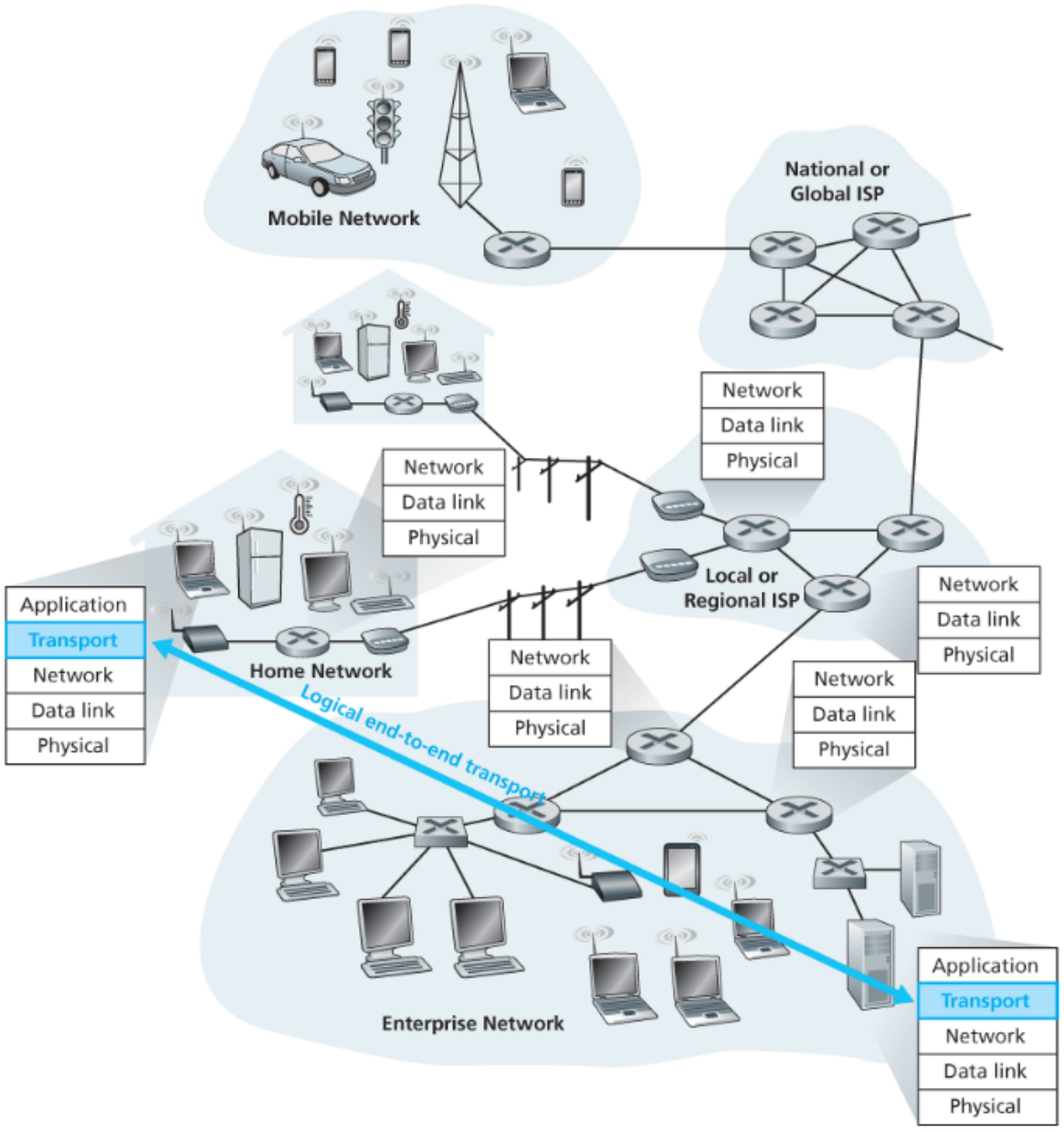
Transport Layer

Transport-layer services

트랜스포트 계층 프로토콜은 각기 다른 호스트에서 동작하는 애플리케이션 프로세스 간의 논리적 통신(logical communication)을 제공

- 논리적 통신 : 어플리케이션 관점에서 프로세스가 동작하는 호스트들이 직접 연결된 것 처럼 보이는 것



Transport vs. network layer services and protocols

- 네트워크 계층 프로토콜은 **호스트들** 사이의 논리적 통신을 제공
- 트랜스포트 계층 프로토콜은 각기 다른 호스트에서 동작하는 애플리케이션 프로세스 간의 논리적 통신 제공
  - 트랜스포트 계층 프로토콜은 종단 시스템에 존재하며, 애플리케이션 프로세스에서 네트워크 계층 사이에서 메시지를 운반하는 역할
  - 메시지가 네트워크 계층 내부에서 어떻게 이동하는지는 언급하지 않음

Two principal Internet transport protocols

Transmission Control Protocol (TCP)

- 신뢰적이고 연결지향형 서비스를 제공한다. (reliable data transfer)
- 혼잡 제어(congestion control) : 혼잡한 네트워크 링크에서 각 TCP 연결이 링크의 대역폭을 공평하게 공유하여 통과하도록 해준다.
- 흐름제어

User Datagram Protocol, UDP

- 비신뢰적이고 비연결형인 서비스를 제공
- 혼잡제어 방식 없어 속도 저하 없음
- UDP 트랜스포트 프로토콜을 사용하는 애플리케이션은 허용이 되는 한, 그것이 만족하는 어떤 속도로든 전송할 수 있다.

Multiplexing/demultiplexing

demultiplexing (역다중화)

- 트랜스포트 계층 세그먼트의 데이터를 올바른 소켓으로 전달하는 작업

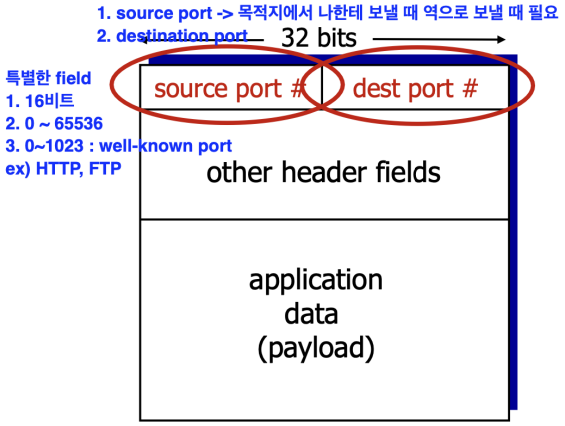
