6일차

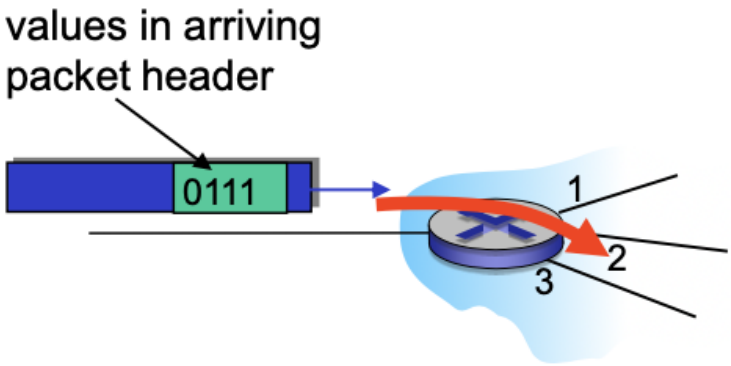
Network Layer

Two key network-later functions

1. network-layer functions
    a. forwarding : router input 에서 적절한 router output로 옮겨줌
    b. routing : 전체길을 보고 최적화된 길을 찾아줌 by routing algorithm

Network layer : data plane, control plane

1. Data plane : datagram을 router 내부에서 어디로(router output) 나갈지 정해줌
    a. forwarding function



    b. local
    c. router 내부에서
2. Control plane : router 경로를 찾아가는 과정
    a. 그래서 network-wide logic 이라고 부르기도 함
    b. routing algorithms
    c. end-end path (source host to destination host)

d. control-plane이 어디있는지에 따라 2가지

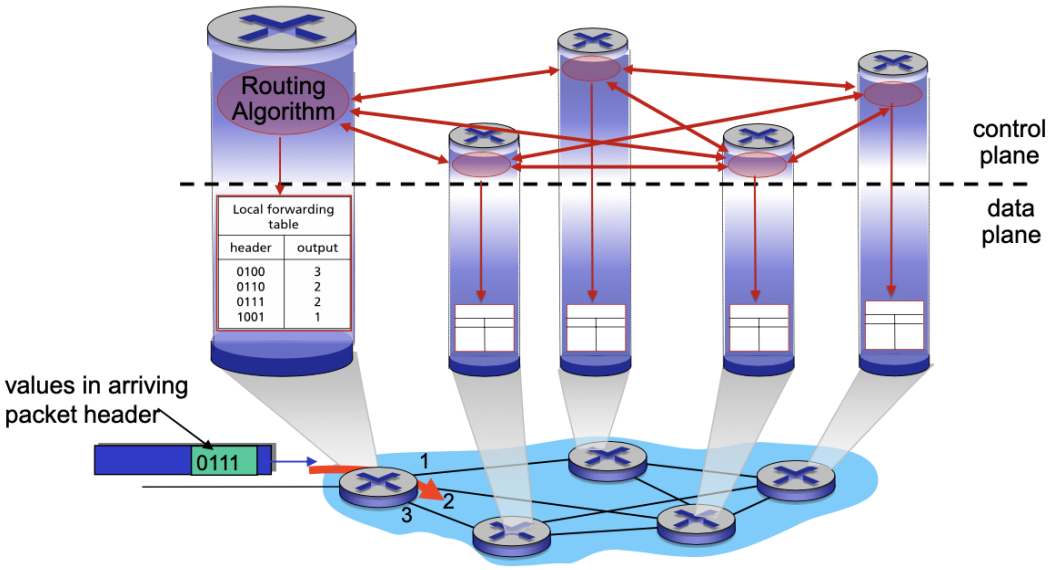
   i. traditional routing algorithms : router 내부

   ii. software-defined networking(SDN) : remote server 안에서 상황에 따라 제어 → 여러 소프트웨어 업그레이드 방식을 바꾸게 됨

