Overview and History of the Internet

Sept 10, 2024

Min Suk Kang

Associate Professor

School of Computing/Graduate School of Information Security

NetS&P Research Lab @ KAIST

KAIST

Lab 1 will be released today

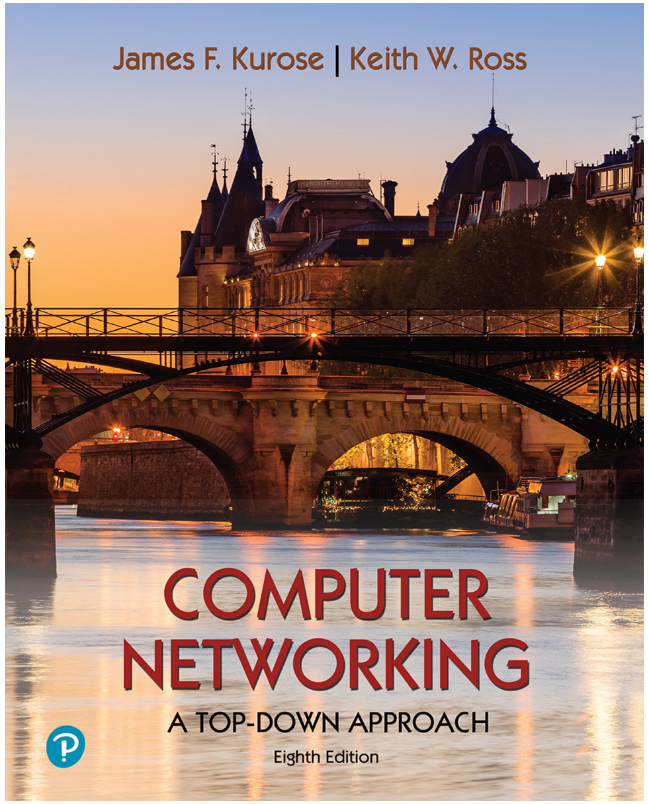
- Lab 1: Socket Programming
  - Due in **1 week + 2 days (due to Chuseok)**

↳ ditail은 classum에

- Please revisit late-submission policy

Chapter 1
Introduction

6
수업전
책
읽어보게
하고



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Chapter 1: introduction

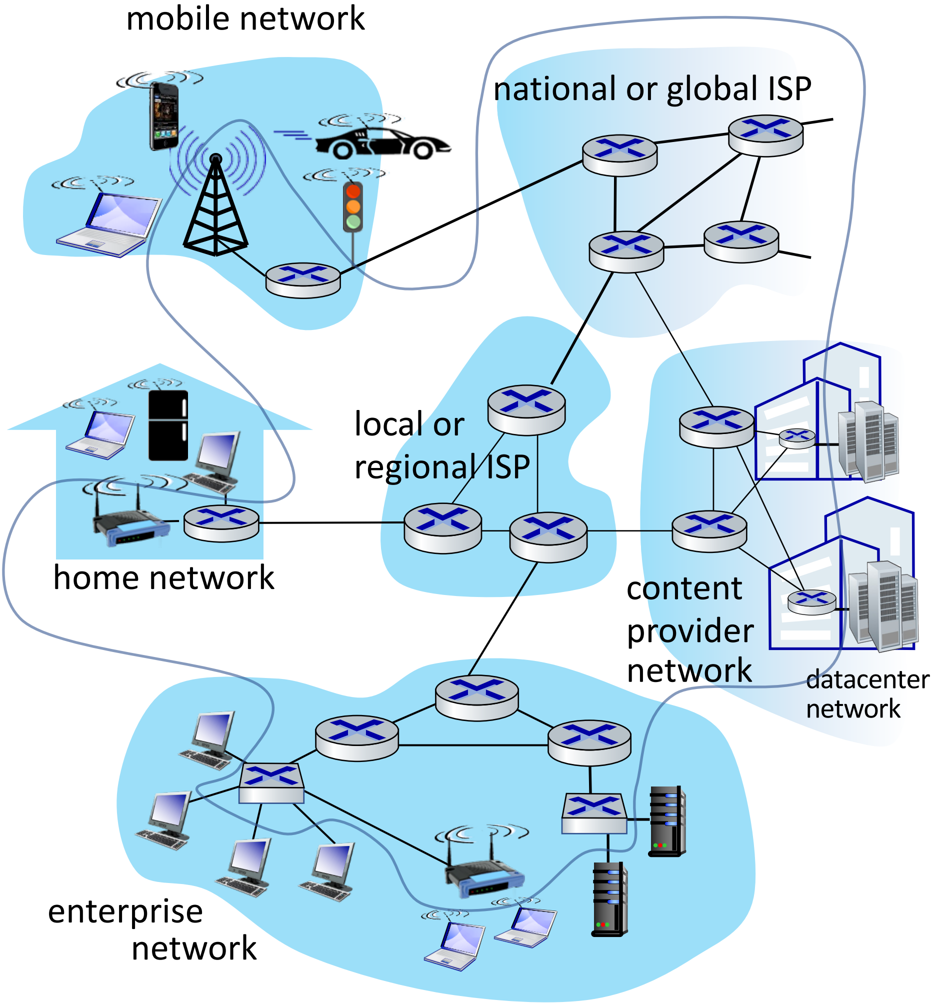
Chapter goal:

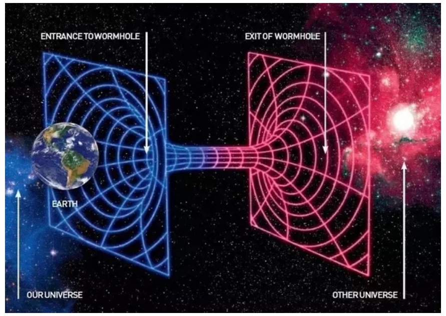
- Get "feel," "big picture," introduction to terminology
  - more depth, detail *later* in course

Overview/roadmap:

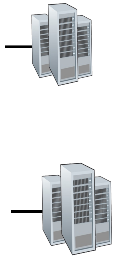
- What *is* the Internet? What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Protocol layers, service models
- Security
- History

Let’s travel through… a wormhole!

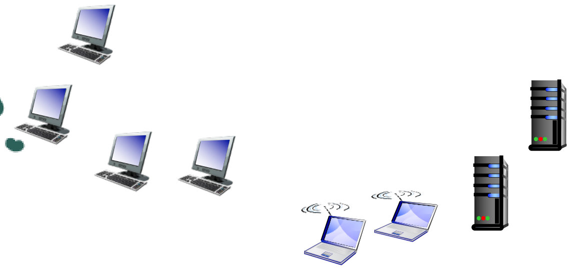




Network is gone!



네트워크가 없어졌대!

