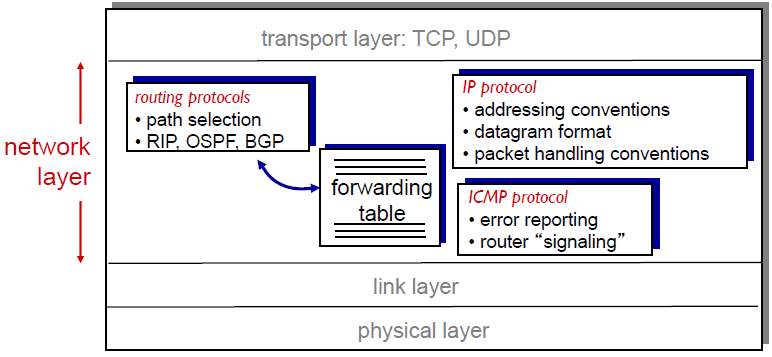
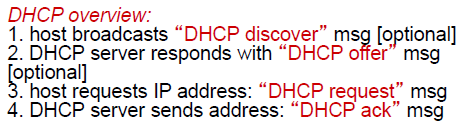
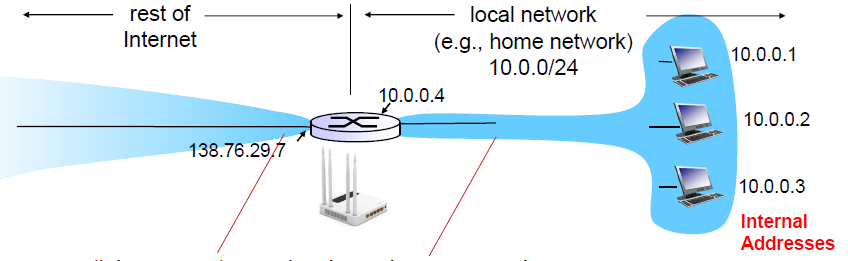
**Computer Network  
Chapter 4: Network Layer – part 3**

**IP Internet Protocol**  
 **Internet network layer  
  
  
  
  
  
  
  
  
  
 IP datagram format**  
 IP: Internet Protocol  
 IP header: IPver – IP protocol version number, head length - 32bit (4bytes) \*5 = 20bytes + a.length = datagram의 전체 길이  
 IP header에는 출발지의 IP address와 목적지의 IP address가 들어간다., IP data에는 TCP의 헤더가 들어간다.  
 **IP fragmentation, reassembly** – MTU (max transfer unit(size)): 링크종류마다 최대 보낼 수 있는 패킷의 크기가 정해짐 – 만약 처음 링크MTU가 4000B, 다음 링크의 MTU가 1500B이면 그 패킷은 갈 수 없다. 🡺 MTU의 한도 때문에 패킷을 쪼개야한다.  
 **Fragmentation**: 패킷을 쪼갬  
 **Reassembly**: 쪼개진 패킷을 받고 다시 합침. 🡪 합치는 이유는 패킷이 쪼개진 상태로 들어오면 패킷의 header에 있는 TCP가 쪼개진 것 중에 한 곳에만 들어가 있어서 받는 쪽에서 그 packet을 보고 ack을 보낼 수 없고 정확한 정보를 읽을 수 없다.  
  
  
  
  
  
 **IPv4 addressing**  
 **interface**: 네트워크가 연결 된 지점 (LAN Port 🡺 사실 호스트마다 IP address를 가지는 것이 아니라 각각의 interface마다 IP address를 가지고 있다.)  
Router는 기본적으로 여러 개의 interface를 가지므로, 여러 IP address를 가지고 있다. – 유선은 스위치로 연결, 무선은 WIFI AP로 연결  
 **IP address**: 32bit로 232로 약 42억개의 주소가 있다. IP주소는 subnet ID와 host ID를 가진다.  
 **Subnet**: **동일한 subnet ID**를 가진 애들의 모임 – 같은 네트워크를 쓰는 애들 == 같은 서브넷을 쓰는 애들 🡪 같은 subnet내에서는 라우터 없이 연결할 수 있다.  
 **Subnet mask**: IP주소의 왼쪽 24bit가 subnet mask이다 /24로 표현 – 그 24bit가 같으면 같은 subnet의 속한 것 // 나머지 오른쪽 8bit는 host ID 🡺 11111111 11111111 11111111 00000000 = 255.255.255.0 , 255.255.254.0이면 23bit가 subnet mask  
 **CIDR**: 위와 같은 방식 – Classless InterDomain Routing  
a.b.c.d/x 🡪 x가 subnet mast의 bit, **subnet ID랑 prefix**는 같은 개념  
Router는 IP주소의 prefix (subnet ID)를 보고 방향을 결정 (forwarding table에는 42억개의 IP주소를 저장해야 하는데, 너무 많기 때문에 prefix로 묶어 놓는다. 그래서 forwarding table에는 prefix가 들어가 있다.) 🡪 검색이 빨라짐 // prefix가 아닌 부분은 host ID  
 **Address allocation with CIDR** – 소규모 네트워크는 필요한 host ID 수가 적기 때문에 prefix를 길게한다. (subnet mask를 크게) /// 대규모 네트워크는 필요한 host ID수가 많기 때문에 prefix를 짧게 한다. (subnet mask를 작게)  
 --CIDR의 Flexible하기 subnet addressing할 수 있다.  
 **DHCP**: IP주소를 자동으로 할당해 주는 것 – DHCP로 설정 된 IP주소 및 subnet mask등등은 임대방식으로 부여된다. 임대기간이 끝나면 IP address는 반환되며 user는 다시 DHCP로 임대 받는다.  
새로 들어온 host는 IP주소도 없고, DHCP server도 모른다. 🡪 host가 들어오면 우선 DHCP server를 찾아야한다 ///   
  