multiplexing (다중화)

- 출발지 호스트에서 소켓으로 부터 데이터를 모으고,
- 이에 대한 세그먼트를 생성하기 위해 각 데이터에 헤더 정보로 캡슐화(encapsulation)한다.
- 그 세그먼트들을 네트워크 계층으로 전달한다.

How demultiplexing works

- host receives IP datagrams
  - each datagram has source IP address, destination IP address
  - each datagram carries one transport-layer segment
  - each segment has source, destination port number
- host uses *IP addresses & port numbers* to direct segment to appropriate socket

1. 소켓은 유니크한 identifier가 있다.
2. 각각의 segment는 어떤 소켓으로 전달돼야 하는지 나타내는 특별한 field가 있다.
   1. source port -> 목적지에서 나한테 보낼 때 역으로 보낼 때 필요
   2. destination port



특별한 field
1. 16비트
2. 0 ~ 65536
3. 0~1023 : well-known port
ex) HTTP, FTP

TCP/UDP segment format

- 포트번호
  - 출발지 포트 번호 필드(source port number field)
  - 목적지 포트 번호 필드(destination port number field)
- 서비스
  - 호스트의 각 소켓은 포트 번호를 할당받는다.
  - 세그먼트가 호스트에 도착하면,
    1. 트랜스포트 계층은 세그먼트 안의 **목적지 포트 번호를 검사**하고,
    2. 그에 상응하는 소켓으로 세그먼트를 보낸다.
  - 세그먼트의 데이터는 소켓을 통해 해당되는 프로세스로 전달된다.