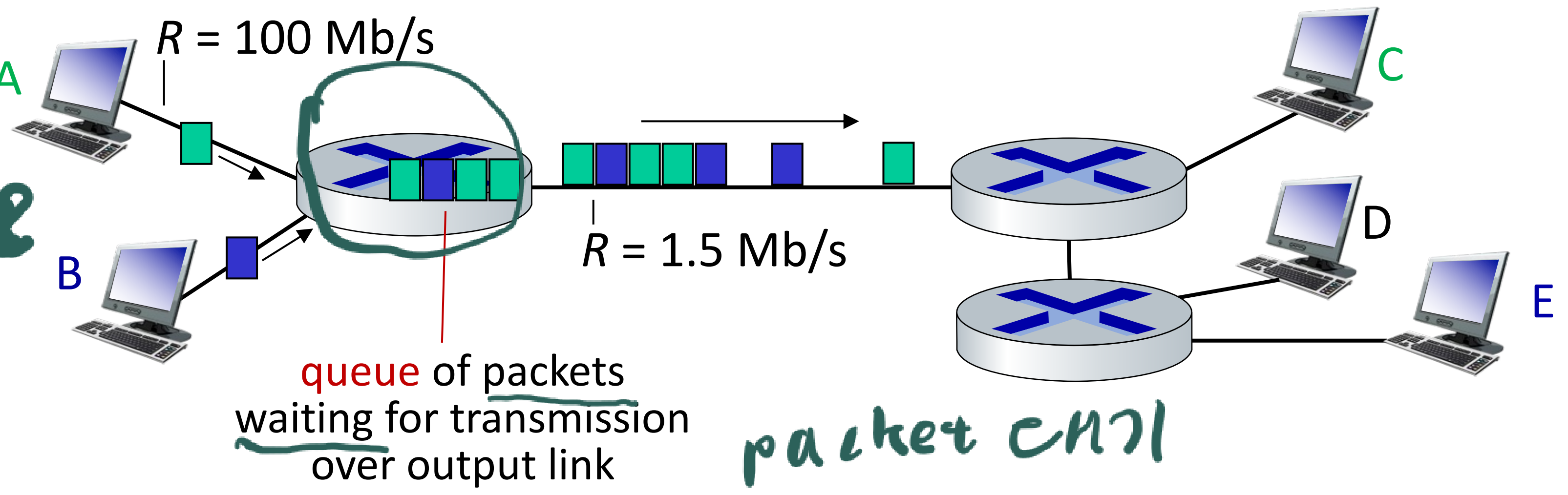


Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

Packet-switching: queueing

stream of bit or byte



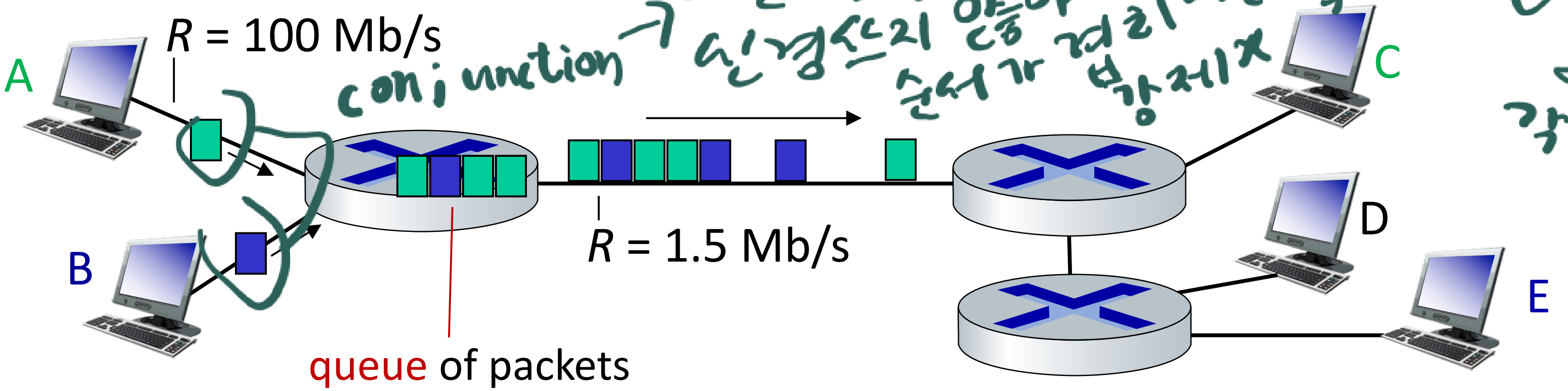
packet 대기

Queueing occurs when work arrives faster than it can be serviced:

패킷 전송 시간동안 대기

Packet-switching: queueing

multiple protocol이을 거치면 conjunction(결합) 발생
↓
각 packet이 도착 정보를 지니고 있기 때문



conjunction → 다른 신호를 신경쓰지 않아 순서가 겹치지 않도록 강제X

도착 속도 > 전송속도 ⇒ 대기열 발생

Packet queuing and loss: if arrival rate (in bps) to link exceeds transmission rate (bps) of link for some period of time:

- packets will queue, waiting to be transmitted on output link queue size↑
- packets can be dropped (lost) if memory (buffer) in router fills up

The bigger queue size, the better?

→ 비용↑, 복잡도↑으로 패킷간 간섭 ⇒ 정체, 우선순위 고려 어려움

-예전 방법

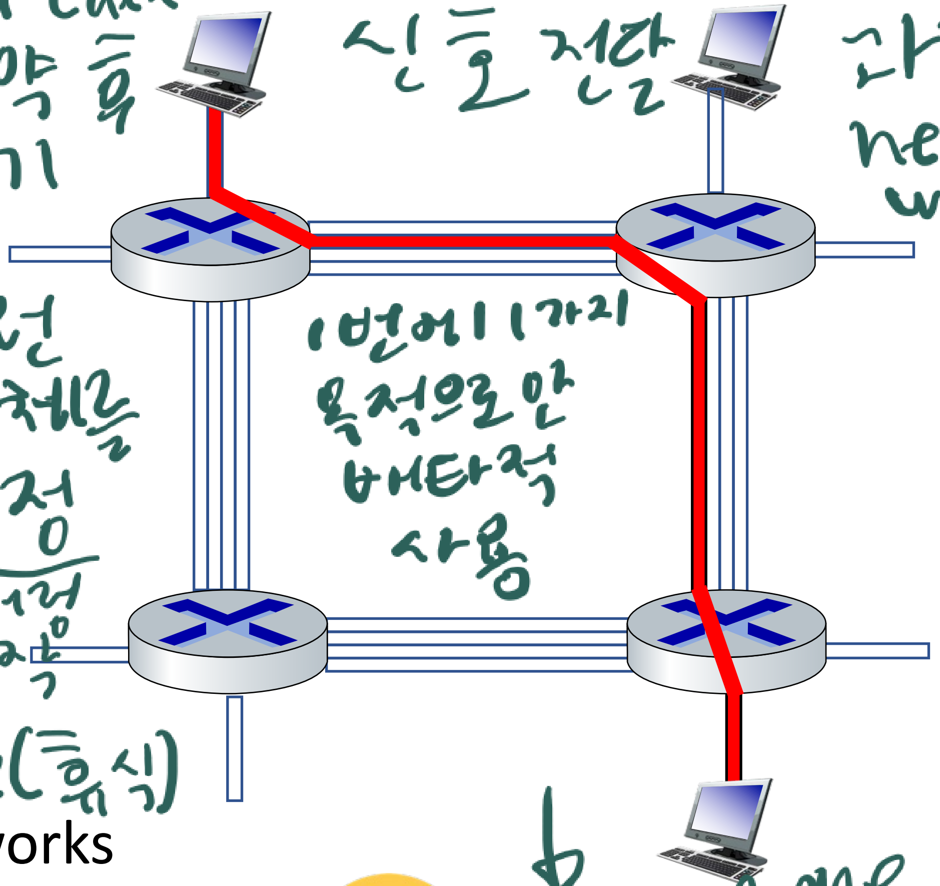
Alternative to packet switching: circuit switching

end-to-end 자원(회선) 할당 후 시작과 끝에서 call 예약 후 대기

end-end resources allocated to, reserved for "call" between source and destination

배타적으로 → 속도↓, 신호 전달

과거의 network



1번에 1가지 목적으로만 배타적 사용

- in diagram, each link has four circuits.
  - call gets 2^{nd} circuit in top link and 1^{st} circuit in right link.

회선 전체를 독점

- 독점 dedicated resources: no sharing
  - circuit-like (guaranteed) performance 회로처럼 동작
- circuit segment idle if not used by call (no sharing) call로 사용하지 않으면 idle(휴식)
- commonly used in traditional telephone networks

실시간 연결이 필요한 전화망 등

What else? Do we need it now?

장점: 1:1 연결이라 queue X, 지연X

telephone network or circuit switching network

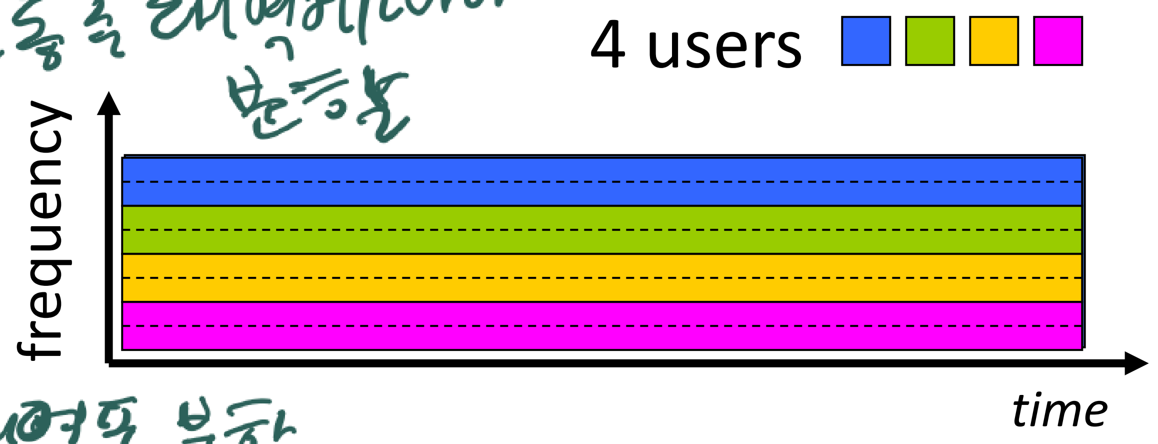
Circuit switching: FDM and TDM

Frequency Division Multiplexing (FDM)

광학적, 전자기학적 진동수 대역에 따라 분할

- optical, electromagnetic frequencies divided into (narrow) frequency bands
- each call allocated its own band, can transmit at max rate of that narrow band

→ 대역폭 분할

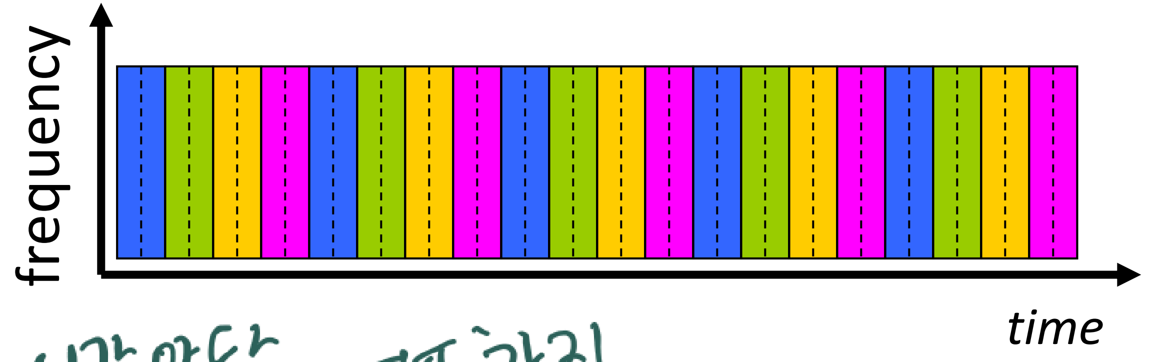


Time Division Multiplexing (TDM)

