데이터 통신

-Static Router-

01 분반 4 조	
201302366	김규태
201302423	신종욱
201302440	윤동현
201302448	이상인
201302480	정석현
201302502	허원철

1. 실습 개요

- **실습 일시 및 장소**

  실습 일시 : 2017.6.6 - 2017.6.13

  실습 장소 : 공대 5 호관 413 호

- **실습목적**

  기본적인 라우터를 구현한다.

  '라우터'란 네트워크와 네트워크 간 데이터 전송을 위한 장치로서 네트워크 간의 최적의 경로를 설정해주고 해당 경로를 따라 원활한 통신을 가능하게 하는 장치 이다. 라우터는 라우터 내부에 라우팅 테이블이 있으며 이 라우팅 테이블에 네트워크에 대한 경로를 저장하고 있다.

  이러한 기능을 수행하기 위하여 라우터는 PC 와 동일한 하드웨어 구성을 가지게 된다. 즉, 라우팅이나 각종 컨트롤 및 운영체제 명령어가 실행되는 CPU 와 라우팅 테이블, ARP 캐시 같이 휘발성 데이터를 저장할 수 있는 RAM, 라우팅의 설정과 기본 검진 소프트웨어 같이 영구적으로 데이터를 저장 시킬 수 있는 ROM 등의 하드웨어적 구성 요소를 가진다.

  이번의 과제에서는 PC 를 라우터로서 사용하며 랜카드가 2 개이상 장착된 PC 사용을 위해 413 호 실습실을 이용하며 PC 사이에 3 개의 라우터를 사용하여 PC 간의 통신을 구현한다.

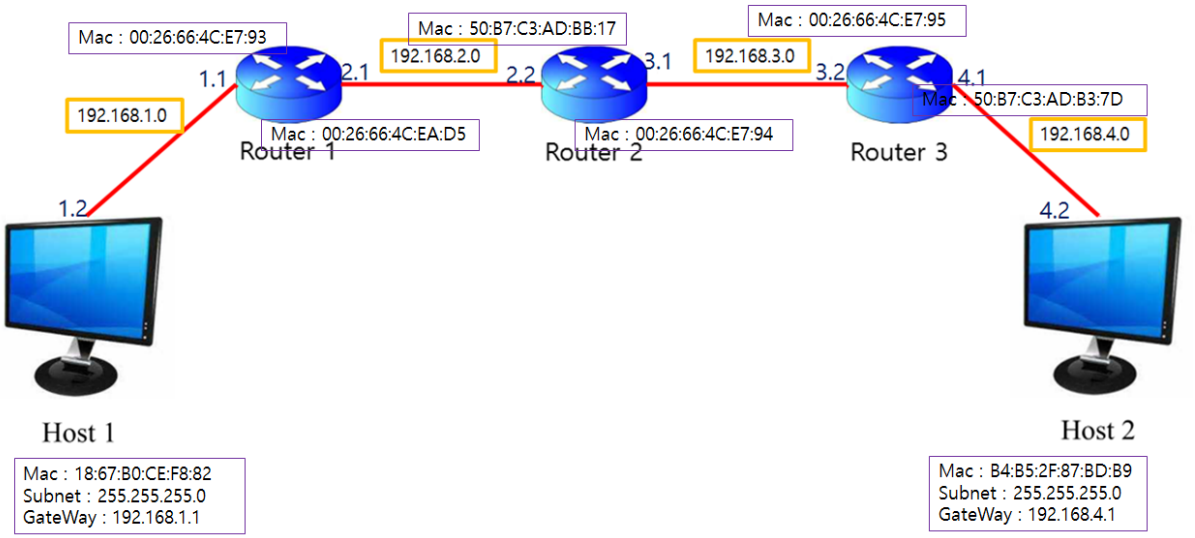
- **실습 시나리오**

  > 라우팅을 위하여 각 PC 의 라우팅 테이블에 네트워크의 정보들을 입력한다.

  > 한쪽 HOST 에서 다른 쪽 HOST 로 ping 을 날려 통신이 정상적으로 이루어지는지 확인한다.

  > wireshark 를 사용하여 전송되는 데이터를 확인하여 정상적으로 전송되는지 확인한다.

