 INS

I. 커널 소스 편집

네트워크 커널 소스(layer 4) 편집

❑ 아래 그림과 같이 printk 문 추가(udp_queue_rcv_skb)

udp.c
/usr/src/linux-4.15.8/net/ipv4

```
 */
static int udp_queue_rcv_skb(struct sock *sk, struct sk_buff *skb)
{
        struct udp_sock *up = udp_sk(sk);
        int is_udplite = IS_UDPLITE(sk);

        printk("udp_queue_rcv_skb\n");
        /*
         *      Charge it to the socket, dropping if the queue is full.
         */
        if (!xfrm4_policy_check(sk, XFRM_POLICY_IN, skb))
                goto drop;
        nf_reset(skb);

        if (static_key_false(&udp_encap_needed) && up->encap_type) {
                int (*encap_rcv)(struct sock *sk, struct sk_buff *skb);

                /*
                 * This is an encapsulation socket so pass the skb to
                 * the socket's udp_encap_rcv() hook. Otherwise, just
                 * fall through and pass this up the UDP socket.
                 * up->encap_rcv() returns the following value:
                 * =0 if skb was successfully passed to the encap
                 *    handler or was discarded by it.
                 * >0 if skb should be passed on to UDP.
                 * <0 if skb should be resubmitted as proto -N
```

C ▾ Tab Width: 8 ▾ Ln 1827, Col 14 ▾ INS

I. 커널 소스 편집

네트워크 커널 소스(layer 4) 편집

❑ 아래 그림과 같이 printk 문 추가(udp_recvmsg)

udp.c
/usr/src/linux-4.15.8/net/ipv4

```c
/*
 *      This should be easy, if there is something there we
 *      return it, otherwise we block.
 */

int udp_recvmsg(struct sock *sk, struct msghdr *msg, size_t len, int noblock,
                int flags, int *addr_len)
{
        struct inet_sock *inet = inet_sk(sk);
        DECLARE_SOCKADDR(struct sockaddr_in *, sin, msg->msg_name);
        struct sk_buff *skb;
        unsigned int ulen, copied;
        int peeked, peeking, off;
        int err;
        int is_udplite = IS_UDPLITE(sk);
        bool checksum_valid = false;

        printk("udp_recvmsg\n");

        if (flags & MSG_ERRQUEUE)
                return ip_recv_error(sk, msg, len, addr_len);

try_again:
        peeking = flags & MSG_PEEK;
        off = sk_peek_offset(sk, flags);
        skb = __skb_recv_udp(sk, flags, noblock, &peeked, &off, &err);
```

C Tab Width: 8 Ln 1574, Col 1 INS

I. 커널 소스 편집

네트워크 커널 소스(layer 4) 편집

❑ 아래 그림과 같이 printk 문 추가(udp_sendmsg)

udp.c
/usr/src/linux-4.15.8/net/ipv4

```
}
EXPORT_SYMBOL(udp_push_pending_frames);

int udp_sendmsg(struct sock *sk, struct msghdr *msg, size_t len)
{
        struct inet_sock *inet = inet_sk(sk);
        struct udp_sock *up = udp_sk(sk);
        struct flowi4 fl4_stack;
        struct flowi4 *fl4;
        int ulen = len;
        struct ipcm_cookie ipc;
        struct rtable *rt = NULL;
        int free = 0;
        int connected = 0;
        __be32 daddr, faddr, saddr;
        __be16 dport;
        u8  tos;
        int err, is_udplite = IS_UDPLITE(sk);
        int corkreq = up->corkflag || msg->msg_flags&MSG_MORE;
        int (*getfrag)(void *, char *, int, int, int, struct sk_buff *);
        struct sk_buff *skb;
        struct ip_options_data opt_copy;

        printk("udp_sendmsg\n");

        if (len > 0xFFFF)
```