Connectionless demultiplexing

💡 UDP 소켓은 목적지 IP 주소와 목적지 포트 번호로 구성된 두 요소로 된 집합에 의해 식별된다

Connection-oriented demultiplexing

TCP 소켓은 아래의 4개로 식별

- 출발지 IP 주소
- 출발지 포트 번호
- 목적지 IP 주소
- 목적지 포트 번호

Connectionless transport: UDP

비연결형 트랜스포트 : UDP

- UDP는 트랜스포트 계층 프로토콜이 할 수 있는 최소 기능으로 동작
- TCP대신 UDP를 선택한다면 IP와 직접 통신하는 것과 비슷
  - UDP동작과 TCP동작의 차이를 보자
  - udp 동작
    - 어플리케이션 프로스세가 데이터를 udp에게 전달
    - 전달하자마자 UDP는 데이터를 UPD 세그먼트로 만든다.
    - 해당 세그먼트를 네트워크 계층으로 전달함
  - TCP
    - 링크가 혼잡해지면 TCP송신자를 조절하여 혼잡제어를 한다.
    - 전달이 오래걸려도 세그먼트 수신여부를 확인 응답할때까지 데이터의 세그먼트 재전송을 계속 진행한다.
  - 요렇게 UDP를 사용하면 UDP의 기본 세그먼트 전달 외에도 필요한 추가 기능 구현 가능해짐

UDP: User Datagram Protocol [RFC 768]

INTERNET STANDARD

RFC 768

J. Postel
ISI
28 August 1980

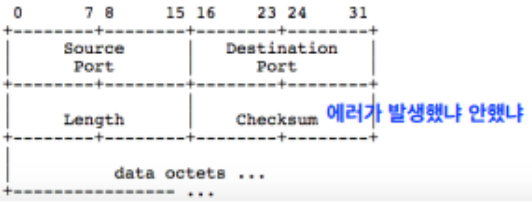
User Datagram Protocol

Introduction

This User Datagram Protocol (UDP) is defined to make available a datagram mode of packet-switched computer communication in the environment of an interconnected set of computer networks. This protocol assumes that the Internet Protocol (IP) [1] is used as the underlying protocol.

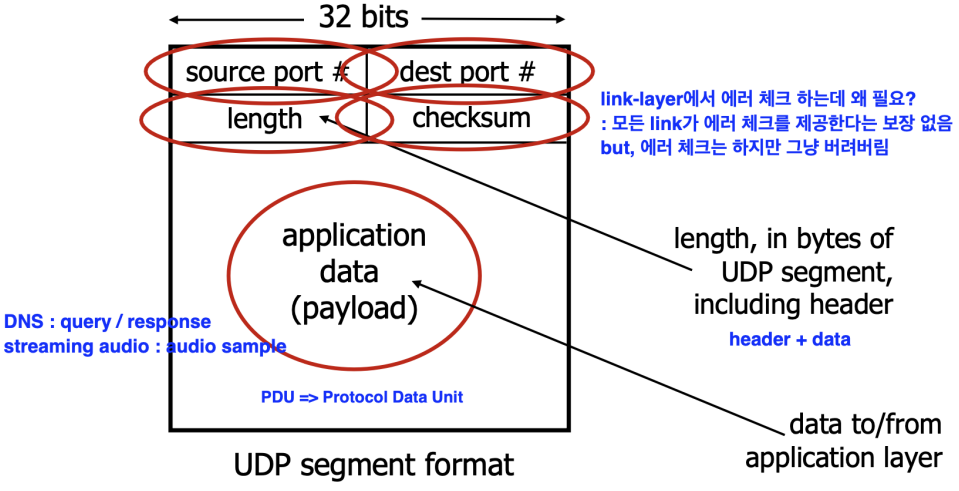
This protocol provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed. Applications requiring ordered reliable delivery of streams of data should use the Transmission Control Protocol (TCP) [2].