2. 동작 과정



- HOST1 에서 HOST2 의 IP 인 192.168.4.2 로 ping 을 전송한다. 이러한 전송이 Router1 에 전달되며 Router1 에서는 테이블에 저장된 NetMask 들과의 and 연산을 통해 192.168.4.0 이라는 주소를 얻어내고 테이블에 저장된 목록 중 해당 IP 와 일치하는 것을 찾아 저장된 것의 Gateway 로 해당 데이터를 전송한다. Router2 에서도 전달받은 IP 를 NetMask 를 사용하여 AND 연산을 실시 한 후 테이블에서 해당하는 IP 주소를 찾아 해당하는 Gateway 로 데이터를 전송한다. Router3 에서도 같은 작업을 반복하는데 Route3 의 192.168.4.0 의 Gateway 는 0.0.0.0 이므로 자신의 네트워크에 데이터를 전달하게 된다. 이후 192.168.4.0 에 속한 192.168.4.2 의 주소를 가지는 HOST2 가 Router2 가 보낸 신호를 받게 되고 자신의 IP 주소와 비교하여 일치하므로 해당 ping 을 수신하며 ping 을 보낸 곳의 IP 주소를 이용하여 응답하는 데이터를 전송하게 된다.

3. 세부 구현

1) ARP Layer

BOOL CARPLayer::Send(unsigned char* ppayload, int nlength,int dev_num)

- ARP Message 를 전송하는 함수이다. 상위 레이어에서 전송을 요청한 패킷에 대해 목적지 IP 가 Cache 테이블에 존재할 경우 목적지 MAC 주소를 변경해 다시 전송해준다. 목적지와 출발지의 IP 가 같은 경우 Gratuitous ARP 이므로 Gratuitous ARP 를 전송해준다. 위 경우에 해당 안되는 경우 버퍼에 메시지를 저장하고 ARP 메시지를 전송한다. 전송에 성공한 경우 true 를 리턴하고 실패한 경우 false 를 리턴 한다.

BOOL CARPLayer::Receive(unsigned char* ppayload,int dev_num)

- ARP Message 를 수신하는 함수이다. opcode 가 request 이면서 목적지가 자기 IP 가 아닌 경우 Proxy 테이블에 존재하면 reply 메시지를 전송한다. Cache 테이블에 존재하면 해당 Entry 를 업데이트하고 없을 경우 테이블에 추가한다. 자신의 IP 로 온 경우 출발지가 자신의 MAC 인 경우 무시한다. 출발지 MAC 이 다른 경우 출발지의 IP 와 자신의 IP 가 같으면 IP 충돌이고 다른 경우 Cache 테이블에 추가하고 reply 메시지를 전송한다.
- opcode 가 reply 인 경우 출발지의 IP 와 자신의 IP 가 같으면 IP 충돌이다. 다른 경우 Cache 테이블에 존재할 경우 해당 Entry 를 업데이트하고 없을 경우 추가한다. 그리고 버퍼에 저장하였던 미 전송 메시지를 다시 전송한 뒤 버퍼를 초기화 한다.

2)RouterDlg.cpp

```
void CRouterDlg::UpdateRouteTable(void)
{
   ListBox_RoutingTable.DeleteAllItems(); //라우팅 테이블에있는거 전부 삭제
   CString dest,netmask,gateway,flag,Interface,metric; //
   POSITION index;
   RoutingTable entry; //head position
   for(int i=0;i<route_table.GetCount();i++){
      flag = "";
      index = route_table.FindIndex(i);
      entry = route_table.GetAt(index);
      dest.Format("%d.%d.%d.%d",entry.Destnation[0],entry.Destnation[1],entry.Destnation[2],entry.Destnation[3]);
      // dest 에 "%d.%d.%d.%d"형식으로 entry.Destnation 저장
      netmask.Format("%d.%d.%d.%d",entry.Netmask[0],entry.Netmask[1],entry.Netmask[2],entry.Netmask[3]);
      // netmask 에 "%d.%d.%d.%d" 형식으로 entry.netmask 저장
      gateway.Format("%d.%d.%d.%d",entry.Gateway[0],entry.Gateway[1],entry.Gateway[2],entry.Gateway[3]);
      // gateway 에 "%d.%d.%d.%d" 형식으로 entry.gateway 저장
      if((entry.Flag & 0x01) == 0x01)//연산해서 0x01 이면
         flag += "U";// 플래그변수에 U 추가
      if((entry.Flag & 0x02) == 0x02)// 연산해서 0x02 이면
         flag += "G"; //플래그 변수에 G 추가
      if((entry.Flag & 0x04) == 0x04)// 연산해서 0x04 면
         flag += "H";// 플래그 변수에 H 추가
      Interface.Format("%d",entry.Interface); //interface 에 entry.Interface(번호) 를 넣는다.
      metric.Format("%d",entry.Metric);// metric 에 Metric(번호)를 넣는다.
      ListBox_RoutingTable.InsertItem(i,dest);// RoutingTable 목적지 추가
      ListBox_RoutingTable.SetItem(i,1,LVIF_TEXT,netmask,0,0,0,NULL);// netmask 추가
      ListBox_RoutingTable.SetItem(i,2,LVIF_TEXT,gateway,0,0,0,NULL);// gateway 추가
      ListBox_RoutingTable.SetItem(i,3,LVIF_TEXT,flag,0,0,0,NULL);// flag 추가
      ListBox_RoutingTable.SetItem(i,4,LVIF_TEXT,Interface,0,0,0,NULL);// interface 추가
      ListBox_RoutingTable.SetItem(i,5,LVIF_TEXT,metric,0,0,0,NULL);// metric 추가
   }
   ListBox_RoutingTable.UpdateWindow();
}//입력받은 라우팅 테이블의 정보들을 라우팅테이블의 기능을 하도록 도착 네트워크주소와 마스킹 게이트웨이 Flag를 설정한다
```