C Tab Width: 8 Ln 871, Col 42 INS

2. 커널 소스 컴파일 및 설치

네트워크 커널 소스 컴파일 및 설치

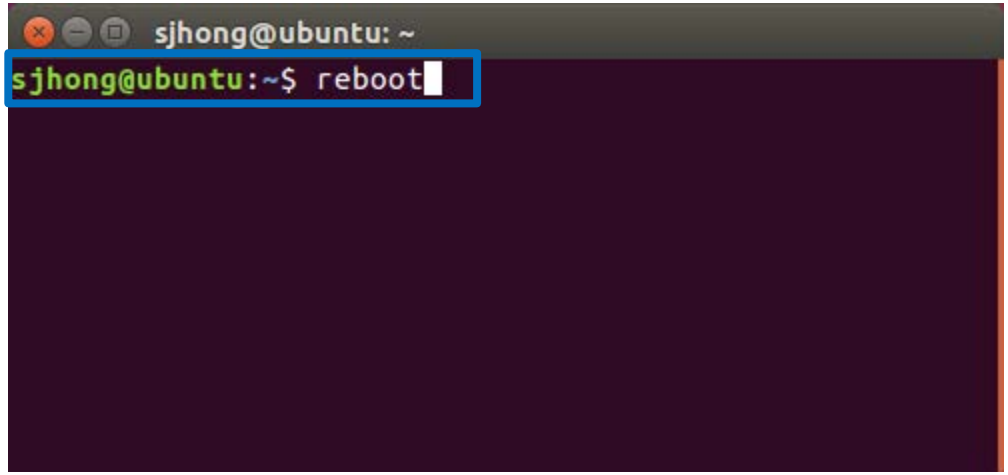
- ❑ 아래 그림과 같이 리눅스 커널의 최상위 폴더로 이동후 컴파일 실행
- ❑ sudo make all && sudo make install로 컴파일 및 설치

```
sjhong@ubuntu: /usr/src/linux-4.15.8
sjhong@ubuntu:~$
sjhong@ubuntu:~$
sjhong@ubuntu:~$
sjhong@ubuntu:~$ cd /usr/src/linux-4.15.8/
sjhong@ubuntu:/usr/src/linux-4.15.8$ ls
arch            drivers   kernel            net          usr
block           firmware  lib               README       virt
built-in.o      fs        MAINTAINERS       samples      vmlinux
certs           include   Makefile          scripts      vmlinux-gdb.py
COPYING         init      mm                security     vmlinux.o
CREDITS         ipc       modules.builtin   sound
crypto          Kbuild    modules.order     System.map
Documentation   Kconfig   Module.symvers    tools
sjhong@ubuntu:/usr/src/linux-4.15.8$ sudo make all && sudo make install
[sudo] password for sjhong:
  CHK     include/config/kernel.release
  CHK     include/generated/uapi/linux/version.h
  CHK     include/generated/utsrelease.h
  CHK     include/generated/timeconst.h
  CHK     include/generated/bounds.h
  CHK     include/generated/asm-offsets.h
  CALL    scripts/checksyscalls.sh
  CHK     scripts/mod/devicetable-offsets.h
  CHK     include/generated/compile.h
```

