네트워크 계층 (1, 2)

URL

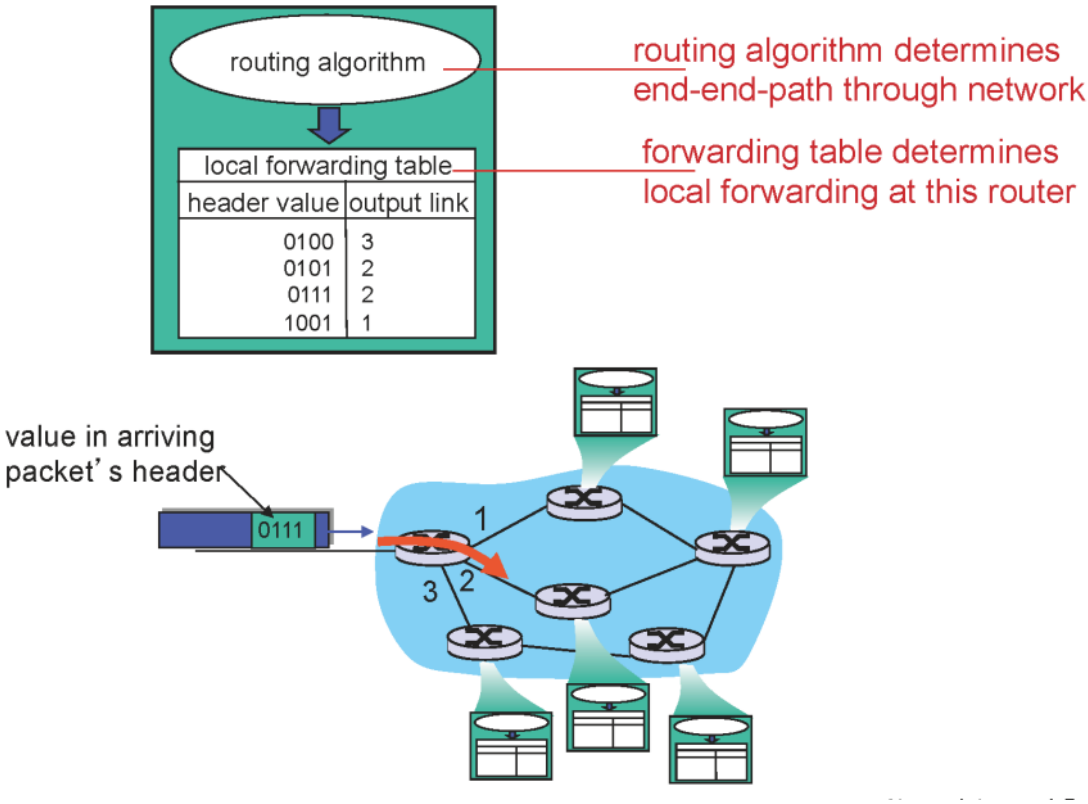
▼ 목차

Network Layer: Intro

Network Layer

- source부터 destination까지 패킷을 어떻게 잘 보낼 것인가 -> **Router!**

Two key network-layer functions



Forwarding

- 들어온 패킷의 목적지 주소와, **_Forwarding Table_**의 엔트리를 매칭해서 엔트리에 해당하는 링크로 보내는 작업

Routing

- Routing algorithm을 활용해 forwarding table을 만드는 작업

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

- IP주소가 워낙 많다보니 특정 IP주소가 아니라,

IP 주소 범위

로 테이블을 정의한다.

Longest Prefix Matching

Longest prefix matching

longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

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

examples:

