20230420_TIL - Network Layer(1)

Recap: Transport Layer

- loss와 delay가 발생하는 이유
    - 패킷들이 라우터에 있는 버퍼에서 기다리기 때문
    - 그 줄에 기다리면 `버퍼링(delay)`이 생긴다
    - 그 줄에 서지도 못하면 `loss`가 생긴다

HTTP protocol

HyperText Transfer Protocol. application layer의 통신 규약

- `클라이언트`: request를 보내고 response를 수신하는 브라우저. 웹 object를 본다(우리에게 보여준다)
- `서버`: request에 대한 response를 송신한다.

HTTP connection

- Non-Persistent HTTP
  - 하나의 TCP 통신으로 최대 하나의 object만 전송 가능. 전송이 끝나면 연결 종료
  - 여러 개의 object를 다운로드 하기 위해선 여러 개의 연결 필요
- Persistent HTTP
  - 여러 개의 object가 하나의 TCP 연결로 전송 가능

end-to-end delay?

- 패킷이 소스부터 목적지까지 가는데 걸리는 시간

UDP

- 멀티미디어 어플리케이션에서 자주 사용됨
  - SNMP(Simple Network Management Protocol)
    - 간이 망 관리 프로토콜.
    - IP 기반 네트워크에서 각 호스트로부터 정기적으로 정보를 수집하거나 실시간으로 상태를 모니터링할 수 있다.
    - UDP 통신이나 네트워크 장비의 효율적 관리를 위해 사용됨.
  - DNS(Domain Name System)
    - 우리가 아는 사이트 주소(도메인 이름)을 가지고 IP 주소 정보를 가져오는 시스템.
    - 웹사이트 데이터가 있는 호스팅 서버는 곧 인터넷 회선이 연결되어 있는 컴퓨터/장치나 마찬가지기 때문에, IP주소가 실제 웹사이트 주소나 마찬가지.
- 이 두가지를 통해 udp도 application layer에 신뢰할 수 있는 통신 서비스 제공 가능.

pipelined protocol

> 💡 송신자가 다수의 패킷을 한 번에 보내는 것을 말함.
> ACK 신호를 받을 때까지 기다리는 방식(`stop-n-wait`)이 아니라
> 한 번에 여러 개의 패킷을 보내고 방식에 따라 잘못된 부분을 처리하는 방식

- go-back-n
  - sender가 전송할 패킷 개수 결정
  - sender 버퍼에 그 개수만큼 저장해서 전송 - > window size
  - 누적 ACK 사용, 수신자는 이번에 받을 패킷의 정보 기억
  - 만약 송신자가 수신자로부터 특정 ACK를 받지 못했다면 timeout 이후 window에 있는 모든 패킷 다시 전송(해당 패킷부터 재전송 될 것)
- selected repeat
  - 위 방식의 비효율성을 개선하기 위해 나온 방식
  - 각 패킷마다 타이머 설정, 타이머 터지는 애들만 재전송
  - receiver도 버퍼가 있음
  - 기다리던 패킷이 오면 상위계층으로 전달, ack 신호 보냄
  - 기다리고 있던 패킷이 안 오면 일단 buffer에 넣고 대신 온 패킷에 대해 ack신호를 보내서 수신표시
  - sender의 send base는 재전송한 패킷에 대한 ack가 왔을 때 비로소 이동
  - window size가 sequence num의 개수와 엇비슷하면 안받은 패킷을 받았다고 착각할 위험성 있음

    → 그래서 selected repeat은 winSize가 seq#의 절반 이하여야.