2. 커널 소스 컴파일 및 설치

네트워크 커널 소스 컴파일 및 설치

- **설치 후 꼭 reboot로 우분투 재시작할 것**
  - reboot를 하기전에는 실제 컴파일된 커널이 적용되지 않음



3. 소켓 프로그램(UDP) 작성

UDP 소켓 프로그램 작성

- 자신의 홈 디렉토리로 이동 혹은 mkdir로 작업용 폴더(자료의 경우 program폴더)를 만들고 이동
- gedit (파일명).c로 소스 코드 편집

```
sjhong@ubuntu: ~/program
built-in.o      fs          MAINTAINERS       samples      vmlinux
certs           include     Makefile          scripts      vmlinux-gdb.py
COPYING         init        mm                security     vmlinux.o
CREDITS         ipc         modules.builtin   sound
crypto          Kbuild      modules.order     System.map
Documentation   Kconfig     Module.symvers    tools
sjhong@ubuntu:/usr/src/linux-4.15.8$ cd
sjhong@ubuntu:~$ ls
Desktop     examples.desktop  ion-3.6.1.tar.gz  nano.save    program    Videos
Documents   ion-3.4.0.tar.gz  ion-open-source   nano.save.1  Public
Downloads   ion-3.6.1         Music             Pictures     Templates
sjhong@ubuntu:~$
sjhong@ubuntu:~$
sjhong@ubuntu:~$
sjhong@ubuntu:~$ ls
Desktop     examples.desktop  ion-3.6.1.tar.gz  nano.save    program    Videos
Documents   ion-3.4.0.tar.gz  ion-open-source   nano.save.1  Public
Downloads   ion-3.6.1         Music             Pictures     Templates
sjhong@ubuntu:~$ cd program/
sjhong@ubuntu:~/program$ ls
udp_echo_client    udp_echo_server    udp_test    udp_test2.c
udp_echo_client.c  udp_echo_server.c  udp_test2   udp_test.c
sjhong@ubuntu:~/program$ gedit udp_echo_server.c
sjhong@ubuntu:~/program$
```

3. 소켓 프로그램(UDP) 작성

UDP 소켓 프로그램 작성 (서버 프로그램)

- ❑ 아래와 같이 UDP 소켓 echo 서버(수신 및 응답) 프로그램 작성
- ❑ gedit udp_echo_server.c 명령어로 c소스 코드를 편집

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>

#define BUFF_SIZE       30


int main(int argc, char **argv)
{
        int     serv_sock;
        char    message[BUFF_SIZE];
        int     str_len;


        struct sockaddr_in      serv_addr;
        struct sockaddr_in      clnt_addr;
        int clnt_addr_size;


        serv_sock = socket(PF_INET, SOCK_DGRAM,0);

        if(-1 == serv_sock)
        {
                printf("sock failed\n");
                exit(1);
        }
```

(계속)

3. 소켓 프로그램(UDP) 작성

UDP 소켓 프로그램 작성 (서버 프로그램)

```
        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family      = AF_INET;
        serv_addr.sin_addr.s_addr= htonl(INADDR_ANY);
        serv_addr.sin_port        = htons(4000);

        if(bind(serv_sock, (struct sockaddr*) &serv_addr, sizeof(serv_addr))==-1)
        {
                printf("bind error\n");
        }

        while(1)
        {
                clnt_addr_size=sizeof(clnt_addr);
                str_len = recvfrom(serv_sock, message, BUFF_SIZE, 0, (struct sockaddr*)&clnt_addr,&clnt_addr_size);
                sendto(serv_sock, message, str_len, 0, (struct sockaddr*)&clnt_addr,sizeof(clnt_addr));

        }

        close(serv_sock);

        return 0;
}
```

3. 소켓 프로그램(UDP) 작성

UDP 소켓 프로그램 작성 (클라이언트 프로그램)

- ❑ 아래와 같이 UDP 소켓 echo 클라이언트(송신 및 응답수신) 프로그램 작성
- ❑ gedit udp_echo_client.c 명령어로 c소스 코드를 편집

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>

#define BUFF_SIZE       30


int main(int argc, char **argv)
{
        int     sock;
        char    message[BUFF_SIZE];
        int     str_len, addr_size;

        struct sockaddr_in      serv_addr;
        struct sockaddr_in      from_addr;

        sock = socket(PF_INET, SOCK_DGRAM,0);

        if(-1 == sock)
        {
                printf("sock failed\n");
                exit(1);
        }

        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family     = AF_INET;
        serv_addr.sin_addr.s_addr= inet_addr("127.0.0.1");
        serv_addr.sin_port       = htons(4000);
```

(계속)