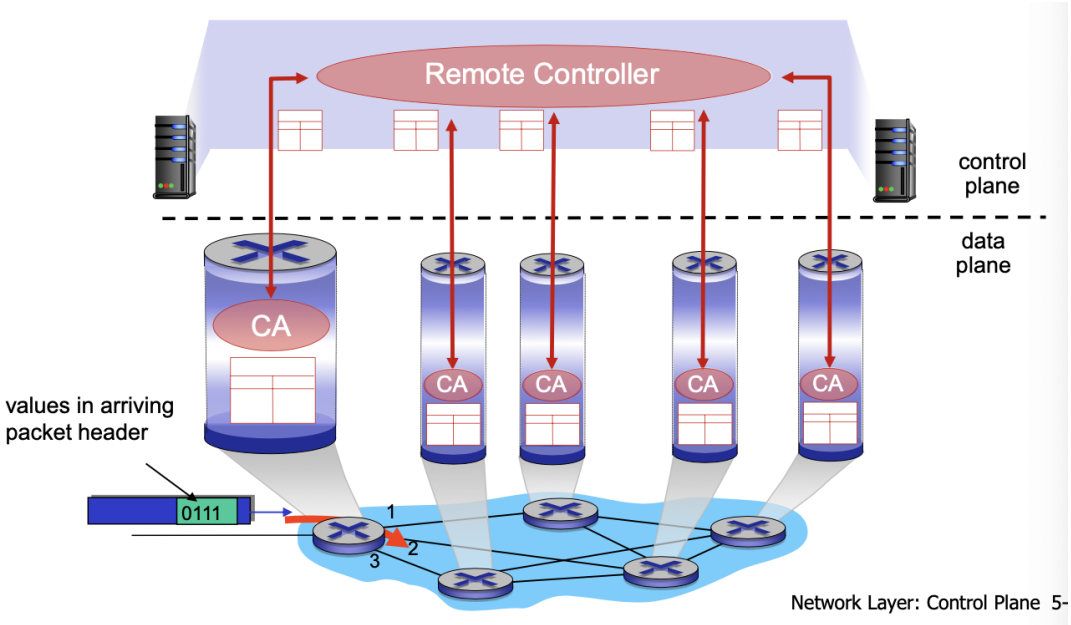
3. 구조

- router 내부에 control plane이 상단 data plane 이 하단에 있음

1. Per-router control plane(routing algorithm)

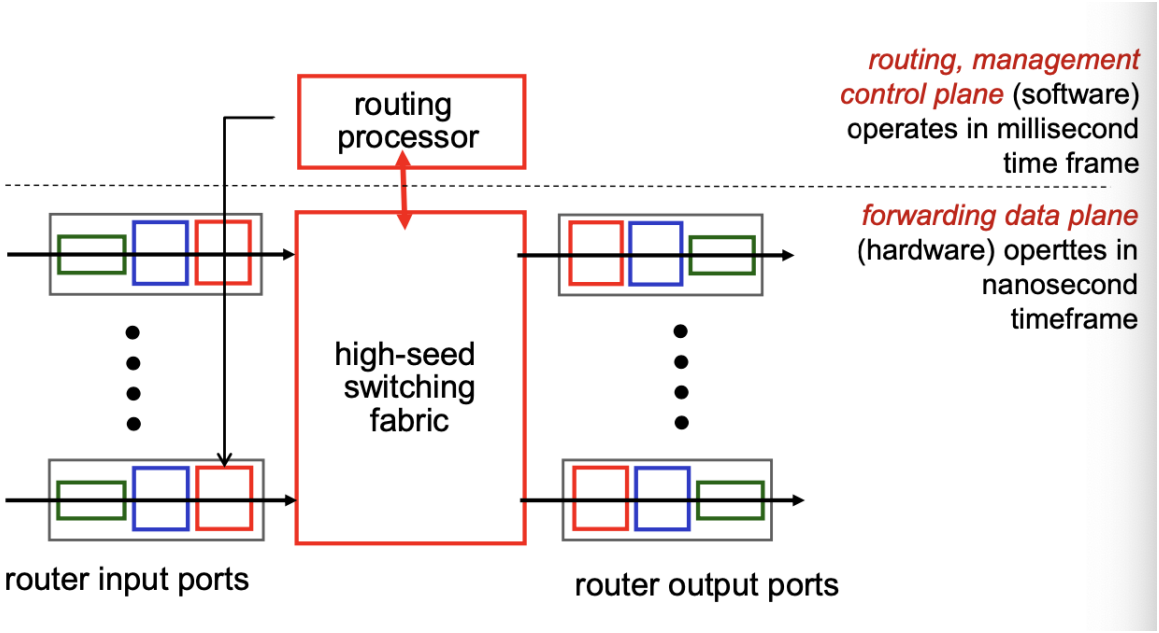


2. Logically centralized control plane(SDN)



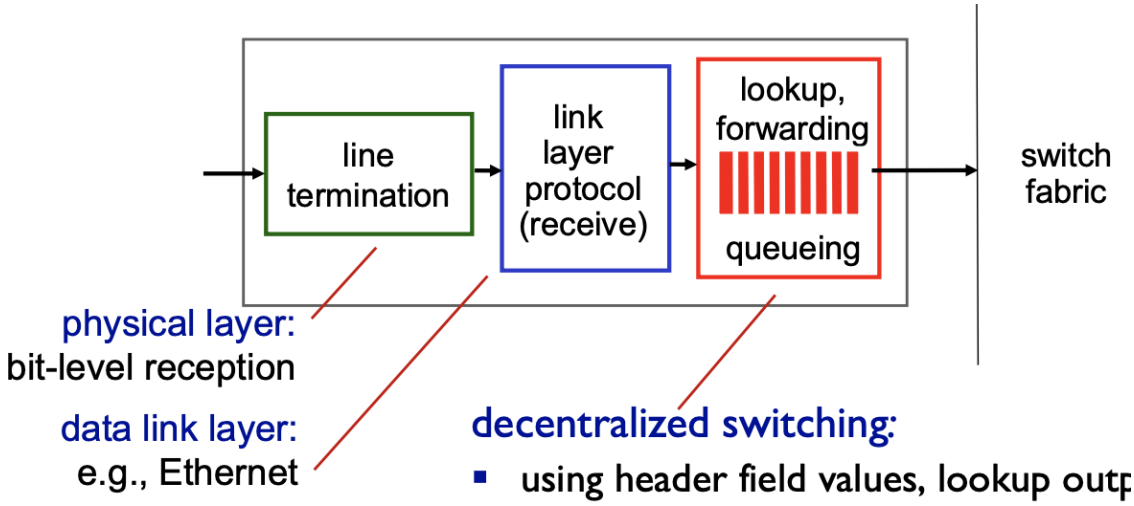
- remote server에서 전체 망의 상황을 보고 라우터 제어를 더욱 효과적으로 해줌