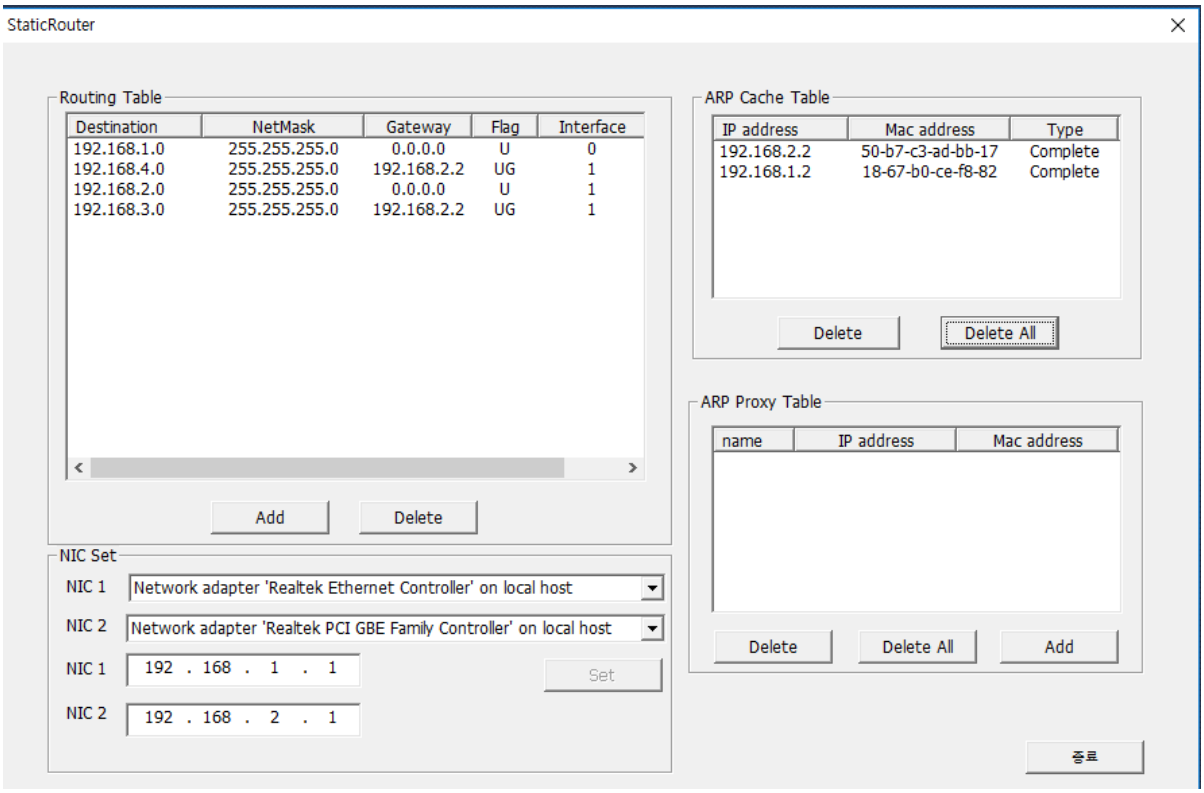
라우팅 테이블 업데이트 하는 함수로 처음에 전부 삭제를 하고 각각 변수를 만든 뒤에 entry 에 저장되어있는 destination netmask gateway interface metric 을 형태를 변환해서 각 변수에 저장한다 중간에 플래그 연산을 통해서 플래그를 추가해주고 마지막에 라우팅 테이블에 set 을 해준다.

```
int CRouterDlg::Routing(unsigned char destip[4]){
    POSITION index;
    RoutingTable entry;
    RoutingTable select_entry;
    entry.Interface = -2;
    select_entry.Interface = -2;
    unsigned char result[4];
    for(int i=0;i<route_table.GetCount();i++){
        index = route_table.FindIndex(i);
        entry = route_table.GetAt(index);
        if(select_entry.Interface == -2){//더이상 entry에 같은게 존재하지않을때
            for(int j=0;j<4;j++)
                result[j] = destip[j] & entry.Netmask[j];//masking을 통해 result값을 구한다
            if(!memcmp(result,entry.Destnation,4)){ //destation일치하는게 있을 경우
                if(((entry.Flag & 0x01) == 0x01) && ((entry.Flag & 0x02) == 0x02)){ //Flag에 UG라서 gateway로 보낸다
                    select_entry = entry;
                    m_IPLayer->SetDstIP(entry.Gateway);
                }
                else if(((entry.Flag & 0x01) == 0x01) && ((entry.Flag & 0x02) == 0x00)){ //gateway가 아닐경우에는 도착주소로 보낸다
                    select_entry = entry;
                    m_IPLayer->SetDstIP(destip);
                }}}
        else{ //entry에 존재할때
            for(int j=0;j<4;j++)
                result[j] = destip[j] & entry.Netmask[j];
            if(memcmp(result,entry.Netmask,4)){ //기존 select비트 보다 더 긴 마스킹을 만족할때
                for(int j=0;j<4;j++)
                    result[j] = destip[j] & entry.Netmask[j];//masking을 하여 나온값을 result에 저장한다
                if(!memcmp(result,entry.Destnation,4)){ //나온 result값이 destation이 같을때
                    if(((entry.Flag & 0x01) == 0x01) && ((entry.Flag & 0x02) == 0x02)){ //Flag가 UG라서 게이트웨이로 보내야한다
                        select_entry = entry;
                        m_IPLayer->SetDstIP(entry.Gateway);
                    }
                    else if(((entry.Flag & 0x01) == 0x01) && ((entry.Flag & 0x02) == 0x00)){ //Flag가 G가 아닐경우에는 해당 주소로 보낸다.
                        select_entry = entry;
                        m_IPLayer->SetDstIP(destip);
                    }
                }
            }
            //더 적을땐 확인할필요 없으니 다시 for문으로간다.
        }
    }
    return select_entry.Interface+1;
}
```