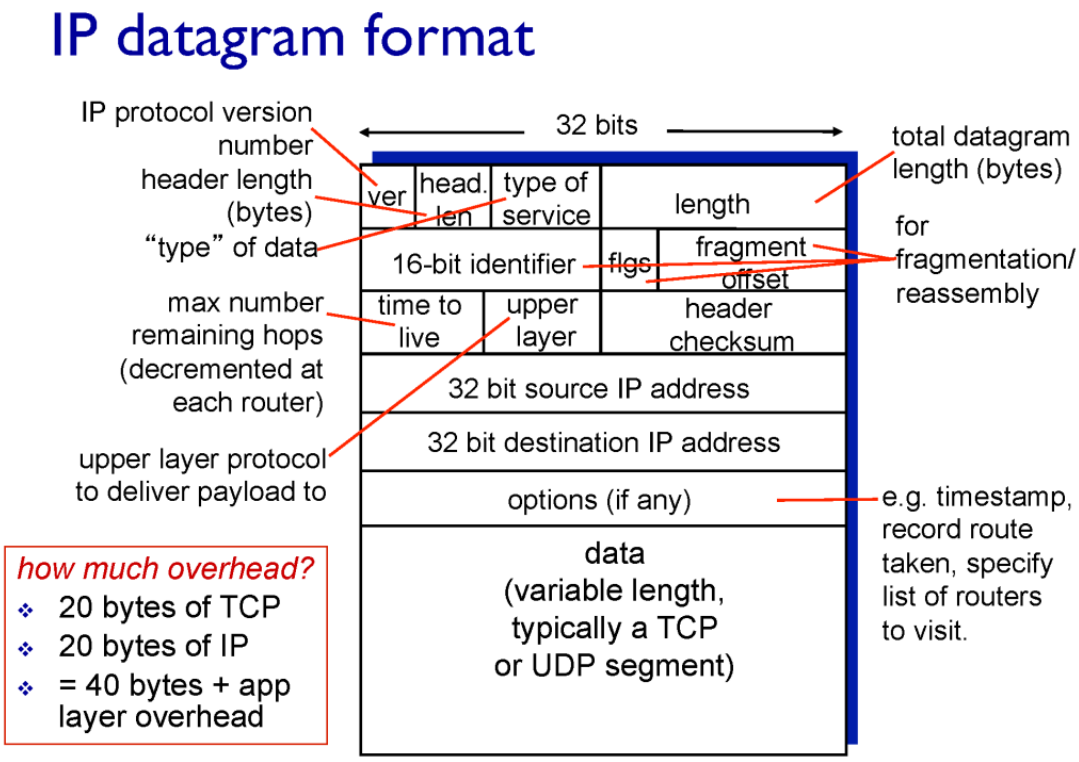
DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

- 가장 길게 일치하는 엔트리와 매칭 시킨다.

IP: Internet Protocol

IP datagram(패킷)의 포맷



length

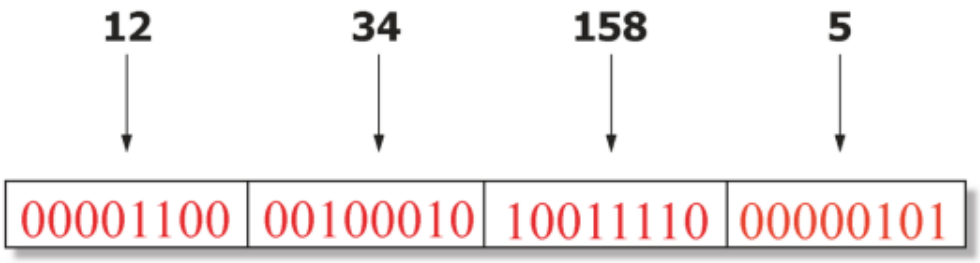
- 패킷의 길이
  - **time to live**
- 라우터를 지날 때마다 1씩 감소하다 0이 되면 패킷이 사라진다
- 라우팅 테이블에 오류가 생겨서 루프가 도는 경우 방지
  - **upper layer**
- 상위 레이어 (transport layer)의 전송 방식(tcp / udp)
  - **source/destination IP address**
- 출발지/목적지의 IP주소
- **IP에서 가장 중요한 필드!**

- TCP의 헤더 20 bytes + IP의 헤더 20 bytes
- 패킷은 기본적으로 40바이트의 헤더를 가지고 데이터는 그 나머지이다.
- 그런데 상당수의 패킷은 40바이트밖에 되지 않는데 이는 ACK 패킷이다!

IP Address(IP v4)

IP Address (IPv4)

- A unique 32-bit number
- Identifies an interface (on a host, on a router, ...)
- Represented in dotted-quad notation



✔️ 32비트의 수

✔️ 사람이 읽기 위해 8비트 단위로 끊어서 10진수로 표시

✔️

Network Interface를 지칭하는 주소

✔️ 여러 개의 네트워크 인터페이스를 가지면 ip주소도 여러개이다.