Router architecture overview : router 의 구조를 더 살펴보자



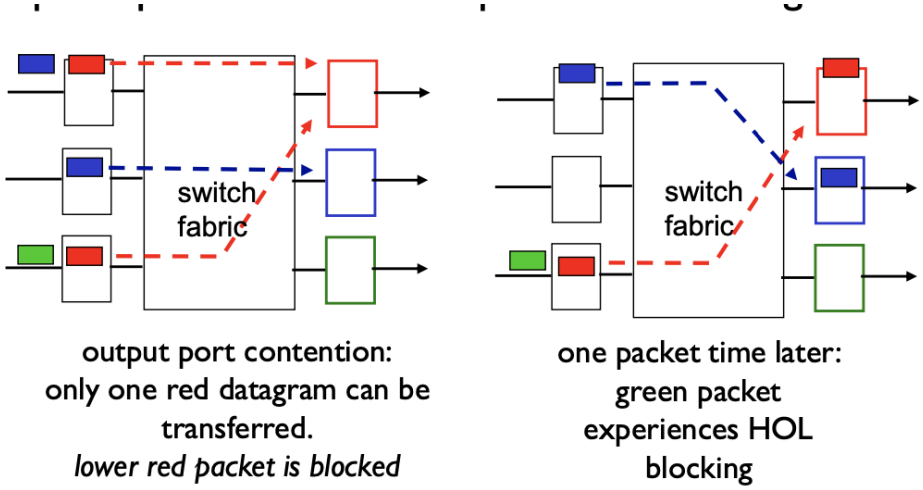
- control plane의 routing processor 에서 routing algorithm 에 의해 switching fabric 안에 forwarding table 을 만들어주고 이 switching fabric 이 input과 output이 적힌 forwarding table을 보고 datagram을 연결해준다.

1. Input port functions



a. phtsical layer(bit 단위) → data link layer(frame 단위) → network layer (datagram 단위)

b. router 의 input port 에서도 queue 가 있다. → 여기서도 HOL blocking 문제가 생김
   → 따라서 switch fabric 을 잘 구성해야 충돌을 줄이고 성능을 높일 수 있다.



- network layer 에서 HOL blocking : queue 앞쪽에 먼저 queued 된 “datagram”이 다른 datagram 이 queue 로 들어 오는 것을 막는다. (HTTP HOL blocking 이랑 구분!)

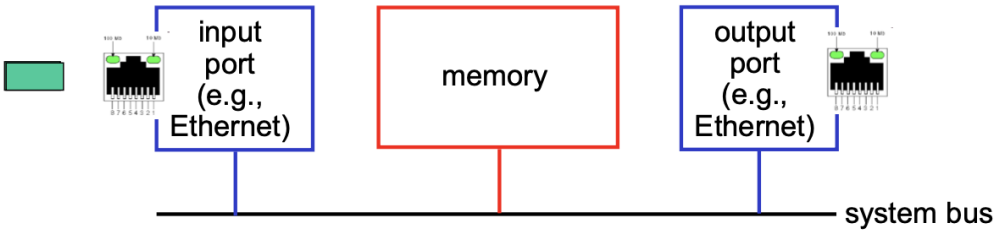
2. Switching fabrics → 충돌의 횟수를 줄여보자!

   a. memory → bus → crossbar **방식으로 개선**

   b. switching via memory

      i. input에서 온 datagram 을 메모리에 적재했다가 output 으로 보내줌

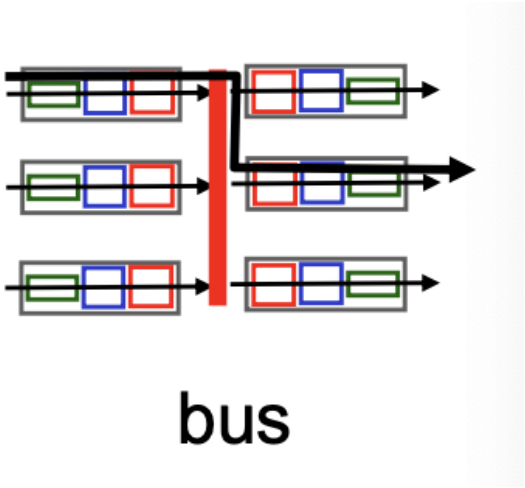
      ii. 메모리에 적재하는 시간 나가는 시간 때문에 bus 방법으로 개선



   c. switching via a bus

      i. input 에서 공유 버스를 거쳐 output 로 보낸다

ii. 공유된 버스를 사용하기 때문에 충돌이 나는 경우가 있다.(속도 저하)
→ crossbar 방식으로 개선

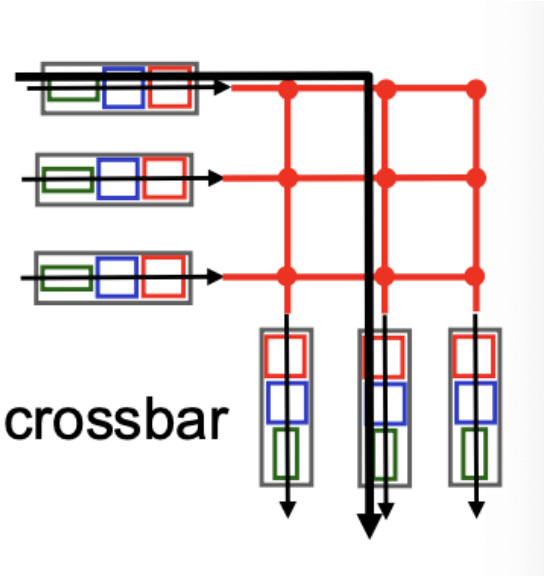


d. switching via interconnection network(crossbar)

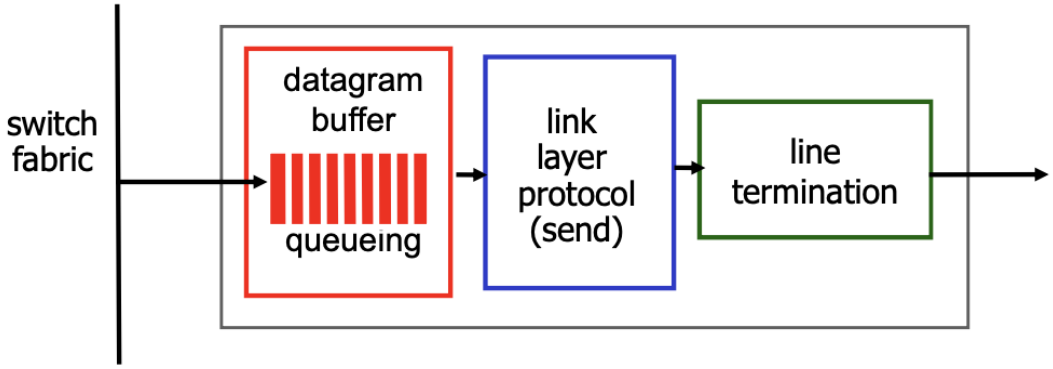
i. 3개의 input 3개의 output 이면 9개의 cell을 만들어서 각자의 길을 갈 수 있게 해줌. → 하지만 비용 증가!!