분할

- time divided into slots 시간 단위 분할
- each call allocated periodic slot(s), can transmit at maximum rate of (wider) frequency band (only) during its time slot(s)

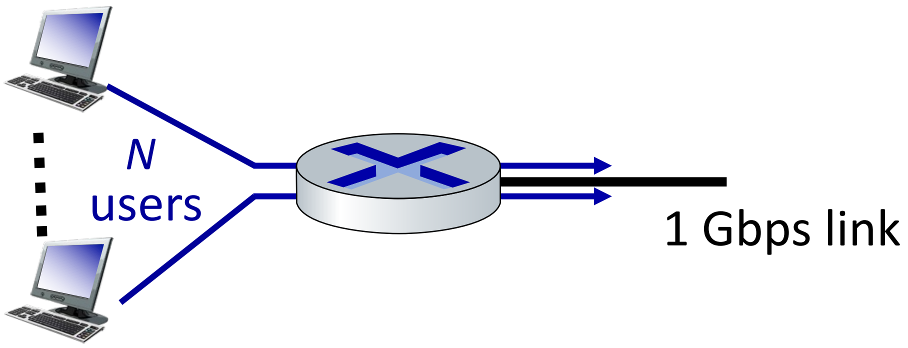
→ 시간마다 모든 대역폭 까지



Packet switching versus circuit switching

데이터를 쪼개 다음 링크로 보내기 전 저장하고 전송

example:
- 1 Gb/s link
- each user:
  - 100 Mb/s when "active"
  - active 10% of time



Q: how many users can use this network under circuit-switching and packet switching?

- *circuit-switching:* 10 users

$\frac{1000\,Mb/s}{100\,Mb/s} = 10$

- *packet switching:* with 35 users, →각 user는 개별적
probability > 10 active at same time is less than .0004

←한 번에 감당 가능한 수

link 독점X

Q: how did we get value 0.0004?

A: HW problem?

이항분포를 따를 때, $P(X>10) = \sum_{k=11}^{35} \binom{35}{k} 0.1^k\, 0.9^{35-k}$

$= 1 - \sum_{k=0}^{10} \binom{35}{k} 0.1^k\, 0.9^{35-k}$

$= 1 - 0.9995775\ldots$

$= 0.0004224\ldots$

What do we do for 0.0004 probability events?

⇒ link capacity 내에서 인원 제한 X

Packet switching versus circuit switching

Is packet switching a “slam dunk winner”?

불연속적 데이터에 적합

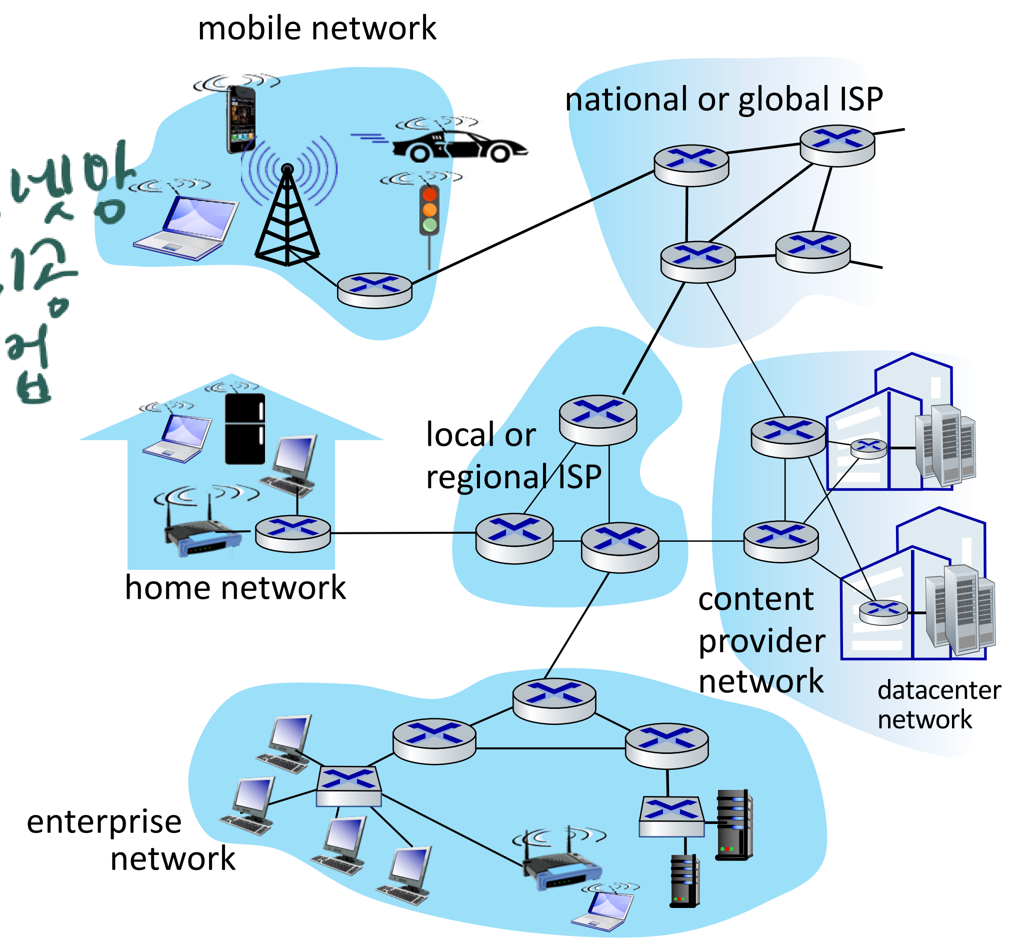
- great for “bursty” data – sometimes has data to send, but at other times not
  - resource sharing 자원 공유
  - simpler, no call setup 단순, 호출 설정 X

- excessive congestion possible: packet delay and loss due to buffer overflow
  과도한 혼잡
  - protocols needed for reliable data transfer, congestion control → 혼잡 제어와 신뢰성 있는 전송을 위해 프로토콜 필요
- *Q:* How to provide circuit-like behavior with packet-switching?
  - “It’s complicated.” We’ll study various techniques that try to make packet switching as “circuit-like” as possible.

⇒ 패킷 교환은 수요에 따라 링크 사용을 공유하므로 낭비X

Internet structure: a "network of networks"

- hosts connect to Internet via access Internet Service Providers (ISPs)
  ↳ 인터넷망 제공 기업
- access ISPs in turn must be interconnected 상호 연결
  - so that *any* two hosts *(anywhere!)* can send packets to each other
- resulting network of networks is very complex
  - evolution driven by economics, national policies



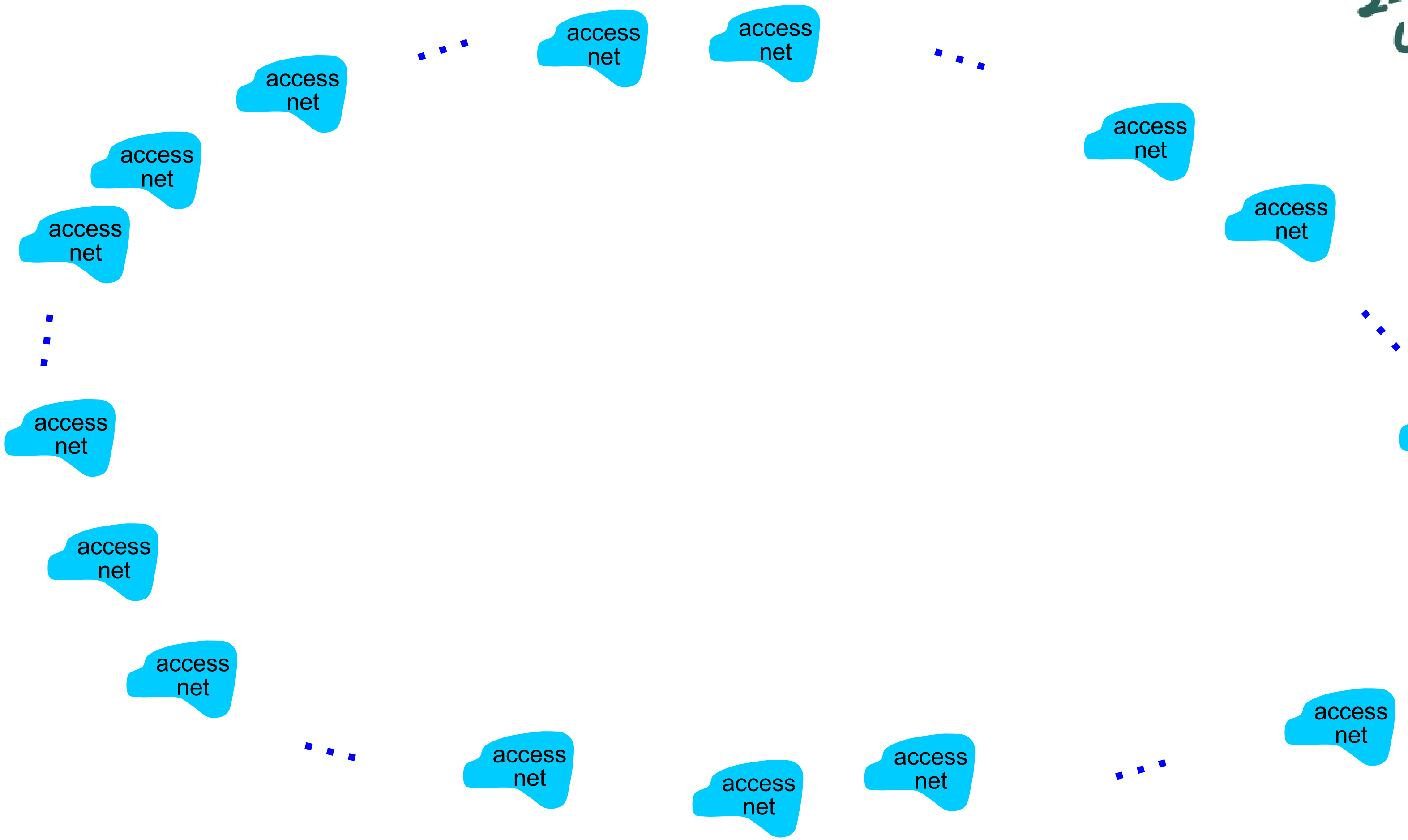
Let's take a stepwise approach to describe current Internet structure