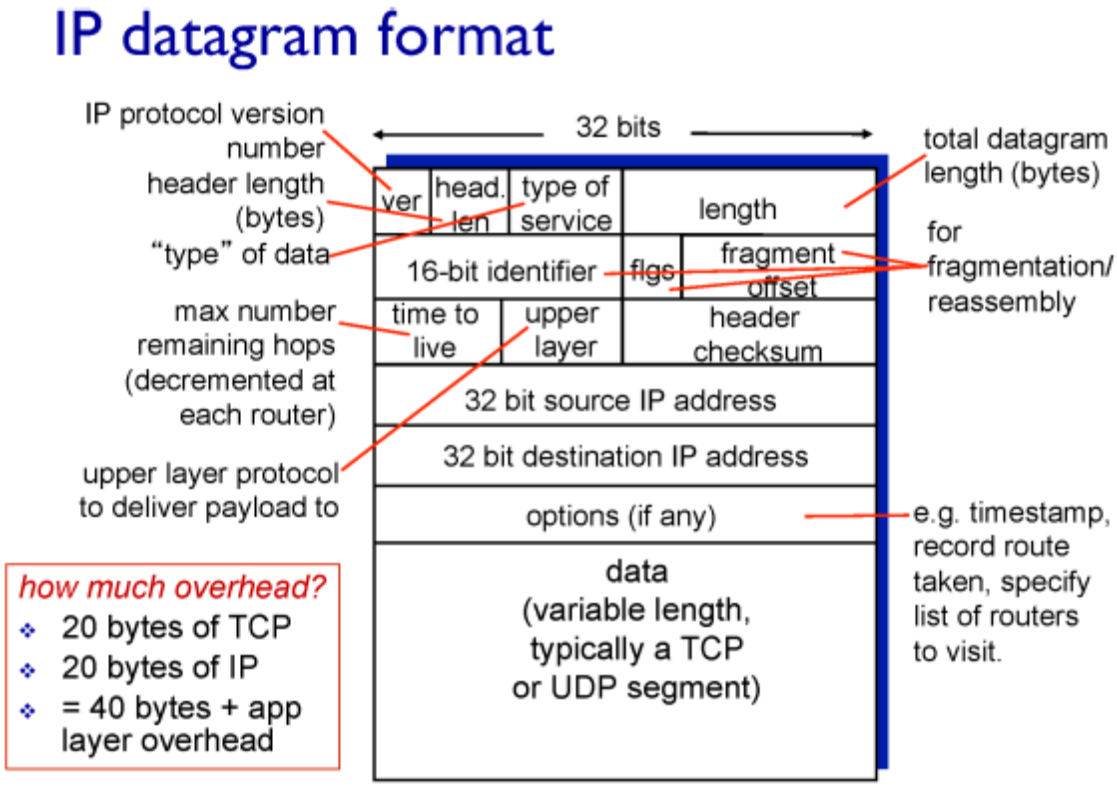
Network Layer

- 전송단위: `segment`
- sender는 세그먼트를 datagram으로 캡슐화해서 보내고, receiver는 세그먼트를 전송계층으로 세그먼트를 보낸다.

주요 기능

- `forwarding`: 패킷을 한 라우터의 input에서 적절한 다음 라우터의 output으로 이동시킴
- `routing`: 패킷이 출발지점(src)에서 도착지(dest)로 가는 경로 결정
- 라우팅(알고리즘)을 통해 path 결정하면 포워딩 테이블 생성, 현재 위치에서 포워딩 테이블을 따라 하나씩 이동(포워딩)시킴

IP datagram format



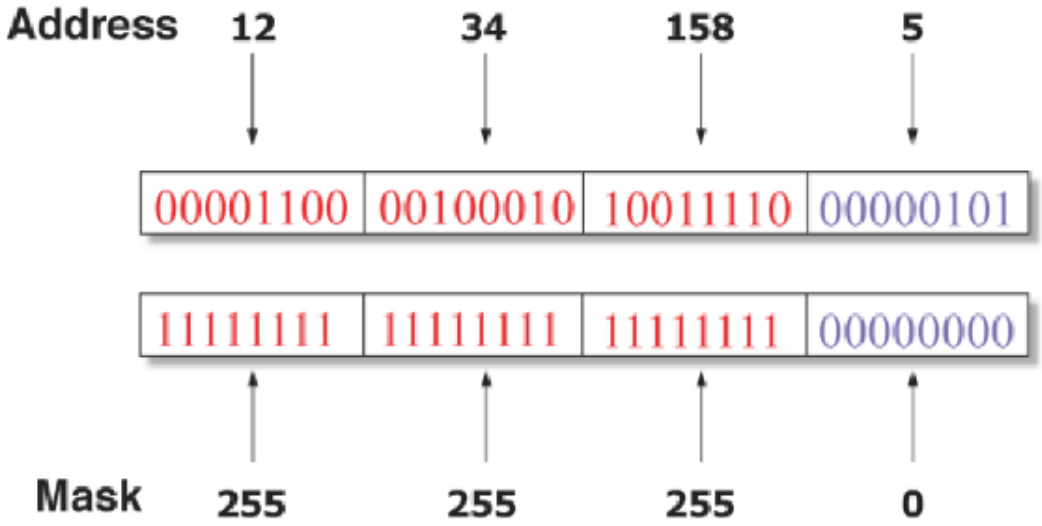
- 헤더만 40바이트 = 320비트, 그중 64비트는 ip 주소(32 + 32)