ii. 비용증가를 해결하기 위한 방법으로 banyan networks(접점의 포인터를 좀 줄여줌) 가 나옴

iii. 버스방식보다 2배 가량 빠름

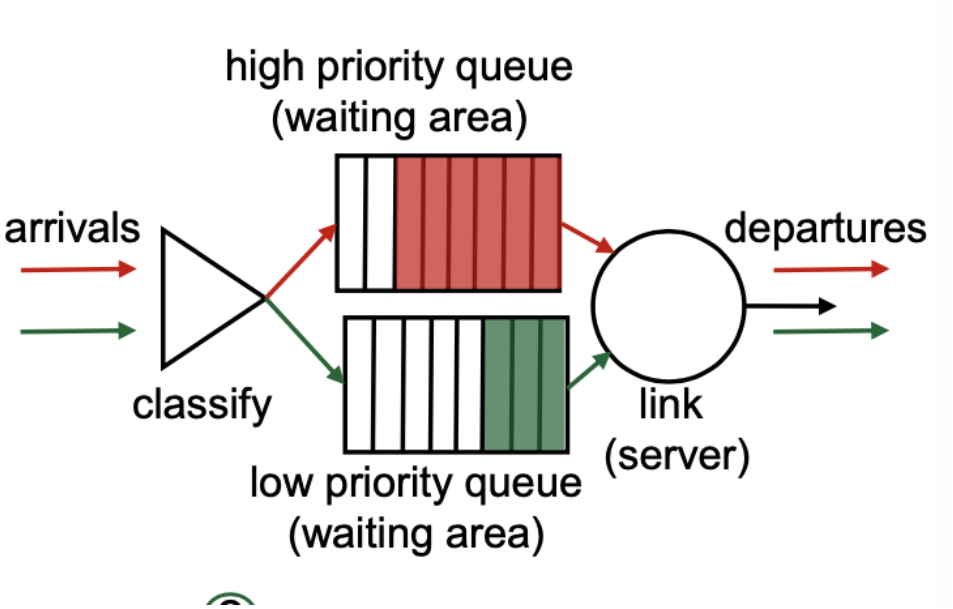


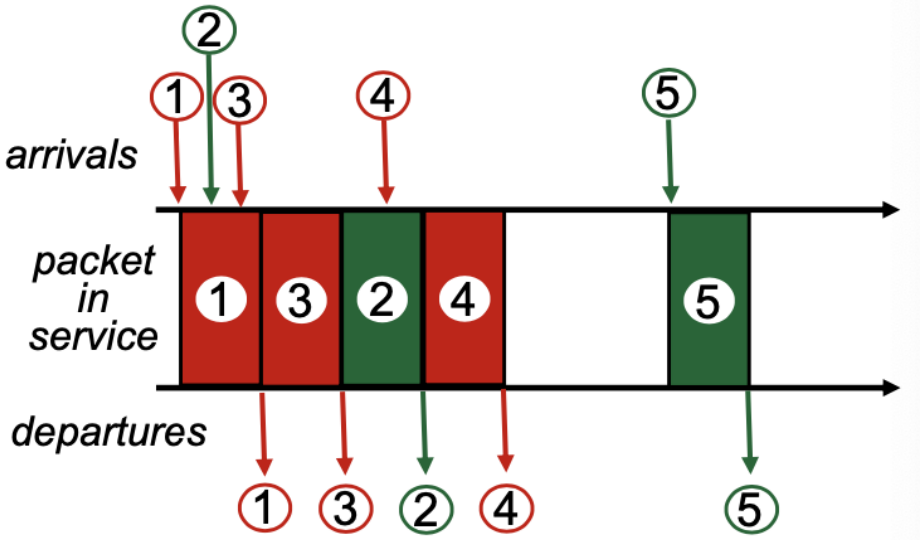
3. Output port functions



- input 과 반대구조, 마찬가지로 datagram 이 queue 에 쌓임
  → 따라서 queue에서 어떤식으로 처리해야 효과적일지 스케쥴링 기법을 생각해보자.
  (링크에 보낼 다음 패킷을 선택하는 과정)

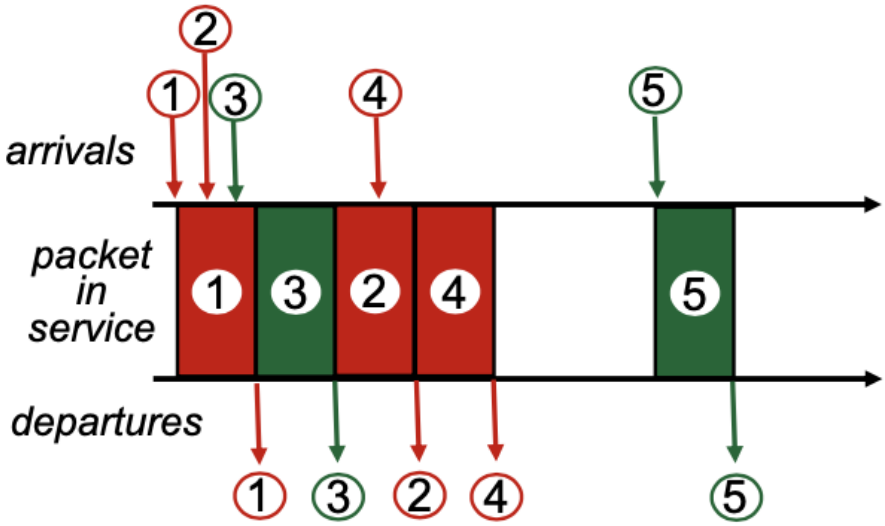
a. Scheduling mechanisms 4가지 방식 (FIFO, priority, RR, WFQ)
    i. FIFO : 순서대로 처리
        i. overflow 가 났을 경우 버리는 순서 : discard policy
            i. tail drop : 도착하는거 버림
            ii. priority : 우선순위 낮은거 버림
            iii. random : 랜덤하게 버림
    ii. priority : 우선순위를 매겨서 처리
        i. 우선순위가 높은, 낮은 큐로 분리
           123이 동시에 도착하면 13먼저 넣고 2넣어줌(우선순위 고려)