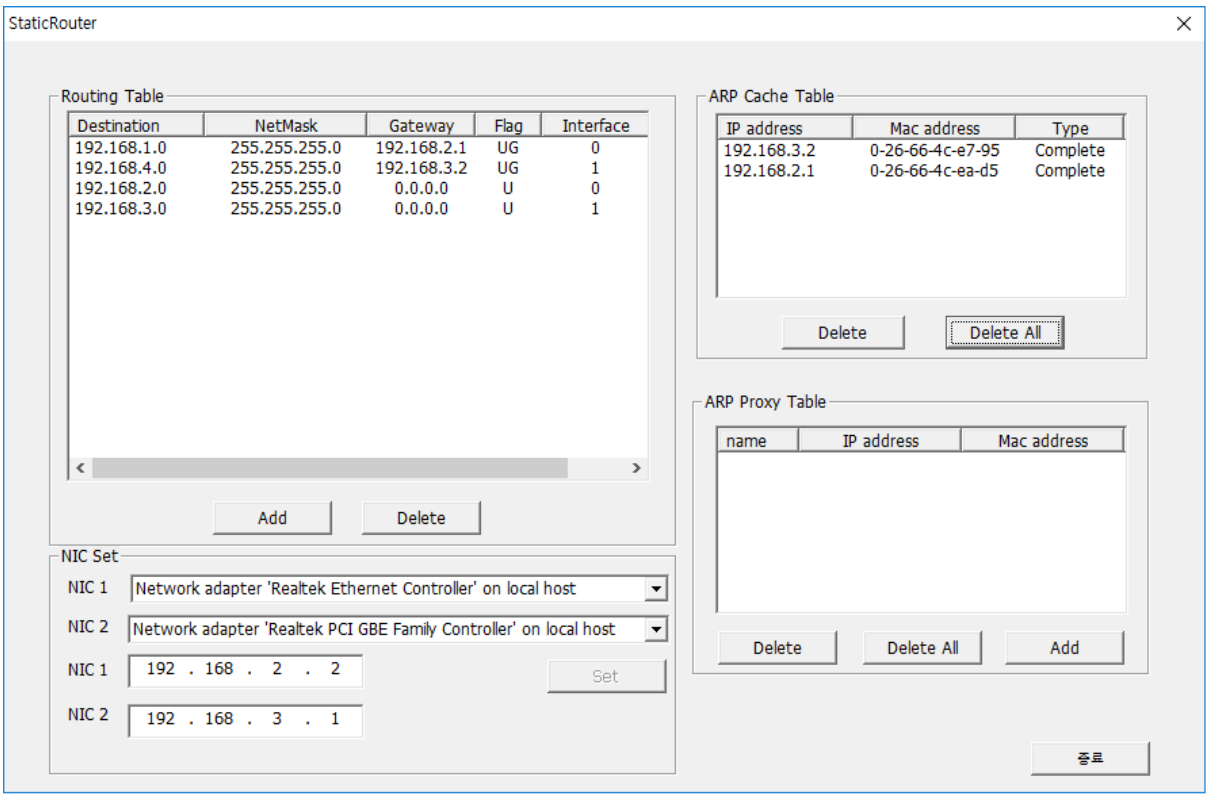
라우터에 패킷이 들어오면 해당 패킷의 Destination IP 주소를 추출해 낸 다음 라우팅 테이블에 있는 mask 값으로 Masking 을하여서 나온 값이 라우팅테이블의 Destination 과 일치 한지 확인하는 과정이다. 최대한 많이 일치하는 것을 찾아 해당 Entry 의 Flag 를 보고 Gateway 인지 네트워크 주소인지 호스트인지 판단 후에 G 일경우에는 하나의 라우터를 더 거쳐야 함으로 Entry 에있는 gateway 주소로 보내고 네트워크주소라면 현재 라우터에 직접 연결된 상태임으로 해당 주소로 패킷을 보내주면 된다.