Format



UDP 세그먼트 구조

UDP segment header



포트 번호

- source port + dest port
- 목적지 호스트가 정확한 프로세스에게 애플리케이션 데이터를 넘기게 해준다.

애플리케이션 데이터

- UDP 데이터그램의 데이터 필드에 위치

체크섬(checksum)

- 세그먼트에 오류가 발생했는지 검사하기 위해 수신 호스트가 사용한다.

길이 필드

- 헤더를 포함하는 UDP 세그먼트의 길이를 바이트 단위로 나타낸다.

UDP 체크섬

왜 체크섬 필요?

출발지와 목적지 사이의 모든 링크가 오류 검사를 제공한다는 보장이 없기 때문에 세그먼트가 라우터의 메모리에 저장될 때 비트 오류가 발생할 수가 있다.

end-end principle (종단간의 원리)

end-end priciple

- 주어진 링크 간의 신뢰성과 메모리의 오류 검사가 보장되지도 않고, 종단 간의 데이터 전송 서비스가 오류 검사를 제공해야 한다면
- UDP는 종단 기반으로 트랜스포트 계층에서 오류 검사를 제공해야만 한다.

Principles of reliable data transfer

신뢰성 있는 데이터 전송의 원리

- rdt : 신뢰적인 데이터 전송
- _send : rdt의 송신 측이 호출되고 있다는 것을 나타냄
- rdt_rcv() : 패킷이 채널의 수신측으로부터 도착했을 때 호출
- deliver_data() : rdt 프로토콜이 상위계층에 데이터를 전달하려고 할 때 호출
- udt : 비신뢰적인 데이터 전송

유한상태 머신(finite-state machine, FSM)에 대한 정의

- 화살표는 한 상태로부터 다른 상태로의 전이를 나타낸다.
- FSM의 초기 상태는 점선 화살표로 표시된다.
- **전이를 일으키는 이벤트(event)**는 변화를 표기하는 가로선 위에 나타낸다.
- 이벤트가 발생했을 때 취해지는 행동, **액션(action)**은 가로선 아래에 나타낸다.
- 이벤트 발생 시 어떠한 행동도 취해지지 않거나, 어떠한 이벤트 발생 없이 행동이 취해질 때**동작이나 이벤트가 없음**을 표시하기 위해 각각 가로선 아래나 위에 기호 Λ를 사용한다.

rdt1.0

- 완벽하게 신뢰적인 채널상에서의 신뢰적인 데이터 전송

rdt2.0

- 비트 오류가 있는 채널 상에서의 신뢰적 데이터 전송
- 송신측과 수신측의 두가지 상태를 가진다.
    - 송신
        - 상위계층으로부터 데이터가 전달되기를 기다림

        - rdt_send(data) 발생
        - 패킷을 udt_send()를 통해 전송
        - 수신자로부터 ACK 또는 NAK패킷 기다림
            - ACK수신하면
                - 송신자가 패킷이 정확하게 수신되었다는 것을 알게됨
            - NAK 수신하면
                - 프로토콜은 마지막 패킷을 재전송
                - 재전송된 패킷에 대한 응답을 기다림
    - 수신
        - 단일상태를 가짐
        - 패킷 도착 시 수신자는 패킷이 손상되었는지 아닌지에 따라 ACK, NAK로 응답

rdt2.1

- 순서 번호 반영
- 순서가 바뀐 패킷이 수신되면 수신자는 이미 전에 수신한 패킷에 대한 긍정 확인응답 전송함
- 손상된 패킷 수신되면
    - 수신자는 부정 확인 응답을 전송
    - NAK를 송신하는 것 대신, 가장 최근에 정확하게 수신된 패킷에 대해 ACK를 송신하면서 NAK를 송신한 것과 같은 효과를 얻을 수 있음