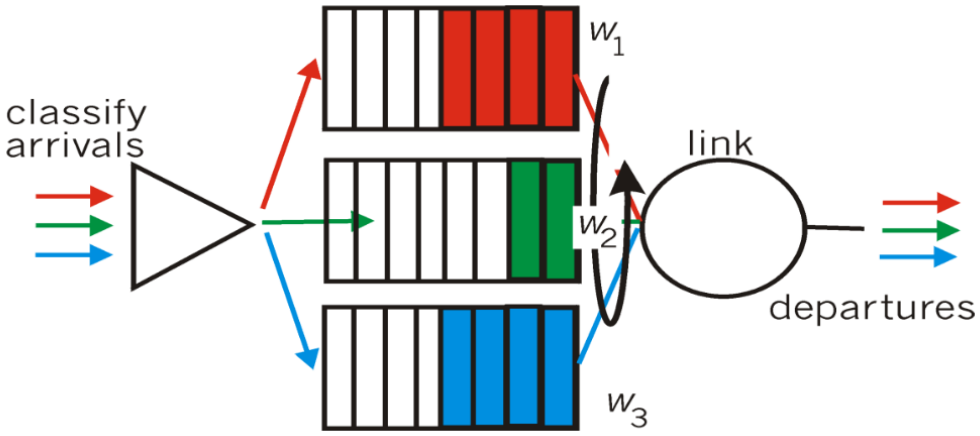


iii. Round Robin(RR) (우선순위 높은 애한텐 손해)

i. 하나씩 번갈아 가면서 처리

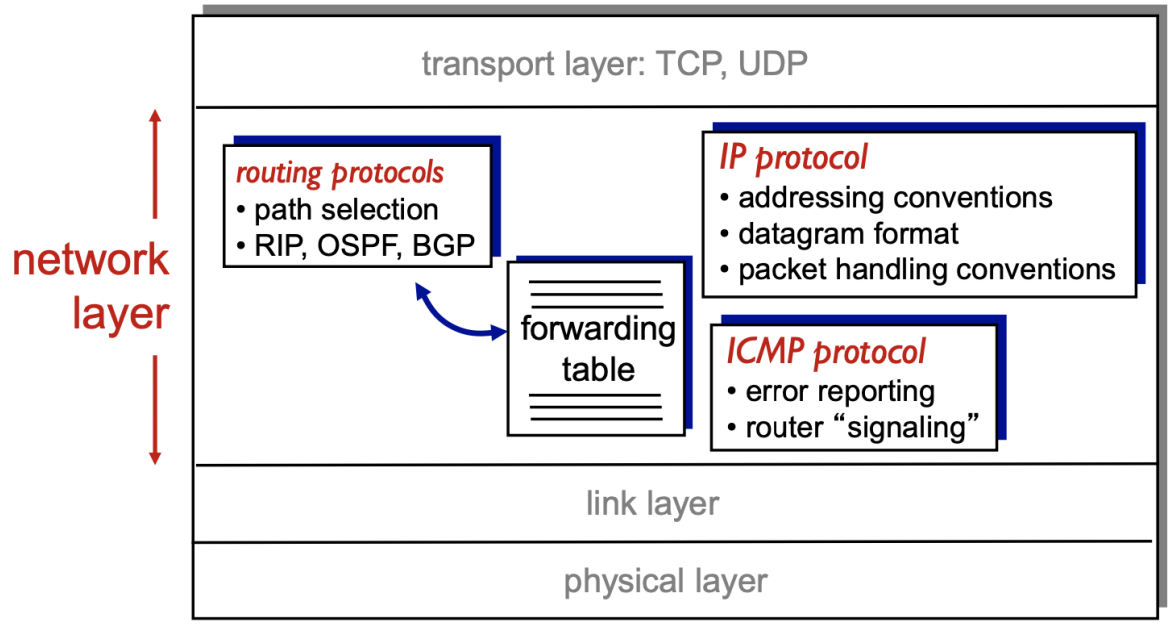


iv. WFQ(Weighted Fair Queuing) → 비중을 둬서 (예를들어 5:2:3 이런식)