4. 실행 결과

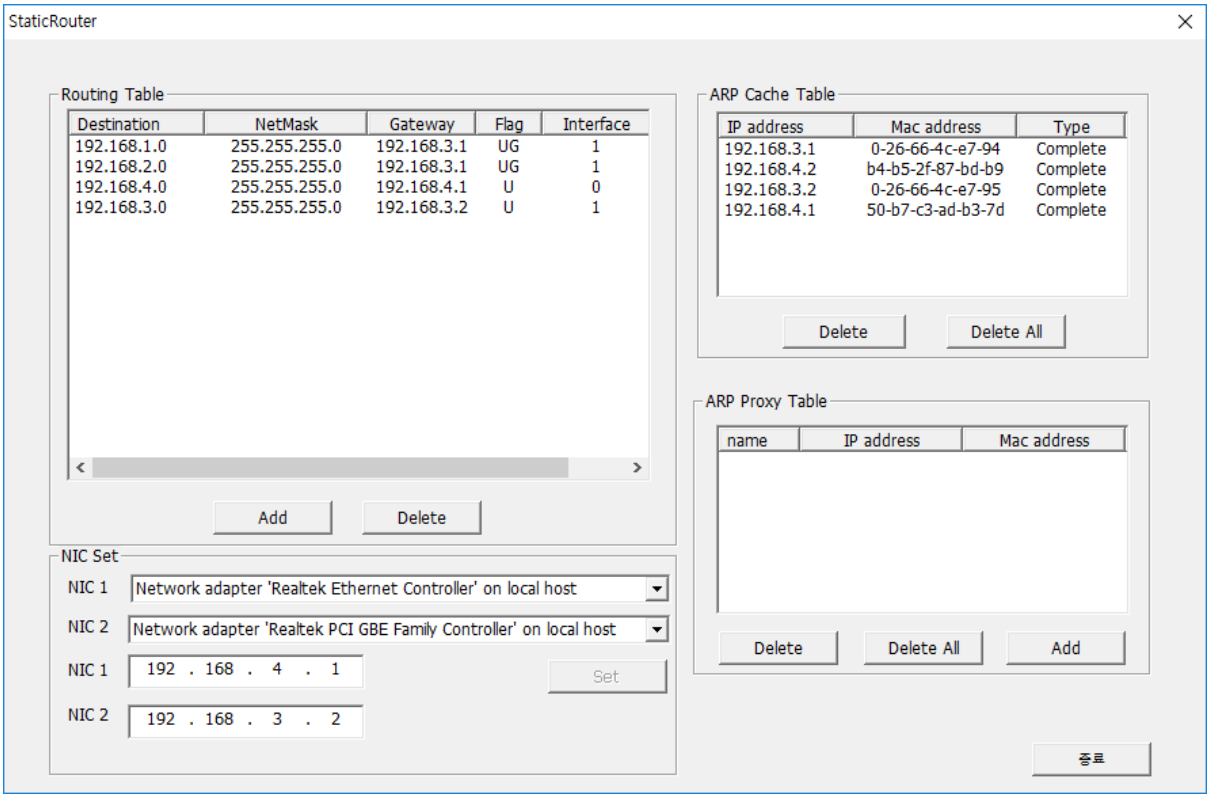
1) Router 1



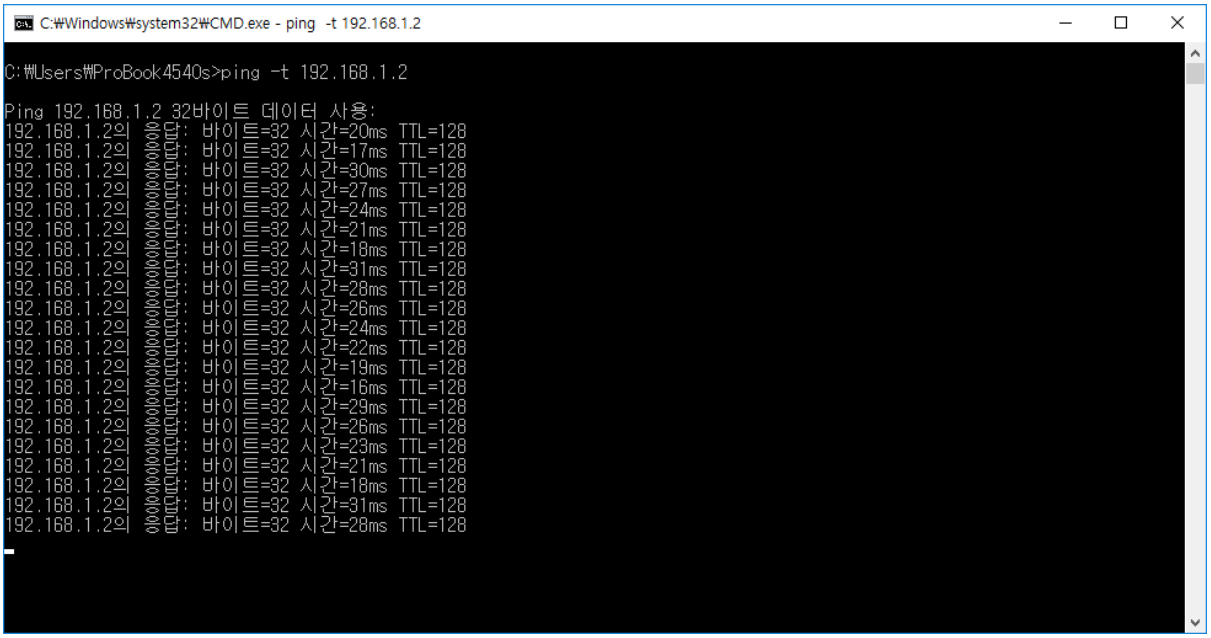
2) Router 2

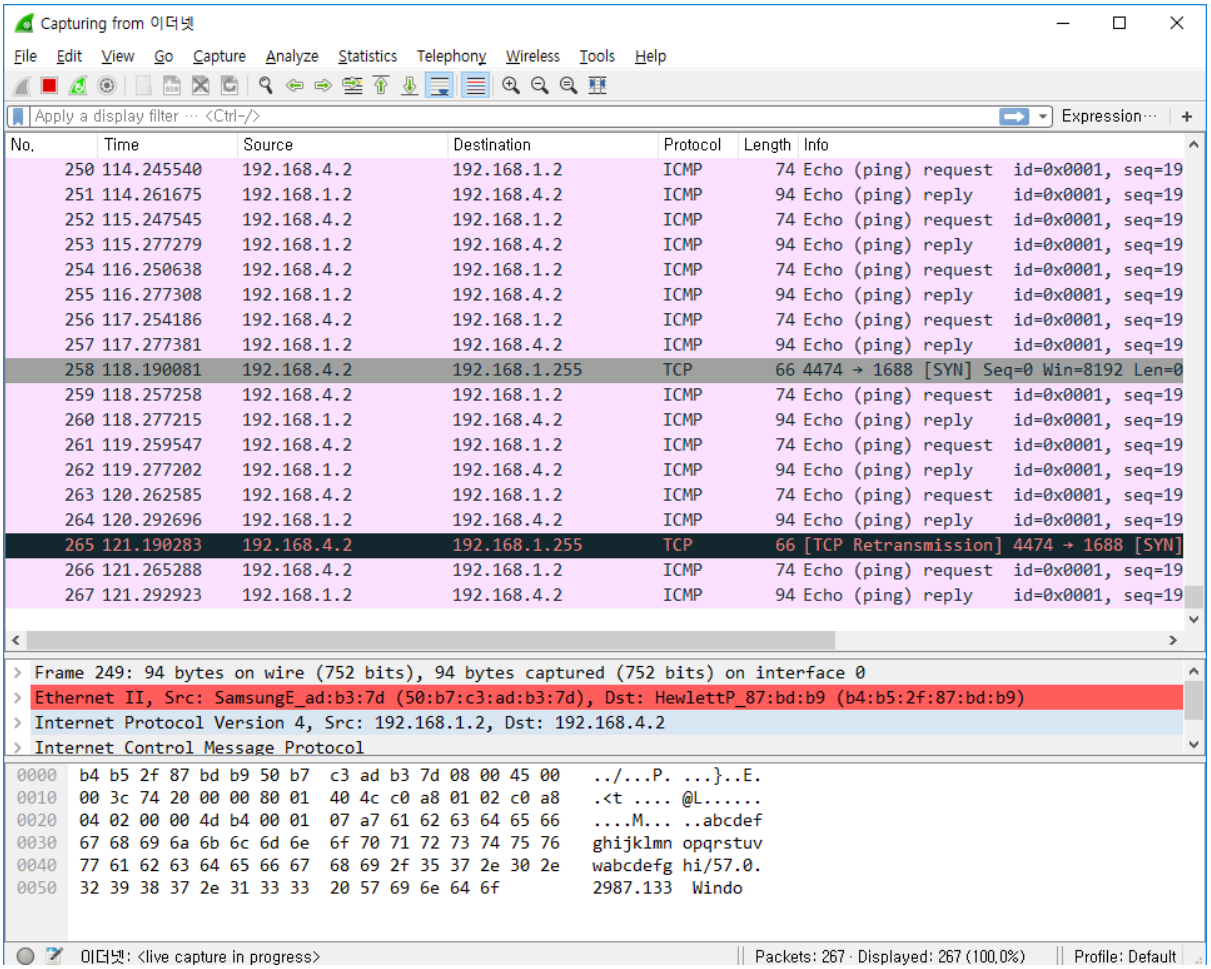