3. 소켓 프로그램(UDP) 작성

UDP 소켓 프로그램 작성 (클라이언트 프로그램)

```
        while(1)
        {
                fgets(message, sizeof(message), stdin);
                if(!strcmp(message, "q\n")) break;
                sendto(sock, message, strlen(message), 0,(struct sockaddr*)&serv_addr, sizeof(serv_addr));
                addr_size=sizeof(from_addr);
                str_len = recvfrom(sock, message, BUFF_SIZE, 0, (struct sockaddr*)&from_addr, &addr_size);
                message[str_len]=0;
                printf("from server: %s", message);

        }

        close(sock);

        return 0;
}
```

4. 소켓 프로그램(UDP) 컴파일

UDP 소켓 프로그램 컴파일

- gcc –o udp_echo_server udp_echo_server.c와
- gcc –o udp_echo_client udp_echo_client.c 명령어 udp 프로그램 컴파일
- 실행 파일 생성 확인(녹색 파일)

```
sjhong@ubuntu: ~/program
sjhong@ubuntu:~/program$ ls
udp_echo_client    udp_echo_server    udp_test   udp_test2.c
udp_echo_client.c  udp_echo_server.c  udp_test2  udp_test.c
sjhong@ubuntu:~/program$ gcc -o udp_echo_server udp_echo_server.c
sjhong@ubuntu:~/program$ gcc -o udp_echo_client udp_echo_client.c
sjhong@ubuntu:~/program$ ls
udp_echo_client    udp_echo_server    udp_test   udp_test2.c
udp_echo_client.c  udp_echo_server.c  udp_test2  udp_test.c
sjhong@ubuntu:~/program$
```

5. 소켓 프로그램(UDP) 실행

UDP 소켓 프로그램 실행

❑ 2개의 프로그램을 각각 터미널에서 띄워서 아래와 같이 동작되는지 확인

```
sjhong@ubuntu: ~/program
sjhong@ubuntu:~/program$ gedit udp_echo_server.c
sjhong@ubuntu:~/program$ ./udp_echo_server
```

```
sjhong@ubuntu: ~/program
sjhong@ubuntu:~/program$ ./udp_echo_client
sdf
from server: sdf
dfb
from server: dfb
swer
from server: swer

from server:
sdf
from server: sdf
wegfsf
from server: wegfsf
sdcf
from server: sdcf
```

6. 커널 메시지 출력 확인

커널 메시지 출력 확인

- 아래 그림과 같이 dmesg 입력 후 net_rx_action/net_tx_action 메시지가 올라오는 지 확인
  - 2계층 메시지는 주기적으로 송수신되기에 계속 출력됨

```
sjhong@ubuntu: ~/program
[ 6255.111209] net_tx_action
[ 6256.040102] net_rx_action ^^
[ 6258.056871] net_rx_action ^^
[ 6260.072248] net_rx_action ^^
[ 6260.999426] net_tx_action
[ 6262.088170] net_rx_action ^^
[ 6264.104102] net_rx_action ^^
[ 6265.863408] net_tx_action
[ 6266.120516] net_rx_action ^^
[ 6268.136896] net_rx_action ^^
[ 6270.151747] net_rx_action ^^
[ 6270.983488] net_tx_action
[ 6272.168190] net_rx_action ^^
[ 6274.184056] net_rx_action ^^
[ 6276.103763] net_tx_action
[ 6276.200050] net_rx_action ^^
[ 6278.216633] net_rx_action ^^
[ 6280.231859] net_rx_action ^^
[ 6281.992172] net_tx_action
[ 6282.248866] net_rx_action ^^
[ 6284.263979] net_rx_action ^^
sjhong@ubuntu:~/program$
```