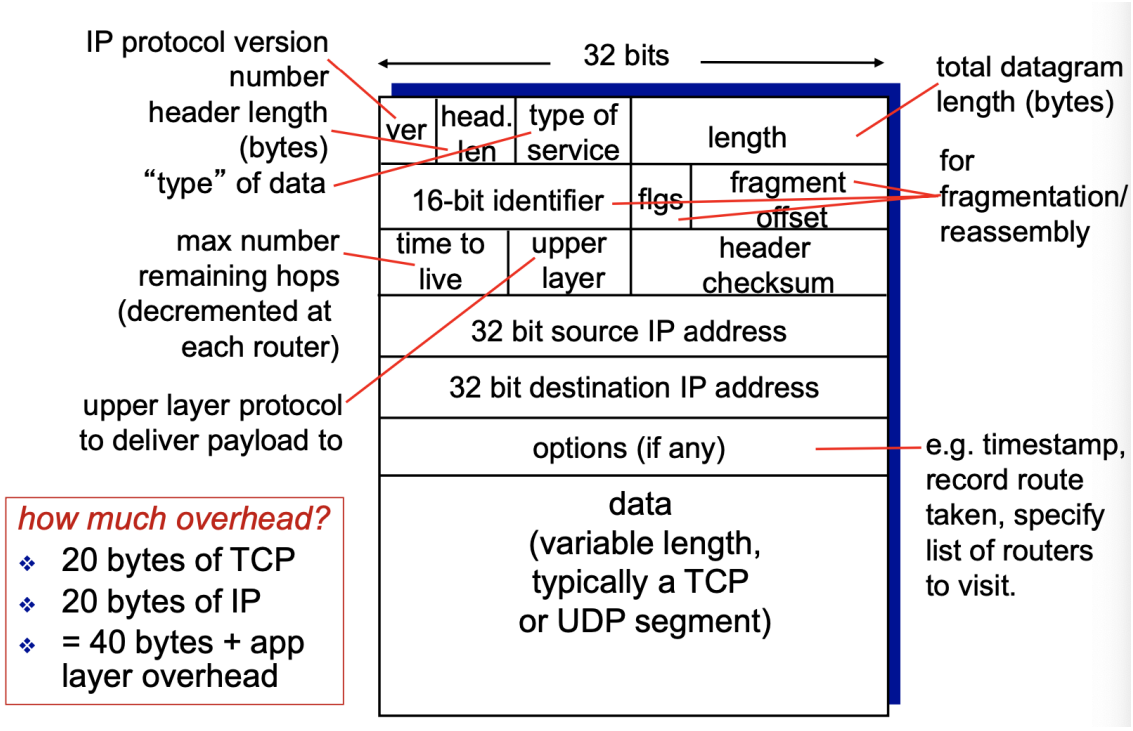


1. RoundRobin에 기반하지만 우선순위에 따른 비중을 두어서 처리

The Internet network layer



1. 종류
    a. routing protocols : 앞에서 한거
    b. IP protocol
    c. ICMP protocol : error 관련
2. IP datagram format



a. head len : 20byte

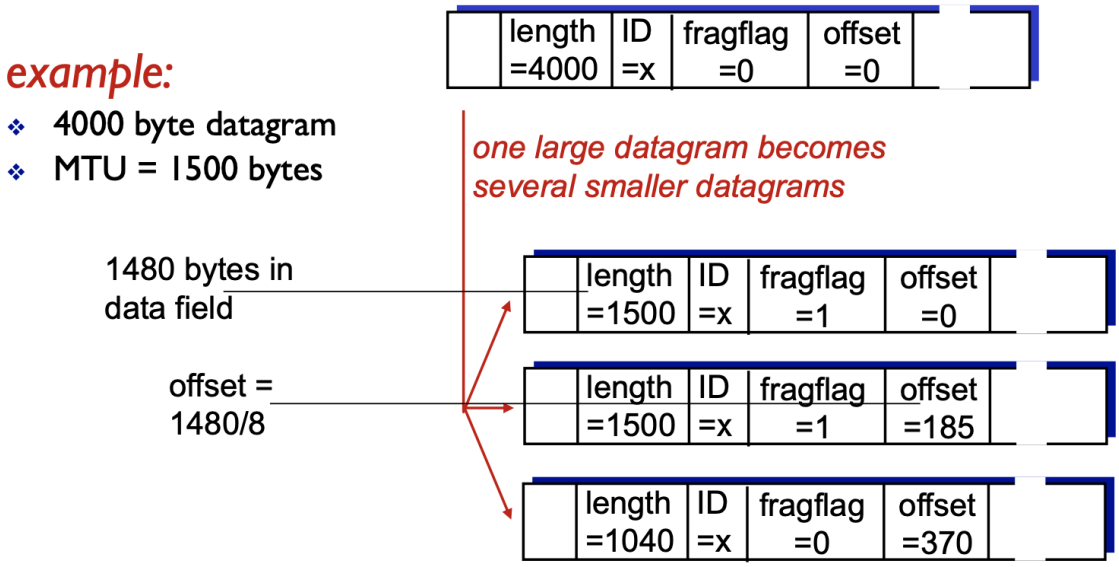
b. time to live (ttl) : 각 router 를 거칠때마다 1씩 감소, 0이 되면 packet 을 버림
   → 잘못된 router 로 들어갔을 경우 두지 않고 없애 주기 위해서

c. length : header 를 포함한 전체 datagram 길이(bytes)

d. 특별히 16-bit identifier / flags / fragment offset가 있음
   → datagram이 큰 경우 쪼개거나 합치는데 사용
   16-bit identifier : 쪼개지는 경우 같은 id를 가져야 하나에서 쪼개졌다는 것으로 구분 가능
   flags : fragment를 한건지 안한건지 했다면 시작인지, 중간인지, 끝인지 (3bit정도)
   fragment offset : 쪼개진 녀석의 정보 (16-3이므로 13비트)

초기화 비트 등 많이 쓰이는 비트는 알아두자!

3. IP fragmentation, reassembly



a. ID를 동일하게 해줌

b. flag 1 offset 0 → 시작

   flag 1 offset 숫자 → 진행

   flag 0 offset 숫자 → 끝

c. offset 은 datafield를 8로 나눠서 씀 (flag가 3비트이므로) 위치를 찾아주기위해서

d. 맨위에건 header 20byte 하나면 되는데

이를 3개로 쪼개면 쪼개지는만큼 각각 20byte header 가 필요하다.