3) Router 3

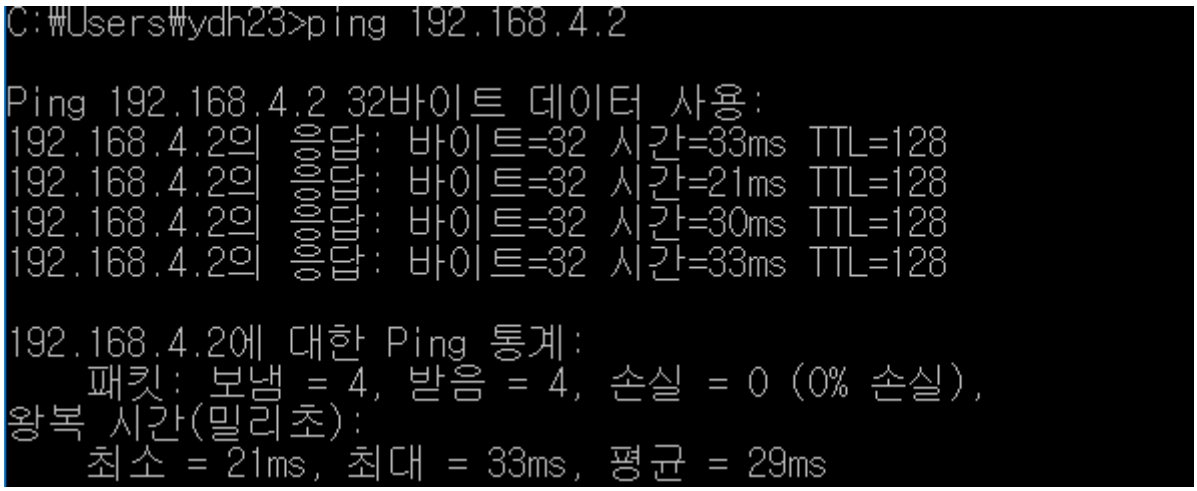


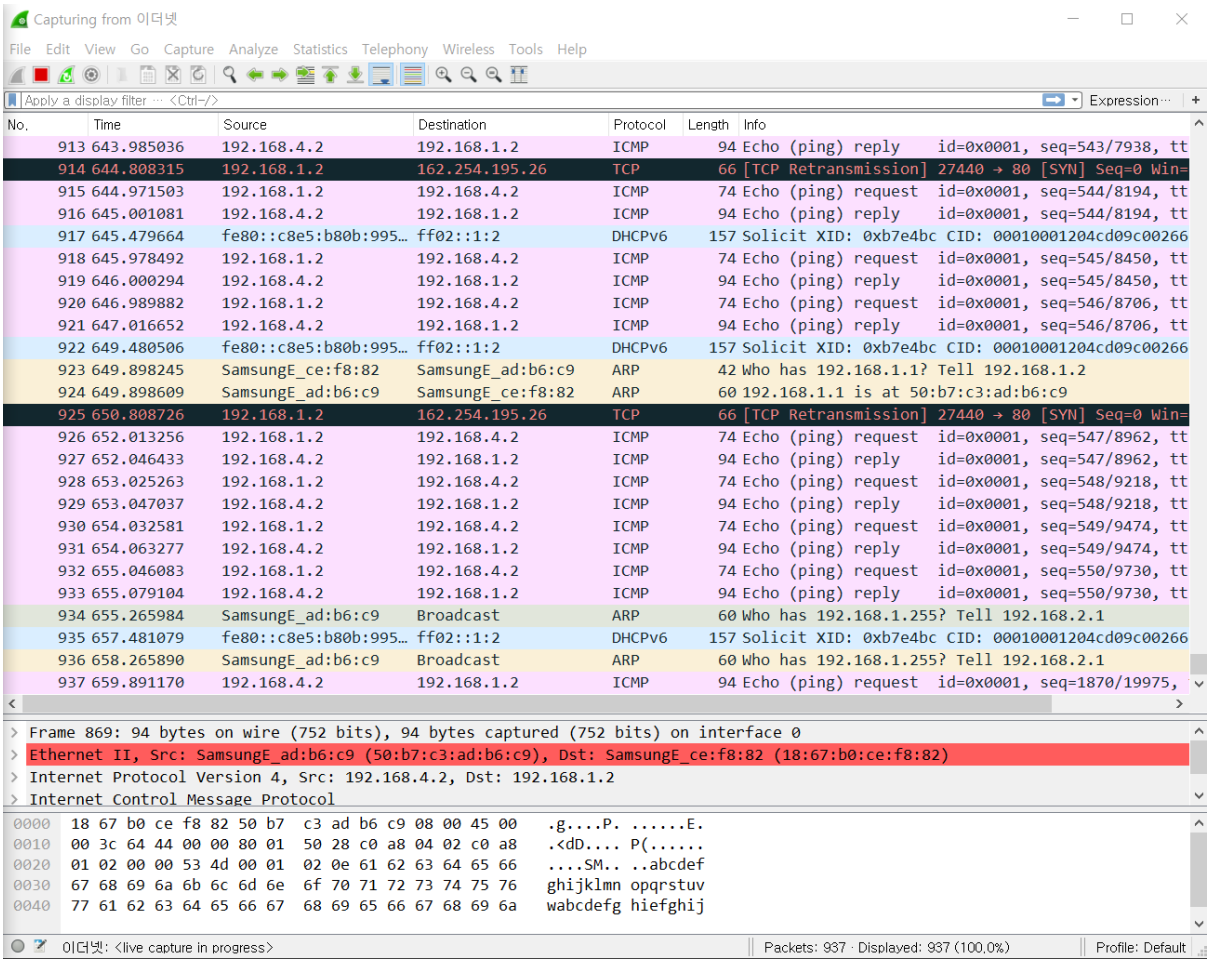
4) Host 2 -> Host 1





5) Host 1 -> Host 2





5. 느낀점

김규태

- Test 하면서 필요한게 많아서 힘들었다.

신종욱

- 구현에 어려움도 많았지만 배운게 많은 과제였다.

윤동현

- 마지막 과제라 힘들었다. 한학기동안 같이 과제한 조원들에게 감사하다.

이상인

- router 를 통한 통신에 대해 더 깊게 공부할 수 있었고 기말고사와 연관되어 공부할 수 있는 실습이었기 때문에 보람찼다.

정석현

- 하기 힘든 과제였다. 구현에 부족한 점이 많은 것 같다

허원철

- 처음 실습은 할만했는데 라우팅은 너무 어려웠다.