rdt3.0

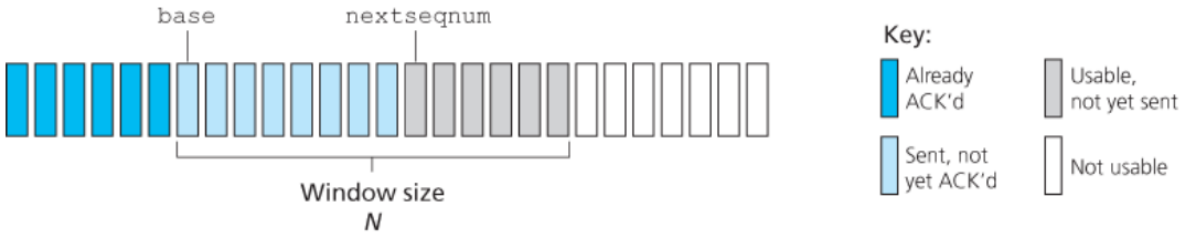
- 비트 오류와 손실있는 채널 상에서의 신뢰적 데이터 전송
- 카운트 다운 타이머 사용
    - 매 패킷이 송신된 시간에 타이머 시작
    - 타이머 인터럽트에 반응
    - 타이머 멈춤

pipelined protocols operation

파이프라이닝 오류 회복의 두 가지 접근 방법

1. N부터 반복
2. 선택적 반복

Go-Back-N



GBN 프로토콜

- 프로토콜 동작 시 순서 번호 공간에서 오른쪽으로 이동됨
- 슬라이딩 윈도우 프로토콜이라고 부름

GBN 송신자는 3가지 타입 이벤트에 반응함

- 상위로부터 호출
  - rdt_send()가 위로부터 호출되면, 송신자는 첫째로 윈도우가 가득 찼는지
  - 즉 N개의 아직 확인 응답 되지 않은 패킷이 있는지 확인
- ACK의 수신
  - GBN 프로토콜에서 순서번호 N을 가진 패킷에 대한 응답은 누적 확인응답으로 인식됨
  - 이 누적 확인 응답은 수신 측에서 올바르게 수신된 n을 포함하여, n까지의 순서 번호를 가진 모든 패킷에 대한 확인 응답임
- 타임아웃 이벤트
  - ACK응답이 제대로 전송되지 않으면 타임아웃 발생
  - 확인응답 안 된 패킷부터 다시 전송

Selective Reapat 선택적 반복

GBN 프로토콜은 송신자가 패킷으로 파이프라인을 채우는 것을 가능하게 하여, 전송-후-대기 프로토콜에서 채널 이용률 문제를 피하도록 하고 있음.

하지만 GBN자체에서도 성능이슈가 있음
GBN은 패킷 하나의 오류로 많은 패킷을 재전송하여 불필요한 작업을 하는 경우가 있음

그래서 선택적 반복 프로토콜 등장

- 오류가 발생한 패킷을 수신했다고 의심되는 패킷만을 송신자가 다시 전송하여 불필요한 재전송을 피한다.
- 윈도우 크기 N은 파이프라인에서 아직 확인응답이 안 된 패킷 수를 제한하는데 사용됨
- GBN가 달리 송신자는 윈도우에서 몇몇 패킷에 대한 ACK를 이미 수신했음

SR 송신자의 이벤트와 행동

- 상위로부터 데이터 받음
- 타임아웃
  - 타이머는 손실된 패킷을 보호하기 위해 다시 사용
- ACK 수신

SR 수신자의 이벤트와 행동

- [rcv_base, rcv_base + N-1]내의 순서번호를 가진 패킷은 손상 없이 수신
- [rcv_base - N, rcv_base - 1]내의 순서번호를 가진 패킷이 수신됨
- 이외의 경우는 무시

GBN VS SR

- GBN은 패킷 전송 시 수신측에서 데이터를 잘못 받은 것이 있거나 못받을 경우에 해당 패킷 번호로 부터 다시 재전송을 하는 기법이고,

- SR은 잘못 받은 패킷 번호만 재전송하는 것