6. 커널 메시지 출력 확인

커널 메시지 출력 확인

- 아래 그림과 같이 dmesg 입력 후 네트워크 계층(ip) 메시지가 올라오는 지 확인
  - 3계층 메시지는 ping 127.0.0.1 등으로 확인할 수 있음.

```
sjhong@ubuntu: /usr/src/linux-4.15.8/net/ipv4
[ 6396.200148] ip_output
[ 6396.200148] ip_finish_output
[ 6396.200149] ip_finish_output2
[ 6396.200150] __dev_queue_xmit
[ 6396.200155] net_rx_action ^^
[ 6396.200155] process_backlog
[ 6396.200156] __netif_receive_skb_core
[ 6396.200157] ip_rcv
[ 6396.200157] ip_rcv_finish
[ 6396.200158] ip_local_deliver
[ 6396.200158] ip_local_deliver_finish
[ 6396.200164] __ip_local_out
[ 6396.200164] ip_output
[ 6396.200165] ip_finish_output
[ 6396.200165] ip_finish_output2
[ 6396.200165] __dev_queue_xmit
[ 6396.200167] __netif_receive_skb_core
[ 6396.200167] ip_rcv
[ 6396.200167] ip_rcv_finish
[ 6396.200168] ip_local_deliver
[ 6396.200168] ip_local_deliver_finish
[ 6397.161305] net_rx_action ^^
sjhong@ubuntu:/usr/src/linux-4.15.8/net/ipv4$
```

6. 커널 메시지 출력 확인

커널 메시지 출력 확인

- ❑ **위의 UDP 프로그램 실행 후 dmesg로 커널 메시지 확인 및 UDP 커널 메시지가 확인되는지 확인하기**
  - 기본적으로 2계층 메시지가 계속 올라오므로
  - sudo dmesg –c명령으로 메시지를 클리어해주고 다시 출력해볼 것(아래 그림 참조)
  - 메시지 클리어 후 다시 udp프로그램 실행하여 테스트할 것

```
sjhong@ubuntu: ~/program
[ 6052.101694] net_tx_action
[ 6052.422761] net_rx_action ^^
[ 6054.441795] net_rx_action ^^
[ 6056.455579] net_rx_action ^^
[ 6057.989857] net_tx_action
[ 6058.471008] net_rx_action ^^
[ 6060.486847] net_rx_action ^^
sjhong@ubuntu:~/program$ dmesg
[ 6062.502896] net_rx_action ^^
[ 6063.109936] net_tx_action
sjhong@ubuntu:~/program$ sudo dmesg -c
[ 6062.502896] net_rx_action ^^
[ 6063.109936] net_tx_action
[ 6064.518867] net_rx_action ^^
[ 6066.535338] net_rx_action ^^
[ 6068.551048] net_rx_action ^^
[ 6068.998174] net_tx_action
[ 6070.567176] net_rx_action ^^
sjhong@ubuntu:~/program$ dmesg
[ 6072.582888] net_rx_action ^^
[ 6073.862495] net_tx_action
sjhong@ubuntu:~/program$
```

```
sjhong@ubuntu: ~/program
[ 6186.166341] ip_local_deliver_finish
[ 6186.166342] udp_rcv
[ 6186.166344] udp_queue_rcv_skb
[ 6186.166350] udp_recvmsg
[ 6186.166364] udp_sendmsg
[ 6186.166367] __ip_local_out
[ 6186.166367] ip_output
[ 6186.166367] ip_finish_output
[ 6186.166367] ip_finish_output2
[ 6186.166368] __dev_queue_xmit
[ 6186.166369] net_rx_action ^^
[ 6186.166369] process_backlog
[ 6186.166370] __netif_receive_skb_core
[ 6186.166370] ip_rcv
[ 6186.166370] ip_rcv_finish
[ 6186.166370] ip_local_deliver
[ 6186.166371] ip_local_deliver_finish
[ 6186.166371] udp_rcv
[ 6186.166371] udp_queue_rcv_skb
[ 6186.166374] udp_recvmsg
[ 6187.496406] net_rx_action ^^
sjhong@ubuntu:~/program$
```

Thank you for your attention !!