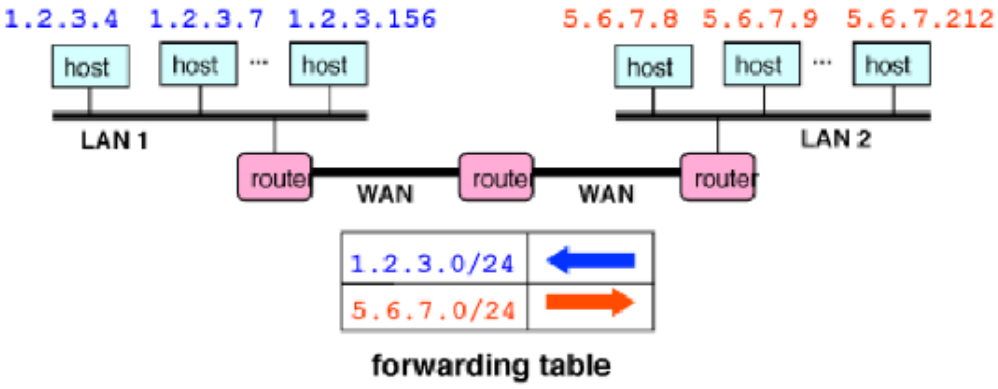
IP 주소

- 고유한 32자리(32bit) 이진수
- 호스트, 라우터 등 interface 식별
- 세자리 마다 점 찍는 notation(== 최대 3자리씩)
  - ex) 12.34.158.5

- IP 주소의 앞 3자리는 네트워크, 앞 1자리는 호스트를 의미한다.
- 같은 네트워크라면 앞 3자리가 같다!
- `/` 로 prefix가 몇자리까진지 표시(~24)

Grouping Related Hosts

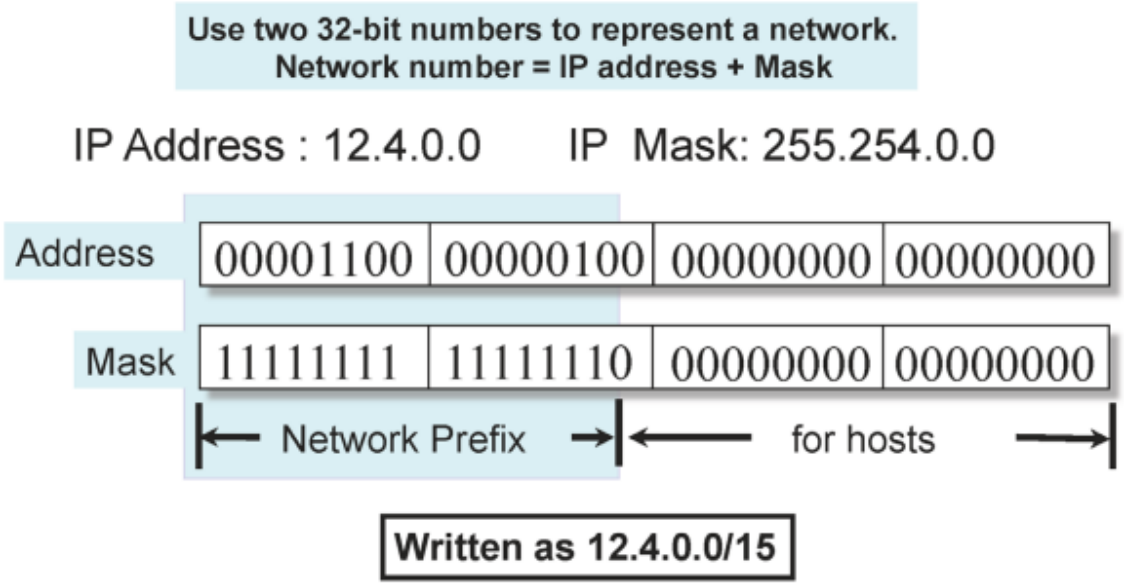




- 호스트(개인 pc) 사이에는 LAN(Local Area Network)로 연결되어 있고, 라우터 사이에는 WAN(Wide Area Network)로 연결되어 있다.
- 라우터에서 라우터로 넘어가면서, prefix가 최대로 match되는 인접 라우터로 이동한다.

Classless Inter-Domain Routing(CIDR)

- prefix의 길이를 앞 3자리(==24bit)로 고정하지 않고, 다양하게 나눔으로써 기존의 클래스 기반 IP 주소 할당 방식보다 더 효율적으로 사용



- IP 마스크를 통해 prefix의 길이, ip address를 통해 앞자리 값 확인

subnet

- 라우터를 거치지 않고 접근이 가능한 IP 주소의 집합 == prefix 동일