

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.

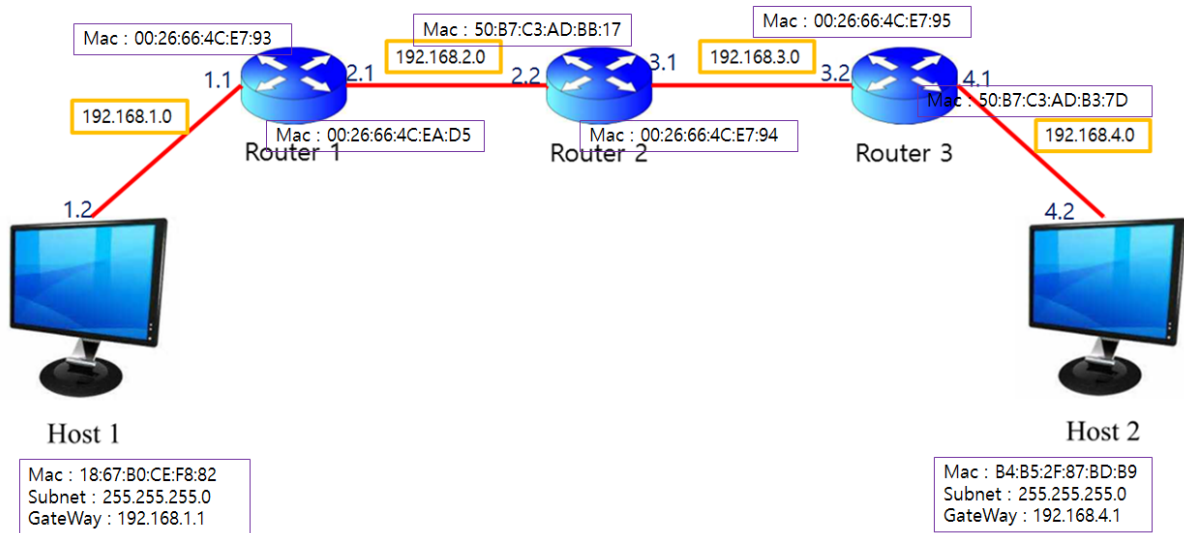
이러한 기능을 수행하기 위하여 라우터는 PC 와 동일한 하드웨어 구성을 가지게 된다. 즉, 라우팅이나 각종 컨트롤 및 운영체제 명령어가 실행되는 CPU 와 라우팅 테이블, ARP 캐시 같이 휘발성 데이터를 저장할 수 있는 RAM, 라우팅의 설정과 기본 검진 소프트웨어 같이 영구적으로 데이터를 저장 시킬 수 있는 ROM 등의 하드웨어적 구성 요소를 가진다.

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 Router3 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.Destination[0],entry.Destination[1],entry.Destination[2],entry.Destination[3]);
        // dest 에 "%d.%d.%d.%d"형식으로 entry.Destination 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"; //플래그 변수에 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.Destination, 4)){ //destination 일치하는게 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 아닐 경우에는 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.Destination, 4)){ //기존 select비트 보다 더 긴 마스크를 만족할때
                for(int j=0; j<4; j++){
                    result[j] = destip[j] & entry.Netmask[j]; //masking을 하여 나온값을 result에 저장한다
                }
                if(!memcmp(result, entry.Destination, 4)){ //나온 result값이 destination이 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 아닐 경우에는 해당 주소로 보낸다.
                        select_entry = entry;
                        m_IPLayer->SetDstIP(destip);
                    }
                }
            }
        }
    }
    //더 적을때 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 일 경우에는 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

StaticRouter

Routing Table

Destination	NetMask	Gateway	Flag	Interface
192.168.1.0	255.255.255.0	0.0.0.0	U	0
192.168.4.0	255.255.255.0	192.168.2.2	UG	1
192.168.2.0	255.255.255.0	0.0.0.0	U	1
192.168.3.0	255.255.255.0	192.168.2.2	UG	1

< Add Delete >

NIC Set

NIC 1 Network adapter 'Realtek Ethernet Controller' on local host

NIC 2 Network adapter 'Realtek PCI GBE Family Controller' on local host

NIC 1 192 . 168 . 1 . 1 Set

NIC 2 192 . 168 . 2 . 1

ARP Cache Table

IP address	Mac address	Type
192.168.2.2	50-b7-c3-ad-bb-17	Complete
192.168.1.2	18-67-b0-ce-f8-82	Complete

Delete Delete All

ARP Proxy Table

name	IP address	Mac address
------	------------	-------------

Delete Delete All Add

종료

2) Router 2

StaticRouter

Routing Table

Destination	NetMask	Gateway	Flag	Interface
192.168.1.0	255.255.255.0	192.168.2.1	UG	0
192.168.4.0	255.255.255.0	192.168.3.2	UG	1
192.168.2.0	255.255.255.0	0.0.0.0	U	0
192.168.3.0	255.255.255.0	0.0.0.0	U	1