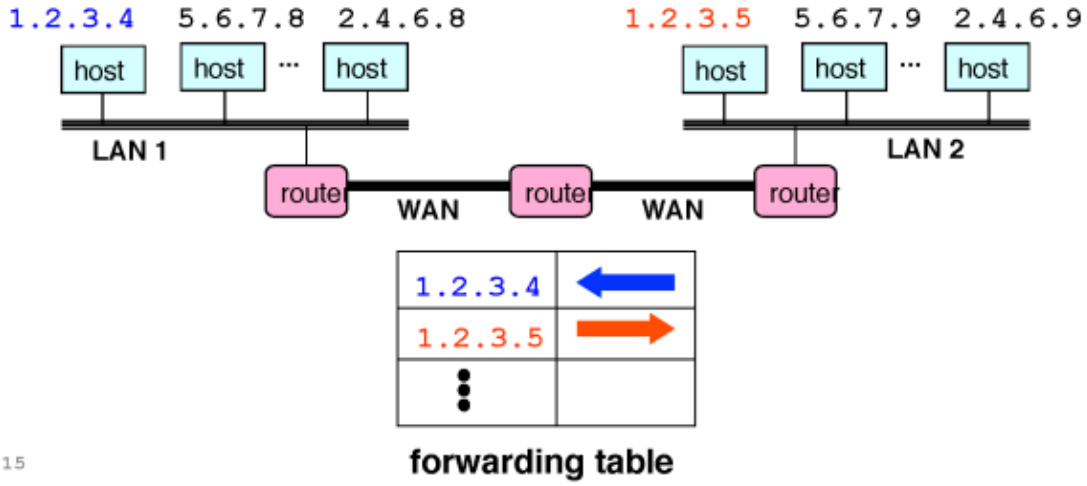
(ex: router)

IP 주소 배정

무작위로 배정하게 된다면...?

Scalability Challenge

- Suppose hosts had arbitrary addresses
  - Then every router would need a lot of information
  - ...to know how to direct packets toward every host



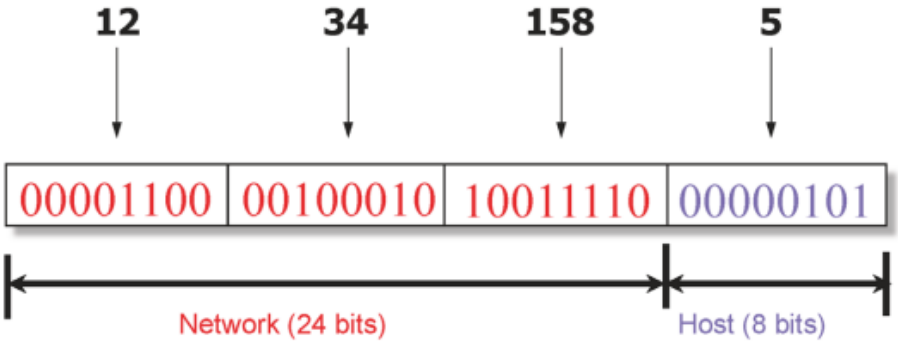
15

- 무작위로 배정하면 라우팅 내부의 forwarding table의 크기가 너무 커진다!

계층적 구조 (Hierarchical Addressing)

Hierarchical Addressing: IP Prefixes

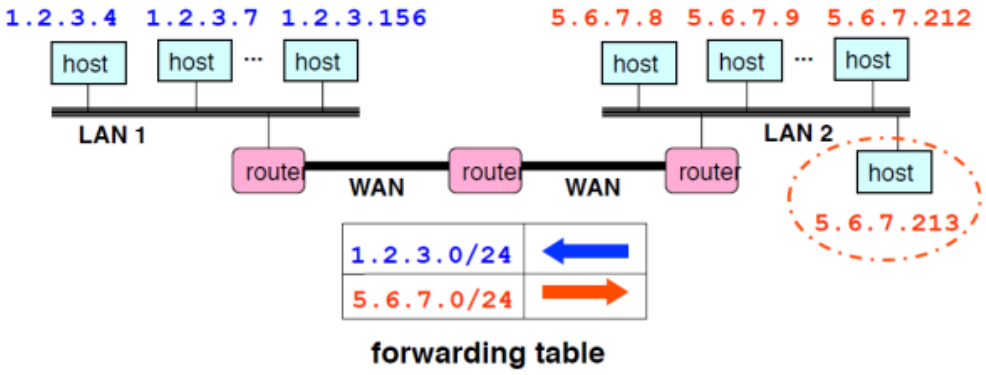
- Network and host portions (left and right)
- 12.34.158.0/24 is a 24-bit **prefix** with 2^8 addresses