Internet structure: a "network of networks"

Question: given *millions* of access ISPs, how to connect them together?

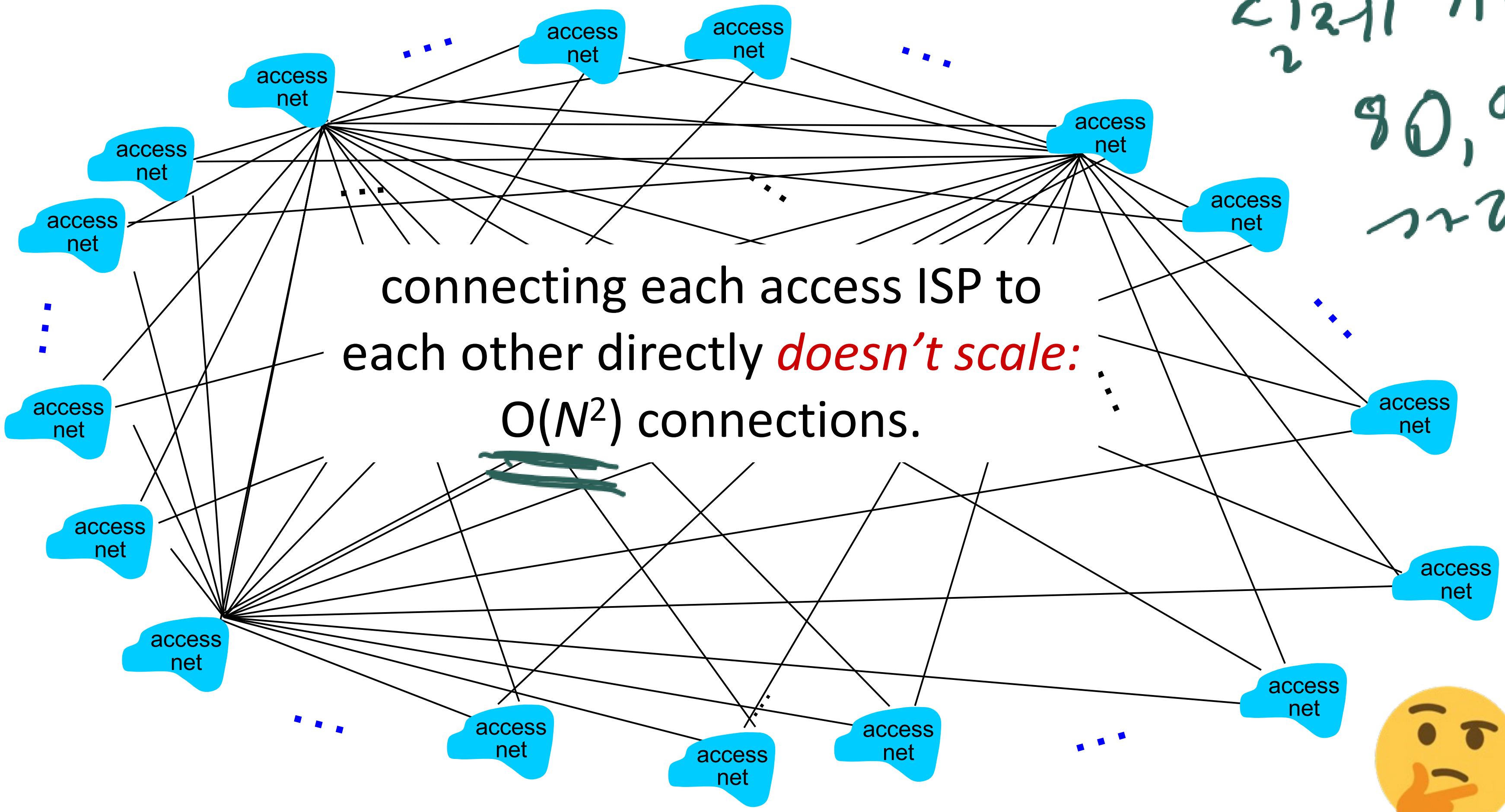
전부 연결하지 못하는 이유

비용↑



Internet structure: a "network of networks"

Question: given *millions* of access ISPs, how to connect them together?



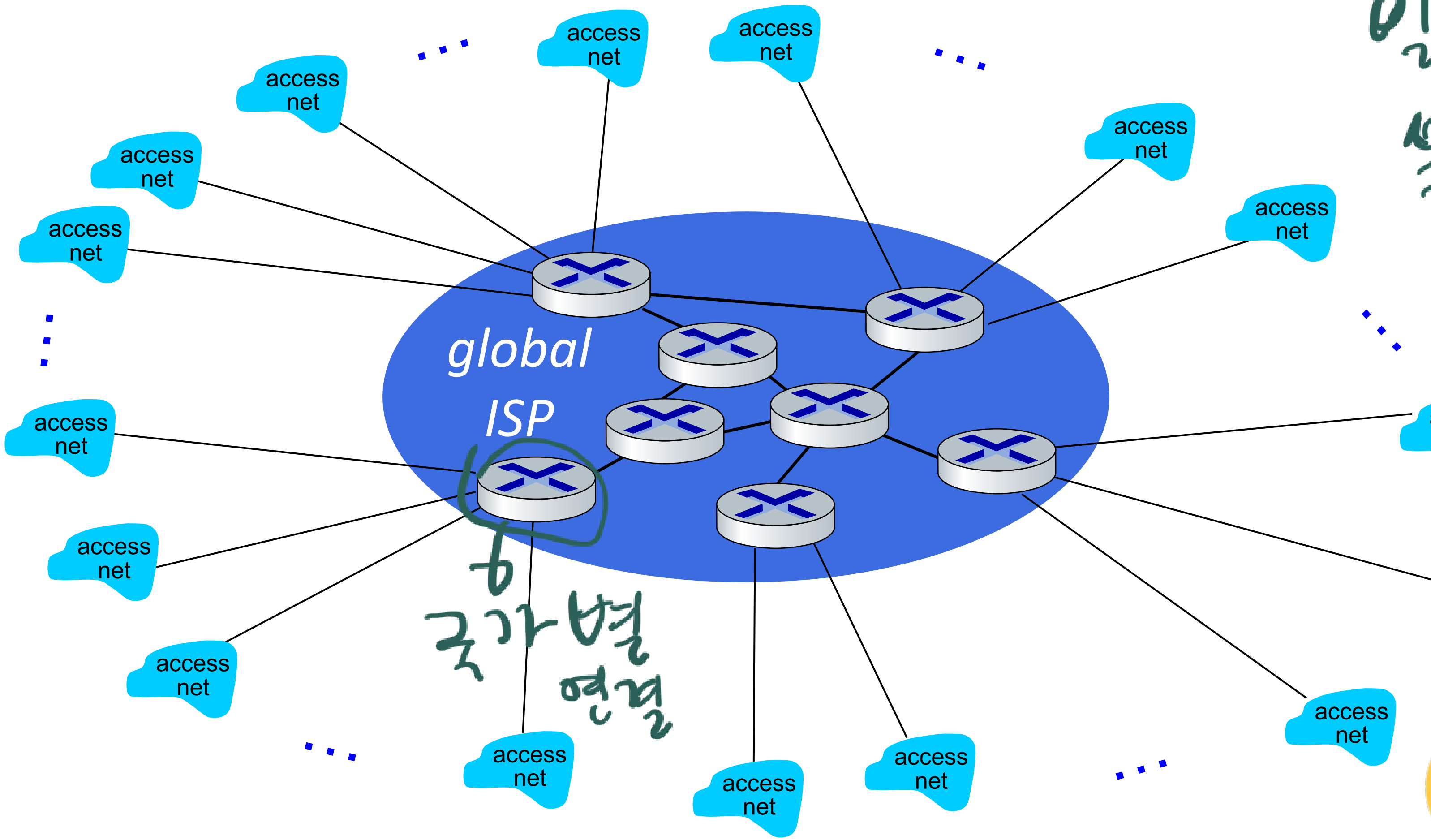
실제 기업은 거의
90,000개
가량의
네트워크에
access

Internet structure: a "network of networks"

Option: connect each access ISP to one global transit ISP?