그림은 length 안에 header 20이 포함되어있음

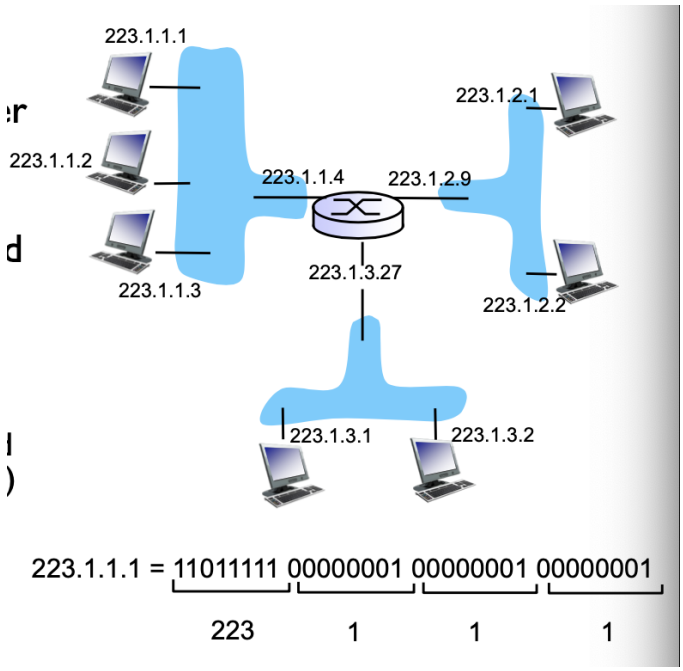
e. MTU (max transfer size)가 1500 bytes 이므로 하나의 fragment 의 길이는 최대 1500

따라서 기존 datagram header 20제외 하면 4000-20 = 3980byte
첫번째 1500 - 20 = 1480
두번째 1500 - 20 = 1480
세번째 1040 - 20 = 1020
1480 + 1480 + 1020 = 3980

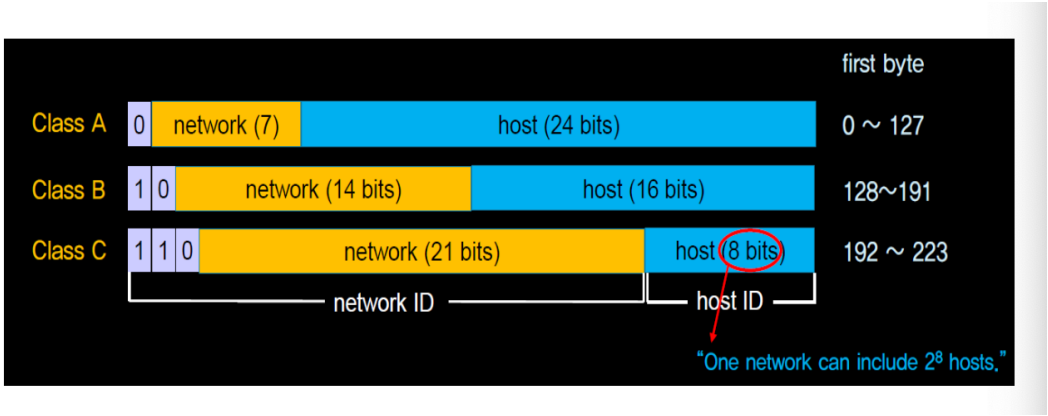
IP addressing

1. 개념

a. IP address : 32bit로 이루어진 host, router 의 식별자



b. 주소체계가 만들어진 과정 : ICANN 에서 ip 정보를 정의 해줌 (표준화)



i. 과거에는 이런식으로 나누었는데 예를들어 class C는 2의 8승 256개의 ip 정도 밖에 나눌 수 없었다. 현재는 host 가 아주 많아져서 이런 문제점을 보완하기 위해 다른 개념을 쓴다!

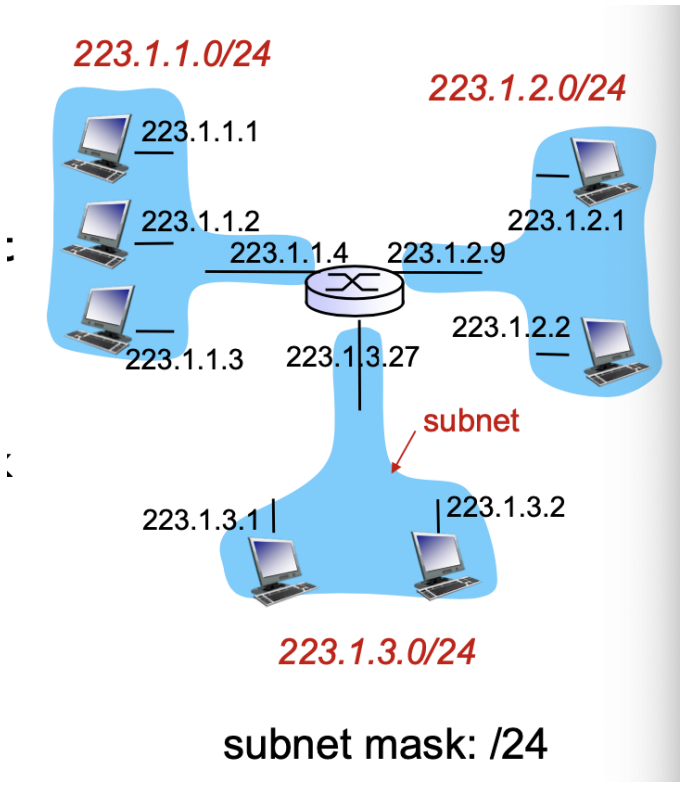
2. Subnets

a. 개념

i. router를 통과하지 않고 내부에서 물리적으로 연결된 device interface

ii. subnet part - 같은 subnet 에 있는 device 들은 공통된 상위 bit 를 가짐

iii. host part - 남은 bits 는 host 구분



iv. subnet mask : 223.1.3.0을 예로 들때 subnet mask 가 24면 24까지를 network id(subnet part)로 본다는 것이다. 따라서 223.1.3 까지는 동일하고 남은부분만 host part 가 되는 것이다. 이 그림에서는 3개의 subnet으로 구성되어 있다.
223.1.1.0/24
223.1.2.0/24
223.1.2.3/24 로 표현