- IP주소를 **네트워크 아이디(prefix), 호스트 아이디** 로 구분하는 계층적 구조
- **앞부분: 네트워크 아이디 (prefix, subnet id...)**
- **뒷부분: 네트워크에 속한 호스트 아이디**

Easy to Add New Hosts

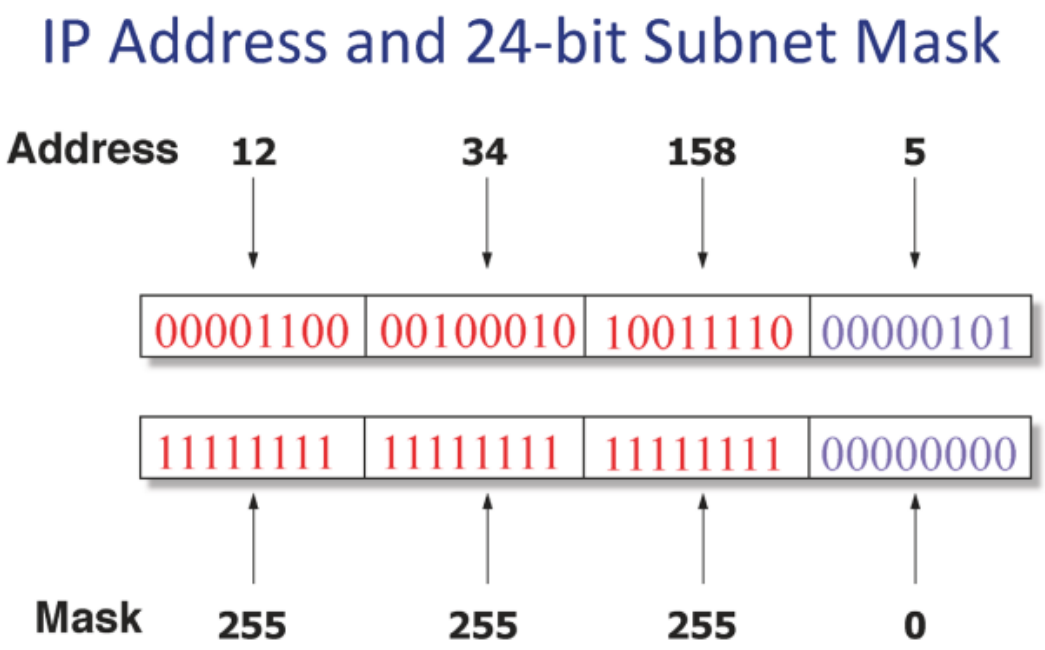
- No need to update the routers
  - E.g., adding a new host 5.6.7.213 on the right
  - Doesn't require adding a new forwarding-table entry



- 같은 네트워크에 속하는 호스트들은 같은 네트워크 아이디를 가지게 된다.

- forwarding table도 단순해지고 새로운 host를 추가해지는 것도 용이해진다.

Subnet Mask



- 기계가 처리하기 쉽게

bit로 prefix와 호스트 아이디를 구분해서 나타낸 체계

History of IP Addressing

Classful Addressing

Classful Addressing

- In the old days, only fixed allocation sizes
  - Class A: 0*
    - Very large /8 blocks (e.g., MIT has 18.0.0.0/8)
  - Class B: 10*
    - Large /16 blocks (e.g,. Princeton has 128.112.0.0/16)
  - Class C: 110*
    - Small /24 blocks (e.g., AT&T Labs has 192.20.225.0/24)
  - Class D: 1110* for multicast groups
  - Class E: 11110* reserved for future use
- This is why folks use dotted-quad notation!

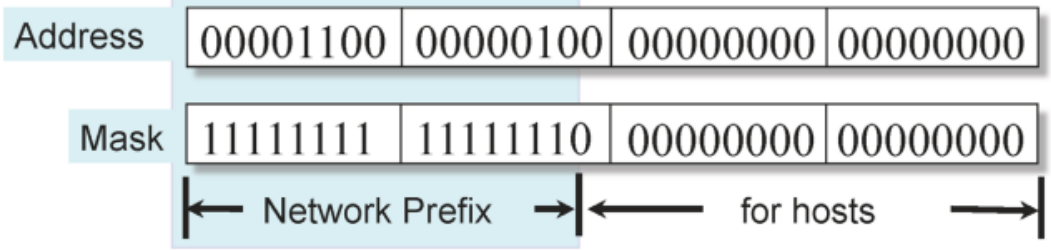
- 과거에는 IP 주소를 클래스로 구분했음
- 클래스 배분의 문제(비효율, 불공평) 발생