Customer and provider ISPs have economic agreement.

일부 access net들을 묶어 국가망을 만들고, 국가망끼리 연결 (global ISP)

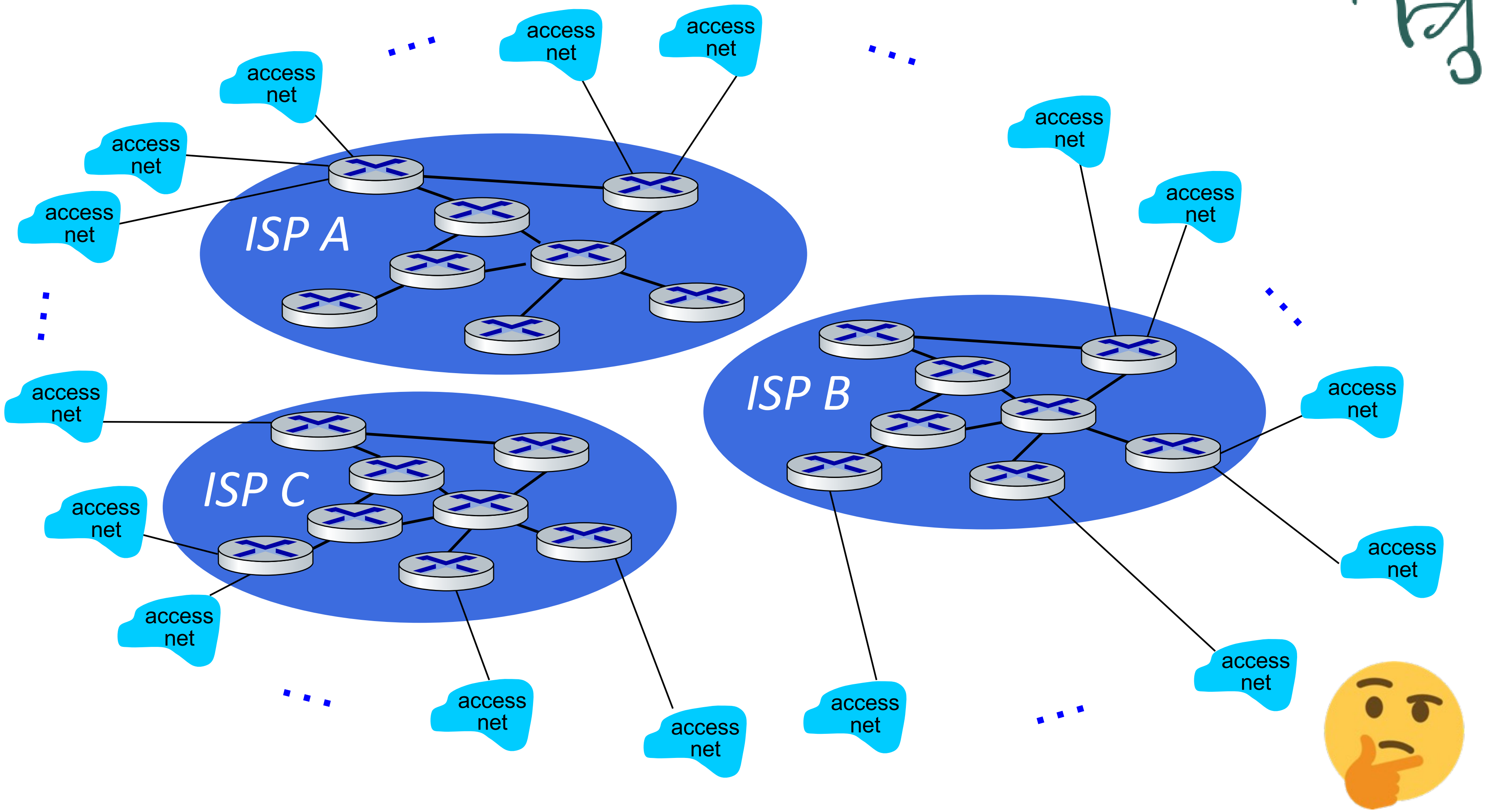


국가별 연결

Internet structure: a “network of networks”

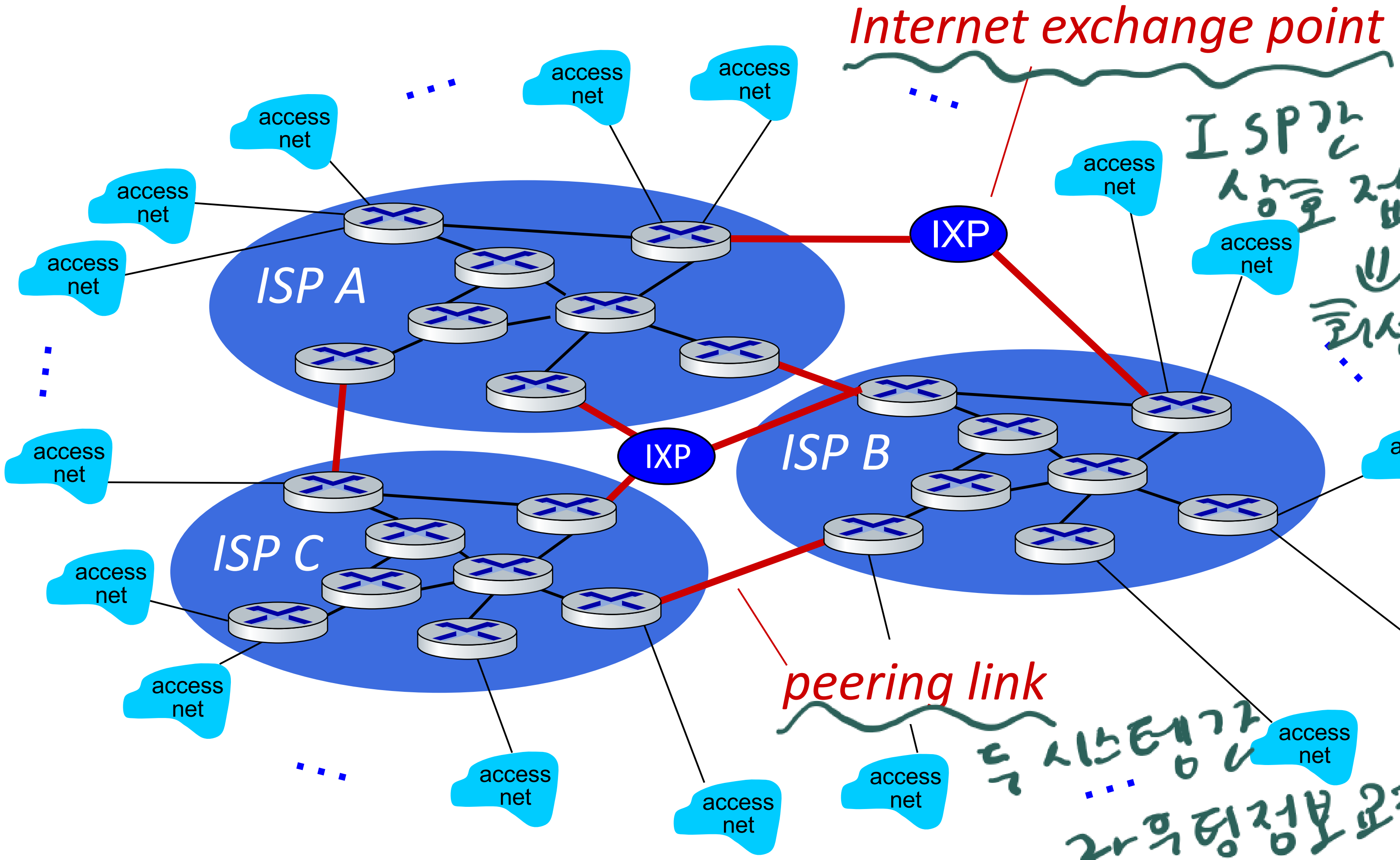
But if one global ISP is viable business, there will be competitors ….

global ISP 시장 확대로
경쟁
발발



Internet structure: a "network of networks"

But if one global ISP is viable business, there will be competitors …. who will want to be connected