< Add Delete >

NIC Set

NIC 1 Network adapter 'Realtek Ethernet Controller' on local host

NIC 2 Network adapter 'Realtek PCI GBE Family Controller' on local host

NIC 1 192 . 168 . 2 . 2 Set

NIC 2 192 . 168 . 3 . 1

ARP Cache Table

IP address	Mac address	Type
192.168.3.2	0-26-66-4c-e7-95	Complete
192.168.2.1	0-26-66-4c-ea-d5	Complete

Delete Delete All

ARP Proxy Table

name	IP address	Mac address
------	------------	-------------

Delete Delete All Add

종료

3) Router 3

StaticRouter

Routing Table

Destination	NetMask	Gateway	Flag	Interface
192.168.1.0	255.255.255.0	192.168.3.1	UG	1
192.168.2.0	255.255.255.0	192.168.3.1	UG	1
192.168.4.0	255.255.255.0	192.168.4.1	U	0
192.168.3.0	255.255.255.0	192.168.3.2	U	1

Add

Delete

NIC Set

NIC 1

Network adapter 'Realtek Ethernet Controller' on local host

NIC 2

Network adapter 'Realtek PCI GBE Family Controller' on local host

NIC 1

192 . 168 . 4 . 1

Set

NIC 2

192 . 168 . 3 . 2

ARP Cache Table

IP address	Mac address	Type
192.168.3.1	0-26-66-4c-e7-94	Complete
192.168.4.2	b4-b5-2f-87-bd-b9	Complete
192.168.3.2	0-26-66-4c-e7-95	Complete
192.168.4.1	50-b7-c3-ad-b3-7d	Complete

Delete

Delete All

ARP Proxy Table

name	IP address	Mac address
------	------------	-------------

Delete

Delete All

Add

OK

4) Host 2 -> Host 1

```

C:\Windows\system32\CMD.exe - ping -t 192.168.1.2
C:\Users\#ProBook4540s>ping -t 192.168.1.2
Ping 192.168.1.2 32바이트 데이터 사용:
192.168.1.2의 : 바이트=32 시간=20ms TTL=128
192.168.1.2의 : 바이트=32 시간=17ms TTL=128
192.168.1.2의 : 바이트=32 시간=30ms TTL=128
192.168.1.2의 : 바이트=32 시간=27ms TTL=128
192.168.1.2의 : 바이트=32 시간=24ms TTL=128
192.168.1.2의 : 바이트=32 시간=21ms TTL=128
192.168.1.2의 : 바이트=32 시간=18ms TTL=128
192.168.1.2의 : 바이트=32 시간=31ms TTL=128
192.168.1.2의 : 바이트=32 시간=28ms TTL=128
192.168.1.2의 : 바이트=32 시간=26ms TTL=128
192.168.1.2의 : 바이트=32 시간=24ms TTL=128
192.168.1.2의 : 바이트=32 시간=22ms TTL=128
192.168.1.2의 : 바이트=32 시간=19ms TTL=128
192.168.1.2의 : 바이트=32 시간=16ms TTL=128
192.168.1.2의 : 바이트=32 시간=29ms TTL=128
192.168.1.2의 : 바이트=32 시간=26ms TTL=128
192.168.1.2의 : 바이트=32 시간=23ms TTL=128
192.168.1.2의 : 바이트=32 시간=21ms TTL=128
192.168.1.2의 : 바이트=32 시간=18ms TTL=128
192.168.1.2의 : 바이트=32 시간=31ms TTL=128
192.168.1.2의 : 바이트=32 시간=28ms TTL=128
  
```

Capturing from 이더넷

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression...

No.	Time	Source	Destination	Protocol	Length	Info
250	114.245540	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
251	114.261675	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
252	115.247545	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
253	115.277279	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
254	116.250638	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
255	116.277308	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
256	117.254186	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
257	117.277381	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
258	118.190081	192.168.4.2	192.168.1.255	TCP	66	4474 → 1688 [SYN] Seq=0 Win=8192 Len=0
259	118.257258	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
260	118.277215	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
261	119.259547	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
262	119.277202	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
263	120.262585	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
264	120.292696	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19
265	121.190283	192.168.4.2	192.168.1.255	TCP	66	[TCP Retransmission] 4474 → 1688 [SYN]
266	121.265288	192.168.4.2	192.168.1.2	ICMP	74	Echo (ping) request id=0x0001, seq=19
267	121.292923	192.168.1.2	192.168.4.2	ICMP	94	Echo (ping) reply id=0x0001, seq=19

> Frame 249: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface 0

> Ethernet II, Src: SamsungE_ad:b3:7d (50:b7:c3:ad:b3:7d), Dst: HewlettP_87:bd:b9 (b4:b5:2f:87:bd:b9)

> Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.4.2

> Internet Control Message Protocol

```

0000  b4 b5 2f 87 bd b9 50 b7 c3 ad b3 7d 08 00 45 00  ../...P. ...}..E.
0010  00 3c 74 20 00 00 80 01 40 4c c0 a8 01 02 c0 a8  .<t ....@L.....
0020  04 02 00 00 4d b4 00 01 07 a7 61 62 63 64 65 66  ....M... ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76  ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67 68 69 2f 35 37 2e 30 2e  wabcdefg hi/57.0.
0050  32 39 38 37 2e 31 33 33 20 57 69 6e 64 6f      2987.133 Windo

```

이더넷: <live capture in progress> | Packets: 267 · Displayed: 267 (100.0%) | Profile: Default

5) Host 1 -> Host 2

```
C:\Users\wydh23>ping 192.168.4.2
```

Ping 192.168.4.2 32바이트 데이터 사용:

```

192.168.4.2의 응답: 바이트=32 시간=33ms TTL=128
192.168.4.2의 응답: 바이트=32 시간=21ms TTL=128
192.168.4.2의 응답: 바이트=32 시간=30ms TTL=128
192.168.4.2의 응답: 바이트=32 시간=33ms TTL=128

```

192.168.4.2에 대한 Ping 통계:

```

패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
최소 = 21ms, 최대 = 33ms, 평균 = 29ms

```


Capturing from 이더넷

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-F> Expression...

No.	Time	Source	Destination	Protocol	Length	Info
913	643.985036	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=543/7938, tt
914	644.808315	192.168.1.2	162.254.195.26	TCP	66	[TCP Retransmission] 27440 → 80 [SYN] Seq=0 win=
915	644.971503	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=544/8194, tt
916	645.001081	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=544/8194, tt
917	645.479664	fe80::c8e5:b80b:995...	ff02::1:2	DHCPv6	157	Solicit XID: 0xb7e4bc CID: 00010001204cd09c00266
918	645.978492	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=545/8450, tt
919	646.000294	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=545/8450, tt
920	646.989882	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=546/8706, tt
921	647.016652	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=546/8706, tt
922	649.480506	fe80::c8e5:b80b:995...	ff02::1:2	DHCPv6	157	Solicit XID: 0xb7e4bc CID: 00010001204cd09c00266
923	649.898245	SamsungE_ad:b6:c9	SamsungE_ad:b6:c9	ARP	42	Who has 192.168.1.1? Tell 192.168.1.2
924	649.898609	SamsungE_ad:b6:c9	SamsungE_ad:b6:c9	ARP	60	192.168.1.1 is at 50:b7:c3:ad:b6:c9
925	650.808726	192.168.1.2	162.254.195.26	TCP	66	[TCP Retransmission] 27440 → 80 [SYN] Seq=0 win=
926	652.013256	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=547/8962, tt
927	652.046433	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=547/8962, tt
928	653.025263	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=548/9218, tt
929	653.047037	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=548/9218, tt
930	654.032581	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=549/9474, tt
931	654.063277	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=549/9474, tt
932	655.046083	192.168.1.2	192.168.4.2	ICMP	74	Echo (ping) request id=0x0001, seq=550/9730, tt
933	655.079104	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) reply id=0x0001, seq=550/9730, tt
934	655.265984	SamsungE_ad:b6:c9	Broadcast	ARP	60	Who has 192.168.1.255? Tell 192.168.2.1
935	657.481079	fe80::c8e5:b80b:995...	ff02::1:2	DHCPv6	157	Solicit XID: 0xb7e4bc CID: 00010001204cd09c00266
936	658.265890	SamsungE_ad:b6:c9	Broadcast	ARP	60	Who has 192.168.1.255? Tell 192.168.2.1
937	659.891170	192.168.4.2	192.168.1.2	ICMP	94	Echo (ping) request id=0x0001, seq=1870/19975, tt

< >

> Frame 869: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface 0

> Ethernet II, Src: SamsungE_ad:b6:c9 (50:b7:c3:ad:b6:c9), Dst: SamsungE_ad:b6:c9 (18:67:b0:ce:f8:82)

> Internet Protocol Version 4, Src: 192.168.4.2, Dst: 192.168.1.2

> Internet Control Message Protocol

```

0000  18 67 b0 ce f8 82 50 b7 c3 ad b6 c9 08 00 45 00  .g....P. ....E.
0010  00 3c 64 44 00 00 80 01 50 28 c0 a8 04 02 c0 a8  .<d.... P(.....
0020  01 02 00 00 53 4d 00 01 02 0e 61 62 63 64 65 66  ....SM.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76  ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67 68 69 65 66 67 68 69 6a  wabcdeghiefghij

```

이더넷: <live capture in progress> Packets: 937 · Displayed: 937 (100.0%) Profile: Default

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.