Classless Inter-Domain Routing (CIDR)

Classless Inter-Domain Routing (CIDR)

Use two 32-bit numbers to represent a network.
Network number = IP address + Mask

IP Address : 12.4.0.0 IP Mask: 255.254.0.0



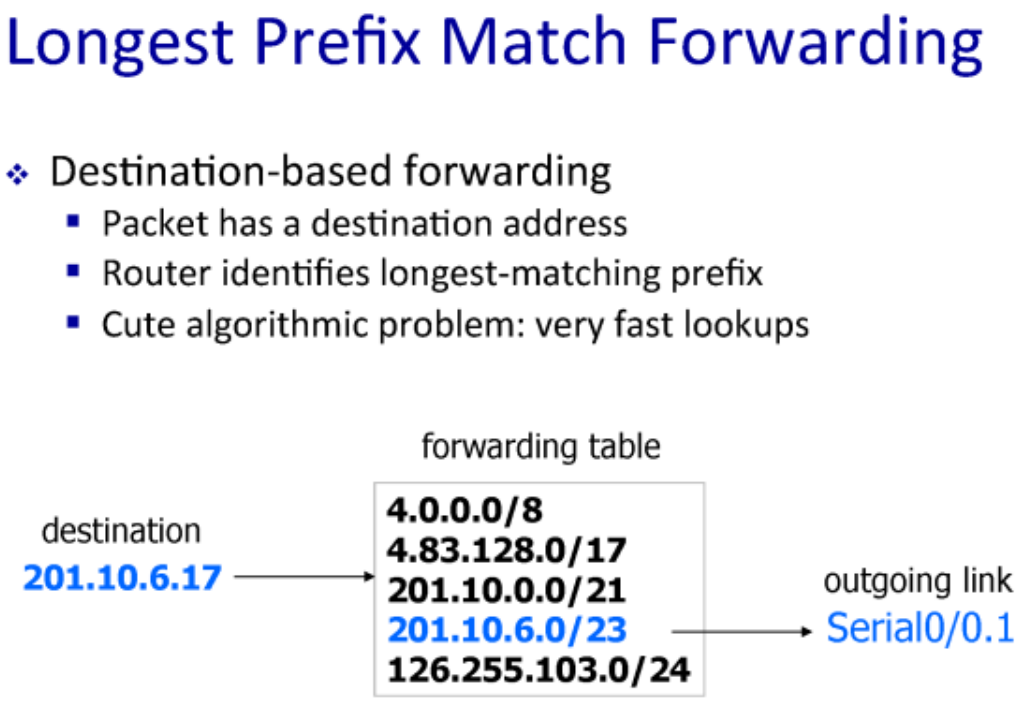
Written as 12.4.0.0/15

Classless

-> 클래스 구분 없이 필요한만큼 유연하게 할당

- 네트워크 라우터 내부의 forwarding table 사이즈도 감소

Longest Prefix Match Forwarding

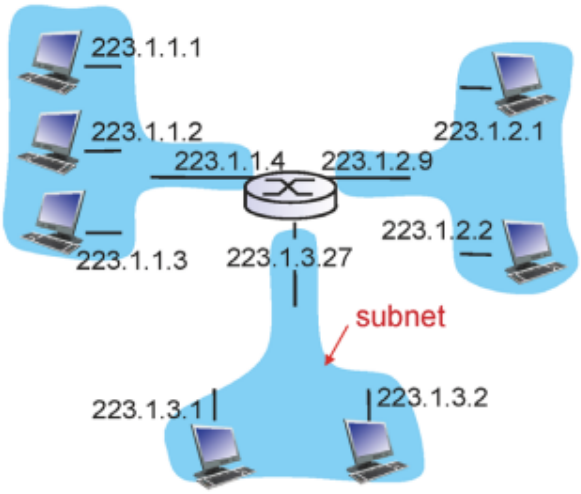


- Prefix 기반으로 가장 구체적으로(길게) 매칭되는 것

router의 주요 업무

Subnets

- IP address:
  - subnet part - high order bits
  - host part - low order bits
- *what's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other *without intervening router*