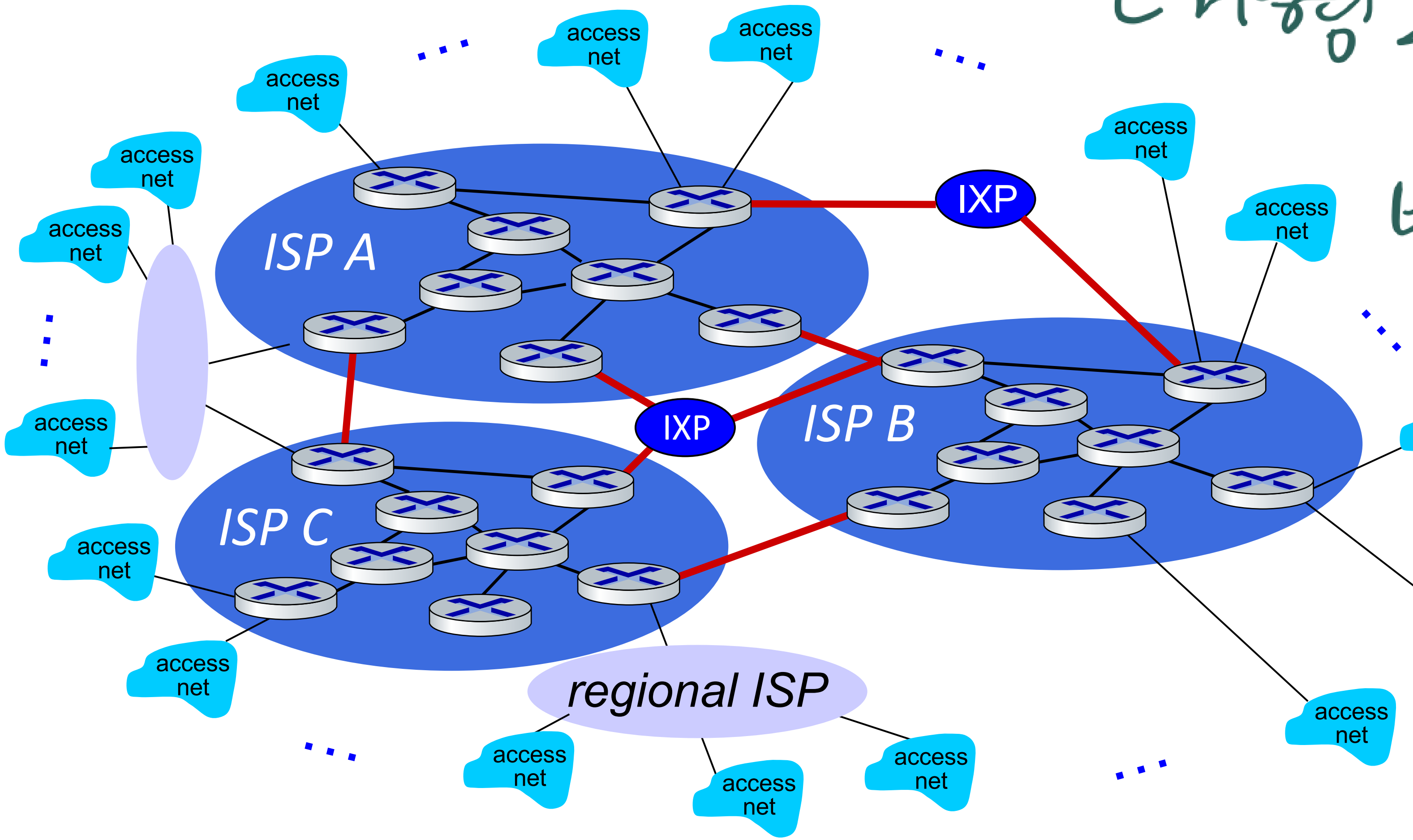
경쟁사끼리 연결해 인접 net과의 연결비용↓
(회선 설치 필요X, 회선 길이 단축)

ISP간 상호 접속
⇓
회선비용↓

두 시스템간 라우팅정보 교환
(네트워크 끼리 직접 트래픽교환)
→ 전송 비용↓, 연결 복잡도↓

Internet structure: a "network of networks"

… and regional networks may arise to connect access nets to ISPs



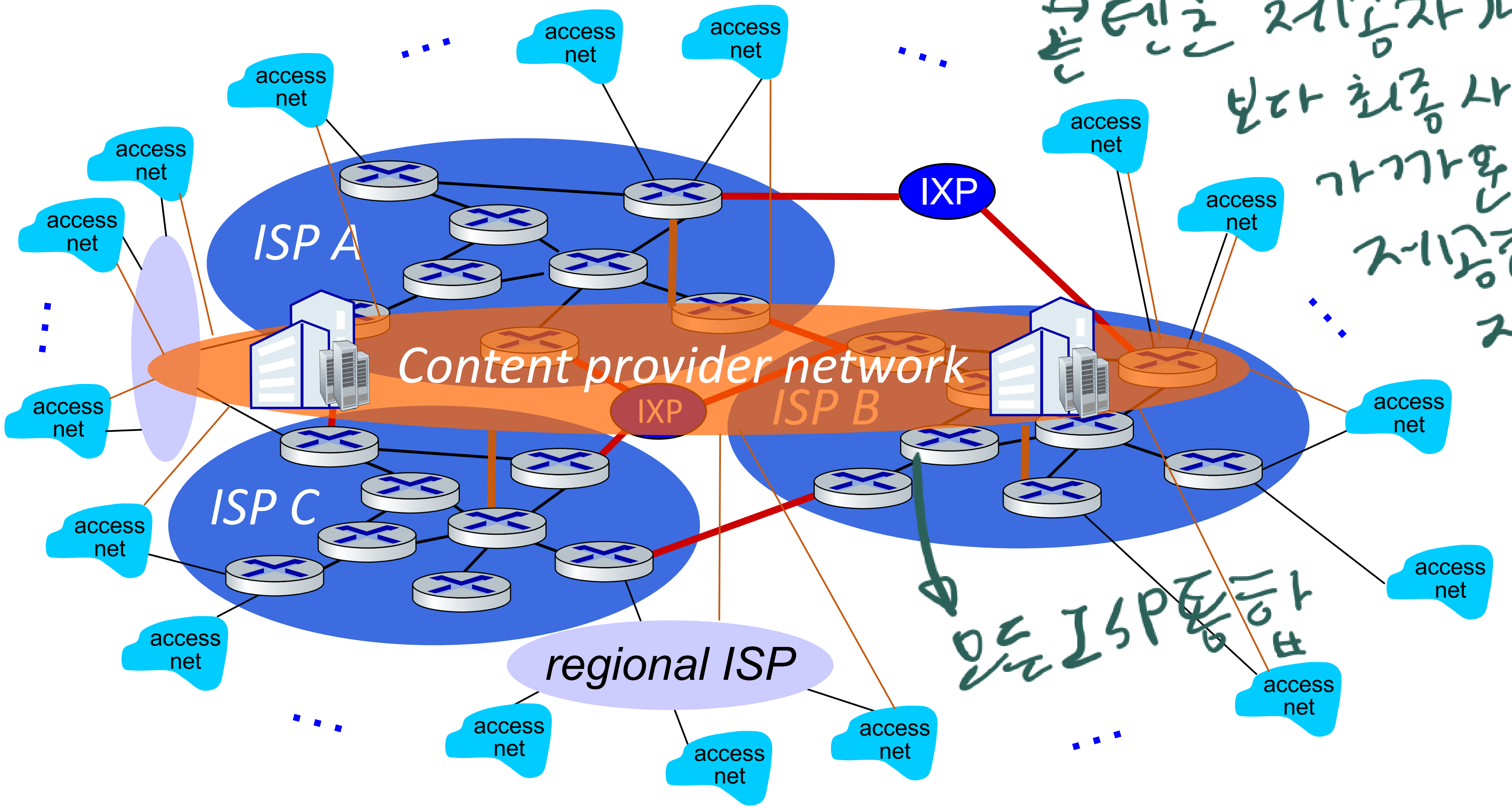
대형 ISP의 반응성↓

↓

반응성이 높고, 보다 개인화된 서비스를 지원하는 regional ISP 탄생

Internet structure: a "network of networks"

… and content provider networks (e.g., Google, Microsoft, Akamai) may run their own network, to bring services, content close to end users