3. IP addressing : CIDR (Classless InterDomain Routing) (사이다라고 부름)
   a. classless 라는 개념이 중요
   → 왜 classless 라고 부를가? : class는 subnet mask를 8,16,24,32만 가능한데 subnet portion 이 임의의 길이를 가지도록 하여 host 를 더 많이 표현 할 수 있다.
   a.b.c.d/x 로 표현 (x는 subnet portion)
   b. subnet과 supernet을 합쳐서 만든 프로토콜
   subnet : network id 를 늘리고 host id 를 줄이는 개념
   supernet : network id 를 줄이고 host id 를 늘리는 개념
   c. Destination-based forwarding
      i. 개념 : destination address를 보고 forwarding table 에 속해있는 곳을 찾아서 목적지를 찾아줌

forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through **11001000 00010111 00010111 11111111**	0
11001000 00010111 00011000 00000000 through **11001000 00010111 00011000 11111111**	1
11001000 00010111 00011001 00000000 through **11001000 00010111 00011111 11111111**	2
otherwise	3

   d. Longest prefix matching

i. 개념 : 가장 긴 주소의 prefix를 보고 맞는 destination address를 찾아주는 것

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

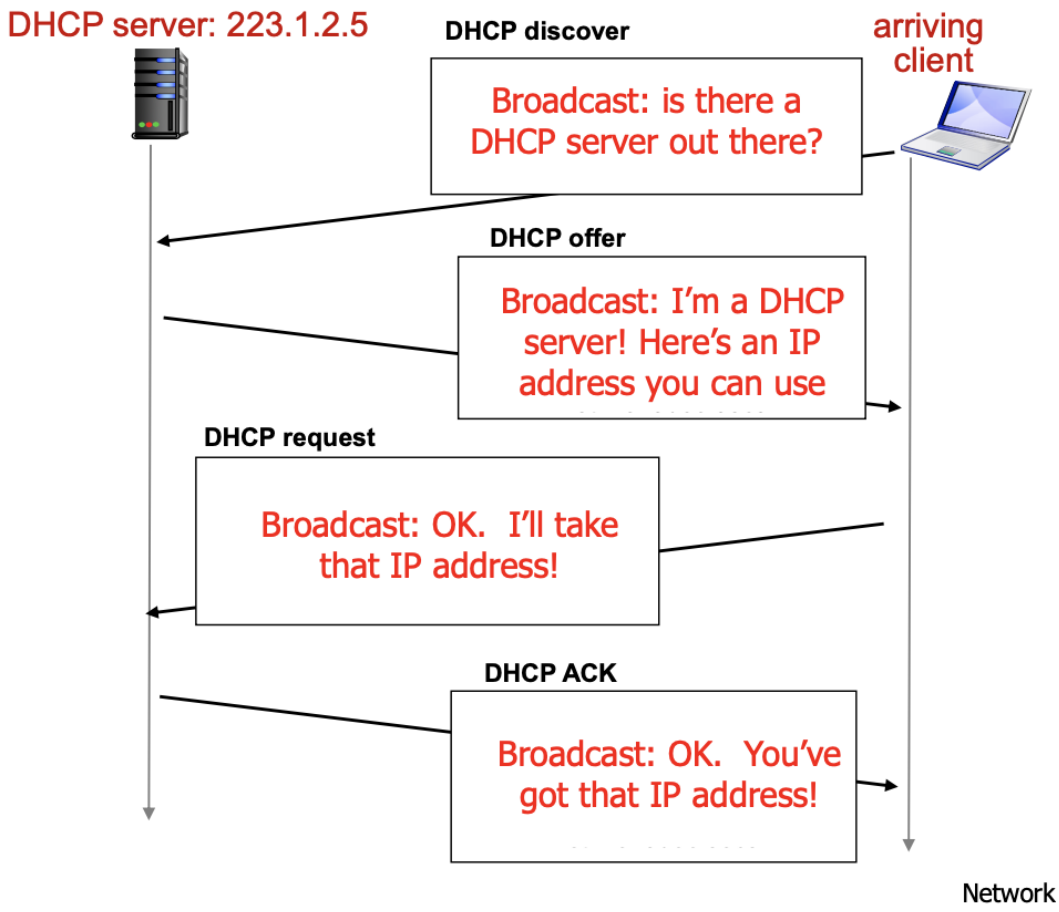
examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

→ longest 이므로 1,2를 비교해보면 11000 뭐시기인건 1,2둘다 되겠지만 1로 보내는 것이다.

4. DHCP : Dynamic Host Configuration Protocol (IP 동적할당)
   a. host 가 네트워크에 접속할때 server에 IP address를 달라고 한다.
   b. 이때 IP address 만 할당 해주는 것이 아니라 first-hop 의 router 도 같이 준다.
      → 동적할당 받은 ip 이기 때문에 다음 라우터는 어디로가라~ 를 알려주는 식으로 최적화 해준다. (hop : 라우터와 라우터 사이 거리 first hop : 가장 가까운 라우터)


DHCP server: 223.1.2.5
DHCP discover
arriving client
Broadcast: is there a DHCP server out there?
DHCP offer
Broadcast: I'm a DHCP server! Here's an IP address you can use
DHCP request
Broadcast: OK. I'll take that IP address!
DHCP ACK
Broadcast: OK. You've got that IP address!
Network