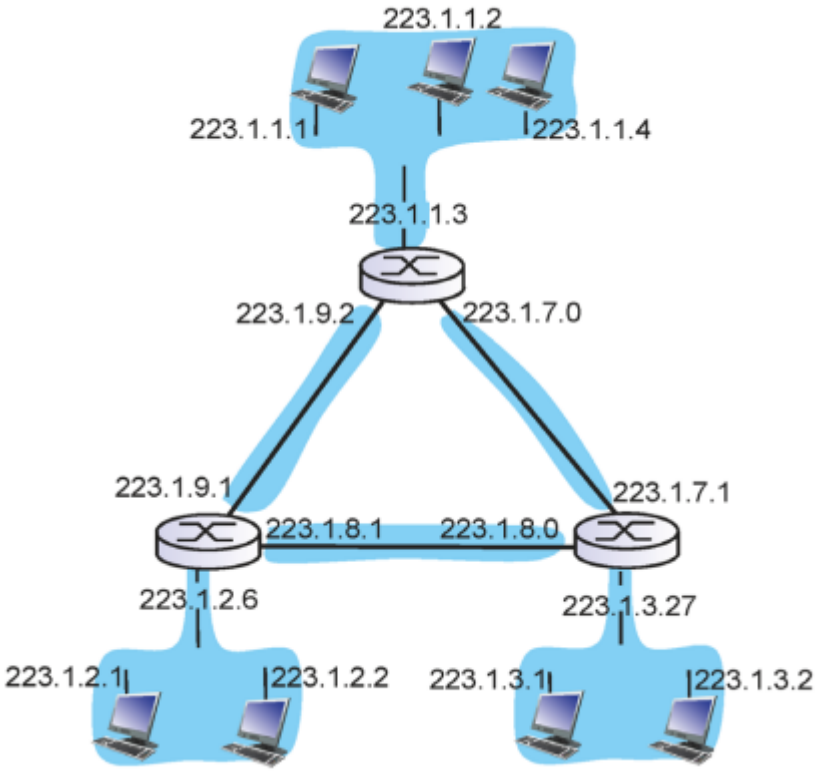


network consisting of 3 subnets

- 같은 prefix를 가진 interface를 가진 device의 집합

router를 거치지 않고

접근 가능한 호스트들의 집합



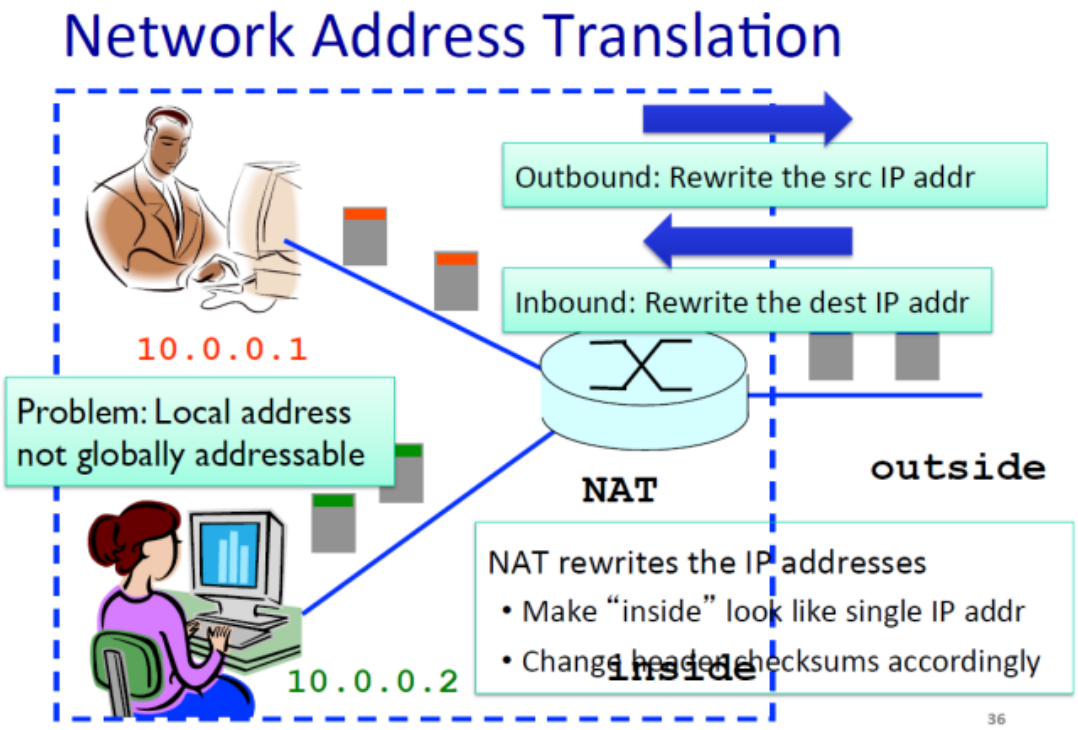
✔️ 위 그림은 6개의 subnet을 가진다.