  
  
  
  
  
  
  
  
  
  
  
우선 client는 접속한 서버에 DHCP server가 있는지 없는지 **DHCP discover**를 Broadcast로 보낸다. (누가 DHCP server인지 모르기 때문에 broadcast로 주변에 다 보낸다, 목적지는 255.255.255.255라는 broadcast IP address를 적고, 출발지는 0.0.0.0으로 적고, Transaction ID를 적는다.) 🡪 DHCP server는 **DHCP offer**를 broadcast로 보낸다. (다른 client는 자신이 discover을 보내지 않았기 때문에 무시, discover을 보냈던 client만 응답 받는다 – 그 응답에 Transaction ID가 적혀 있어서 client는 자신이 보낸 것인지 알 수 있다.) 🡪 client는 **DHCP request**를 broadcast로 보낸다. (이 때는 DHCP가 여러 개 일수도 있기 때문에, 자신이 어떤 DHCP를 선택했다고 다른 DHCP서버들에게 알리기 위함) 🡪 최종적으로 DHCP server는 **DHCP ACK**을 broadcast로 보낸다 (client의 주소가 아직 안 정해졌기 때문에) 그 ACK에는 yiar (your IP address)가 있기 때문에 client는 그것을 받고 설정  
🡺 DHCP로 IP address뿐만 아니라, local DNS server정보, first hop router주소 등등을 알 수 있다., DHCP를 사용함으로써 관리자도 그 한정된 아이피 개수를 효율적으로 사용할 수 있게 된다.  
  
 **NAT (network address translation)**internal IP주소는 내부 아이피주소로 외부로 나가면 사용불가, 내부 네트워크에서만 사용가능 🡪 **NAT장비로 internal address를 external address로 변환해준다**.  
host가 특정 목적지에 datagram을 전송할 때 자신의 IP address와 port#을 적어준다. NAT은 그 IP와 port#을 공용IP address로 변환시키고 포트번호는 다른 host와 겹치기 때문에 구분할 수 있도록 변환하여 목적지에 보낸다. (목적지에서는 NAT이 그 datagram을 보냈다고 착각하여 응답을 NAT한테 보낸다) NAT은 그 응답의 목적지 IP주소를 보고 자신의 NAT translation table을 보고 그 아이피에 해당하는 내부 아이피 주소로 보낸다.  
NAT의 port#은 host와 application의 connection 하나하나 모두 구분한다. – 이론적으로 NAT은 port number가 16bit이므로 60000개정도 사용할 수 있다.  
  
 내부 아이피가 외부 아이피 호스트로 먼저 요청할 때는 문제 없는데, 외부 아이피 호스가 내부아이피 host로 먼저 요청할 때는 문제발생 (NAT translation table에 정보가 없어서…) ex) torrent는 internal ip 주소만 알기 때문에…  
🡪 그래서 port forwarding, DMZ등으로 해결  
   
 **IPv6** 128bit…2128=3.4\*1038, 16bytes이므로 쓰기 쉽게 16진수이용  
 절반이 subnet id(prefix) / host id  
 IPv6은 수동할당 할 수 없기 때문에, 자동으로 할당한다.  
 1단계에는 **link local address**: 같은 subnet에서만 사용할 수 있는 link local address라는 것을 host가 직접 설정함. Host id부분에는 mac address를 입력 // 이 주소는 local network에서만 이용가능 🡪 2단계는 router의 도움으로 (router가 prefix를 알려줌) **global prefix**를 설정  
  
 **Transition from IPv4 to IPv6**기존의 전세계 망은 IPv4로 제작됨. 그래서 갑자기 모두 IPv6를 쓰는 flag days가 없다.  
🡪 기존에 IPv4가 있는 상황에서 IPv6를 deploy해야한다. 🡪 IPv6라우터들은 IPv4의 패킷을 이해해야한다. (IPv4, IPv6 모두 이해하도록 / IPv4는 IPv6를 이해못함) 🡺 Tunneling: IPv6의 패킷을 encapsulation시키고 IPv4의 헤더를 덫붙인다. 그래서 IPv4라우터들이 IPv6의 패킷을 이해함. 그 후 그 패킷을 IPv6가 받으면 다시 decapsulation시켜서 IPv6패킷을 본다. (IPv6라우터가 다음 링크가 IPv4일 때 tunneling)