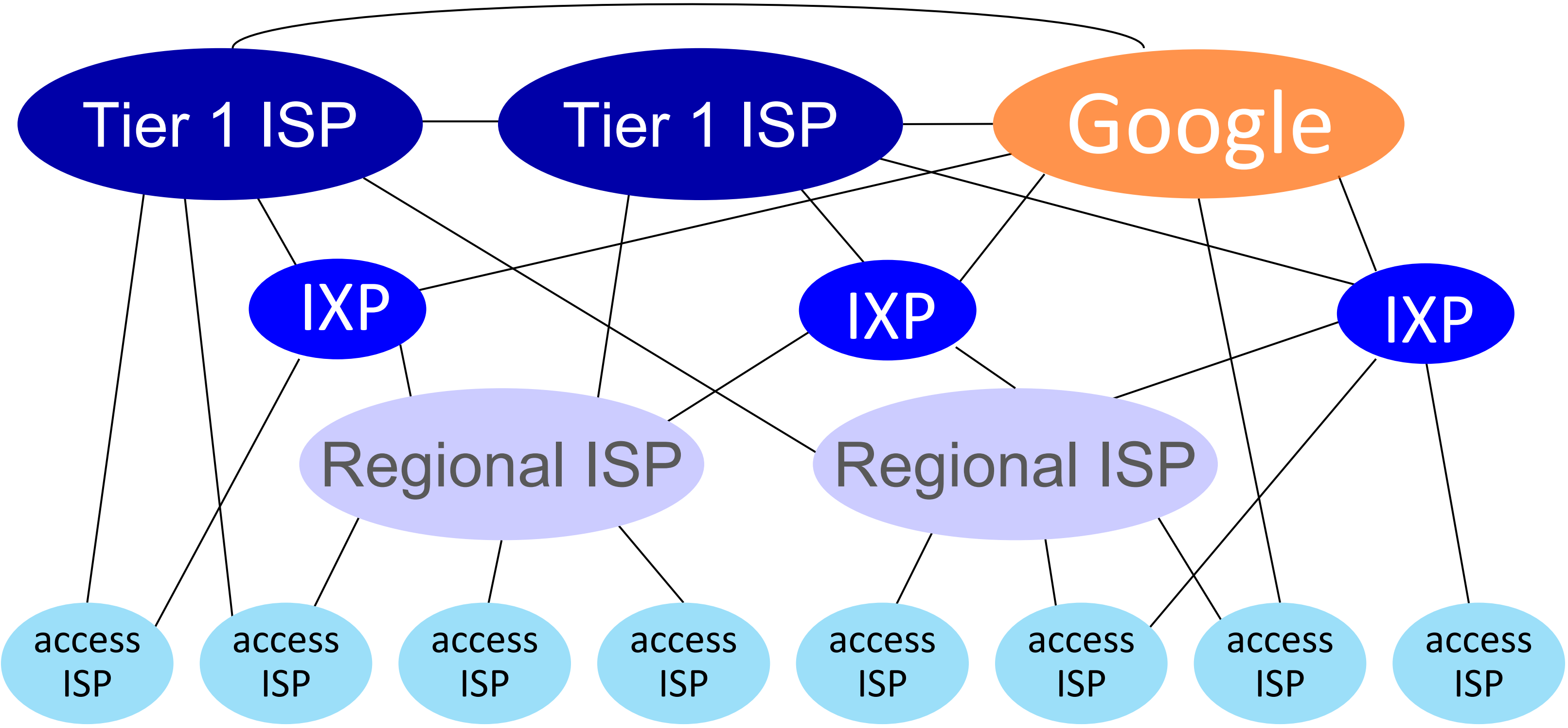


콘텐츠 제공자가 보다 최종 사용자에게 가까운 서비스를 제공하기 위해 자체 망생성

모든 ISP통합

Internet structure: a "network of networks"

ISP



At "center": small # of well-connected large networks

- "tier-1" commercial ISPs (e.g., Level 3, Sprint, AT&T, NTT), national & international coverage
- content provider networks (e.g., Google, Facebook): private network that connects its data centers to Internet, often bypassing tier-1, regional ISPs 다른 망을 통해연결 우회하다

Have you seen a system like this?

global scale
inclusive → inclusive all countries
open → run any application
greedy → everyone get money
low-cost
resilient(유연한)
scalable → more faster (확장가능한)
general

Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

How do packet delay and loss occur?

buffer에서 전송 대기

- packets *queue* in router buffers, waiting for turn for transmission
  - queue length grows when arrival rate to link (temporarily) exceeds output link capacity

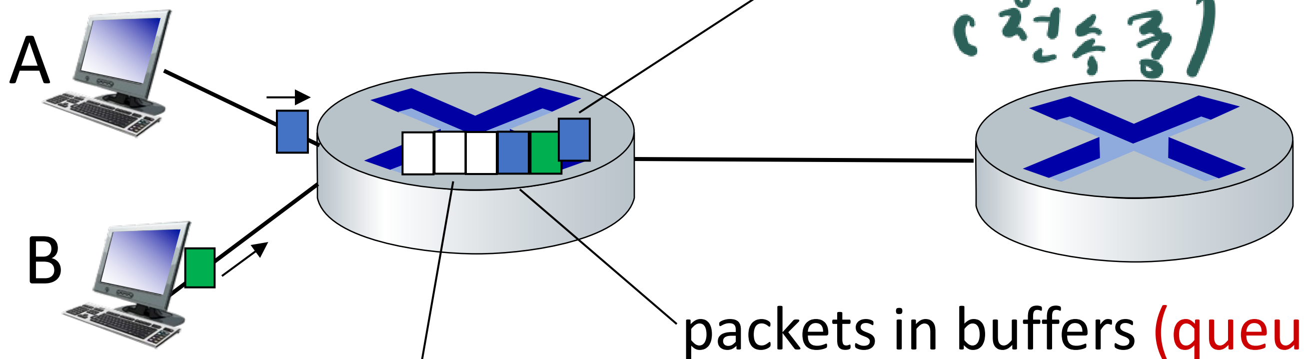
링크 도착 속도 > 출력 링크 용량 => 대기열 증가

- packet *loss* occurs when memory to hold queued packets fills up

ㄴ> buffer가 전부 차면 저장 공간 부족으로 loss 발생

packet being transmitted (transmission delay)

전송 대기 (전송 중)



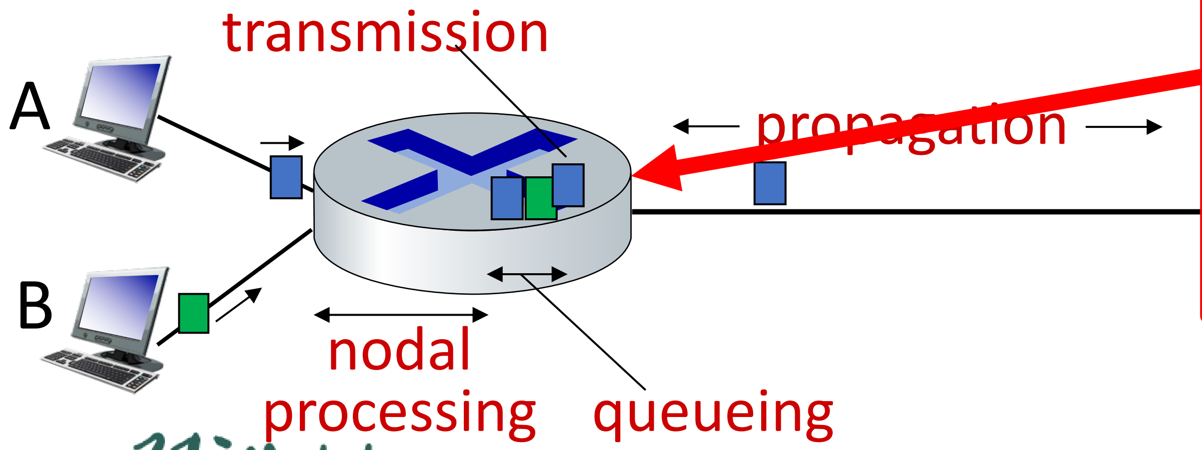
packets in buffers (queueing delay)

queue에서 대기

free (available) buffers: arriving packets dropped (loss) if no free buffers

남는 buffer 없으면 loss 발생

Packet delay: four sources



전체 delay

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

라우터에서 노드 처리 지연

d_{proc}: nodal processing

- check bit errors – bit 수준 오류 검사 (총합 이용)
- determine output link 출력링크 결정
- typically < microsecs

고속 라우터의 평균 delay는 수 마이크로초 이내

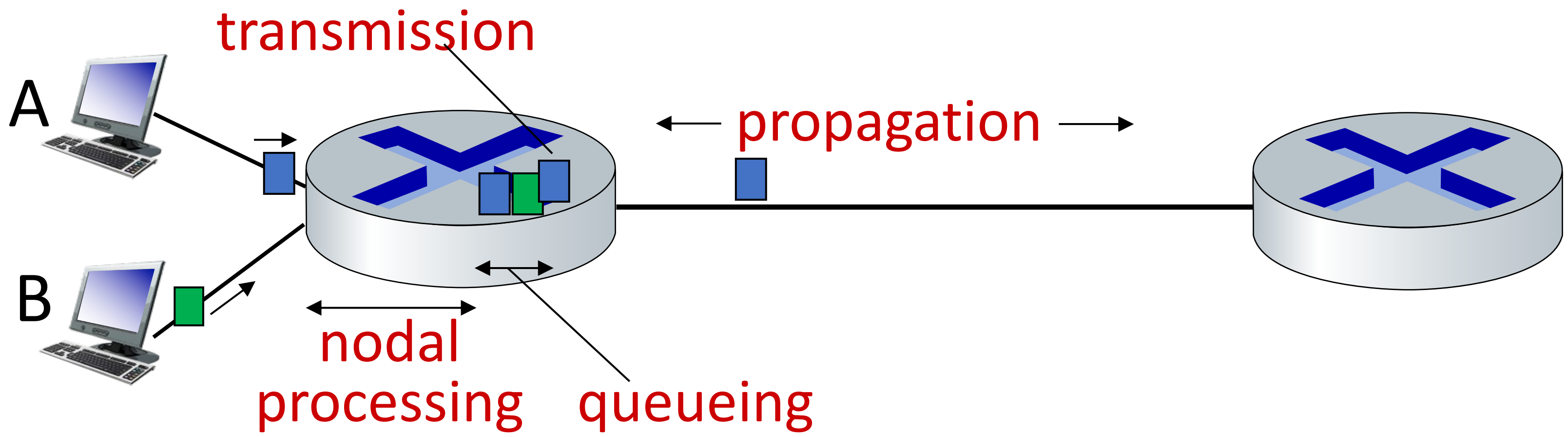
일반적으로 수밀리초~마이크로초

d_{queue}: queueing delay

- time waiting at output link for transmission 전송 대기 시간
- depends on congestion level of router 이미 큐에 저장된 패킷 수