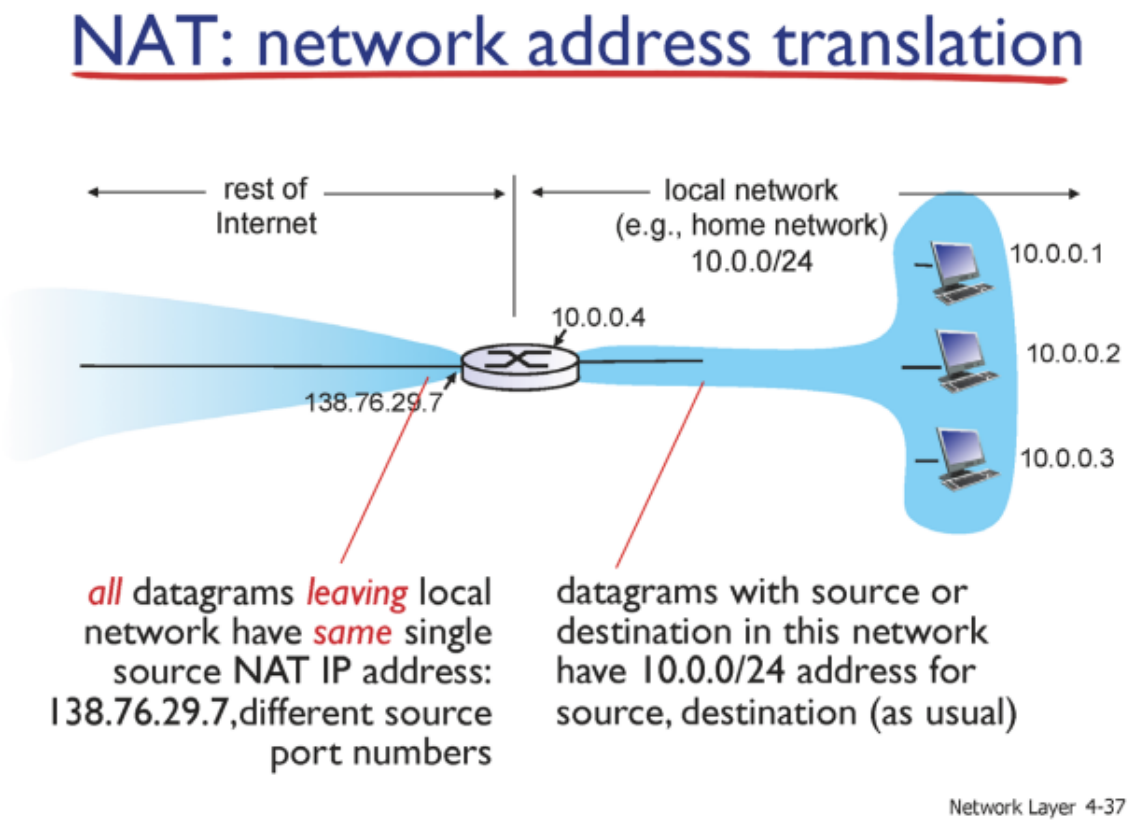
Network Address Translation

IPv6

- 1990년대 말 32bit 기반의 주소공간 40억개가 고갈될 우려에 처하자 **128bit 기반의 IPv6** 발표
- **현재도 IPv4를 사용**한다.

Network Address Translation





- 주소들은 **내부적으로만 유일** 하고 외부로 나갈 때 라우터를 거치면서 **라우터의 src 주소와 prt# 로 바뀌어서**나간다.

NAT의 문제점

Layer Violation

- Network layer의 router가 Transport layer의 port#까지 접근하고 수정하는 문제 발생

Port#의 잘못된 활용

- 앞선 과정에서 port#를 변환해서 사용하기 때문에 NAT내부에서 port#를 활용한 서버를 사용할 때 외부에서 해당 서버로 접근하기 어렵다는 문제 발생

문제