라우터의 혼잡정도에 비례

Packet delay: four sources



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

전송 지연: 패킷을 네트워크로 전송

physical link를 따라 실제 전송

d_{trans}: transmission delay:
- L: packet length (bits)
- R: link *transmission rate (bps)*
- $d_{trans} = L/R$

d_{trans} and d_{prop} *very* different

d_{prop}: propagation delay:
- d: length of physical link
- s: propagation speed (~2x10^8 m/sec)
- $d_{prop} = d/s$

전파 속도 (매개별)

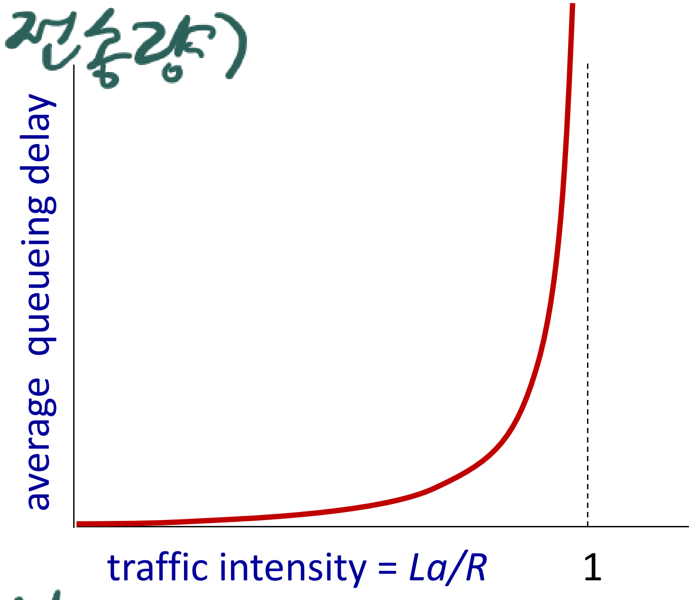
라우터간 거리 (물리적거리)

Packet queueing delay (revisited)

평균 도착 속도 (초당 전송량)

- *a:* average packet arrival rate
- *L:* packet length (bits) 패킷 길이
- *R:* link bandwidth (bit transmission rate) 링크 대역폭 (비트 전송 속도)

$$\frac{L \cdot a}{R} : \frac{\text{arrival rate of bits}}{\text{service rate of bits}}$$ ***"traffic intensity"***



대역폭: 단위 시간 동안 최대량을 전송하는 통신 링크 용량

- *La/R* ~ 0: avg. queueing delay small
  La<R
- *La/R* -> 1: avg. queueing delay large
  La=R
- *La/R* > 1: more "work" arriving is more than can be serviced - average delay infinite!
  → 도착이 전송보다 빠르므로 계속 적체 ⇒ loss 발생

대역폭↑ ⇒ 초당 전송량↑ ⇒ 전송 속도↑

La/R ~ 0

La/R -> 1

"Real" Internet delays and routes

(1) 100k 이상

- what do "real" Internet delay & loss look like?
- **traceroute** program: provides delay measurement from source to router along end-end Internet path towards destination. For all *i:*

  디버깅을 위해서 사용

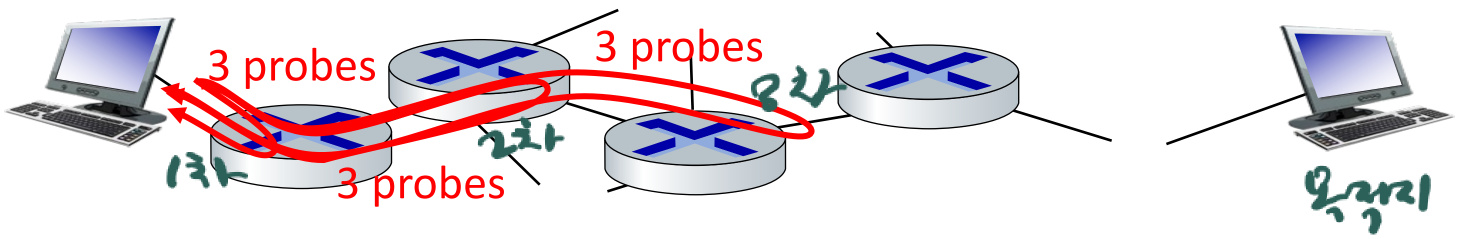
  라우터로 3개의 패킷 전송 => return을 받아 전체 delay 확인

  - sends three packets that will reach router *i* on path towards destination (with time-to-live field value of *i*)

    => 일정 시간이 지나면 error 메시지 리턴

    => i를 점차 늘려 다음 라우터의 ip 반환

  - router *i* will return packets to sender
  - sender measures time interval between transmission and reply



Real Internet delays and routes

traceroute: ISP (Singapore) to www.kaist.ac.kr

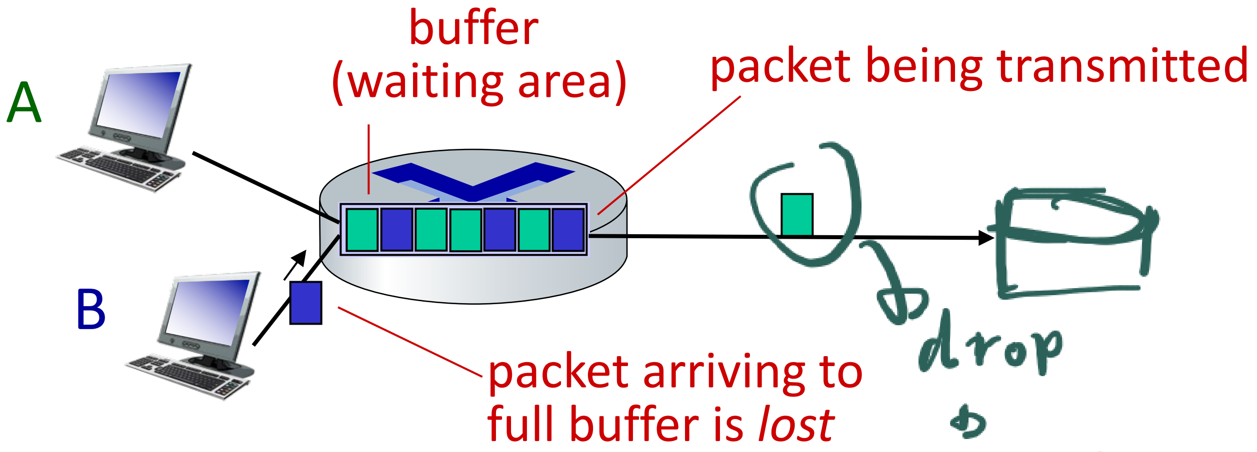
traceroute to www.kaist.ac.kr (143.248.155.65), 30 hops max, 60 byte packets
1 202.150.221.169 (202.150.221.169) 0.222 ms 0.252 ms 0.269 ms
2 10.15.62.214 (10.15.62.214) 0.297 ms 0.307 ms 0.316 ms
3 203.174.80.54 (203.174.80.54) 0.747 ms 1.255 ms 0.755 ms
4 203.208.182.254 (203.208.182.254) 0.875 ms 0.772 ms 0.894 ms
5 203.208.182.253 (203.208.182.253) 1.371 ms 203.208.171.190 (203.208.171.190) 32.856 ms 203.208.172.106 (203.208.172.106) 34.487 ms
6 203.208.178.18 (203.208.178.18) 40.844 ms 203.208.172.110 (203.208.172.110) 34.592 ms 203.208.152.74 (203.208.152.74) 34.505 ms
7 203.208.183.102 (203.208.183.102) 41.036 ms 40.994 ms 46.681 ms
8 112.174.91.61 (112.174.91.61) 94.857 ms 112.174.80.157 (112.174.80.157) 89.563 ms 203.208.183.102 (203.208.183.102) 45.944 ms
9 112.174.86.165 (112.174.86.165) 88.457 ms 112.174.80.157 (112.174.80.157) 89.602 ms 112.174.86.217 (112.174.86.217) 86.885 ms
10 112.174.86.217 (112.174.86.217) 86.952 ms 112.174.91.57 (112.174.91.57) 86.913 ms 112.188.134.130 (112.188.134.130) 101.694 ms
11 211.230.12.2 (211.230.12.2) 95.464 ms 112.188.135.134 (112.188.135.134) 96.173 ms 112.174.8.121 (112.174.8.121) 94.736 ms
12 112.188.134.130 (112.188.134.130) 103.271 ms 211.230.12.2 (211.230.12.2) 93.891 ms 143.248.117.5 (143.248.117.5) 94.649 ms
13 143.248.117.30 (143.248.117.30) 104.176 ms 143.248.117.5 (143.248.117.5) 95.448 ms 143.248.117.30 (143.248.117.30) 102.113 ms
14 112.188.135.134 (112.188.135.134) 92.817 ms 143.248.117.5 (143.248.117.5) 92.914 ms 143.248.117.62 (143.248.117.62) 98.498 ms
15 * 143.248.117.30 (143.248.117.30) 102.597 ms *
16 * * *
17 * * *
(skip)
29 * * *
30 * * *

실제 결과

packet을 보낼 때마다 조금씩 경로 달라짐

Packet loss

- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving to full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all



drop

A가 재전송?

라우터가 재전송? ⇒ router가 재전송

retransmission의 책임은 누가 지는가?

retransmission 항상 필요?

⇒ 반복되는 packet 이라면 생략 가능 (빈칸 유추가능)

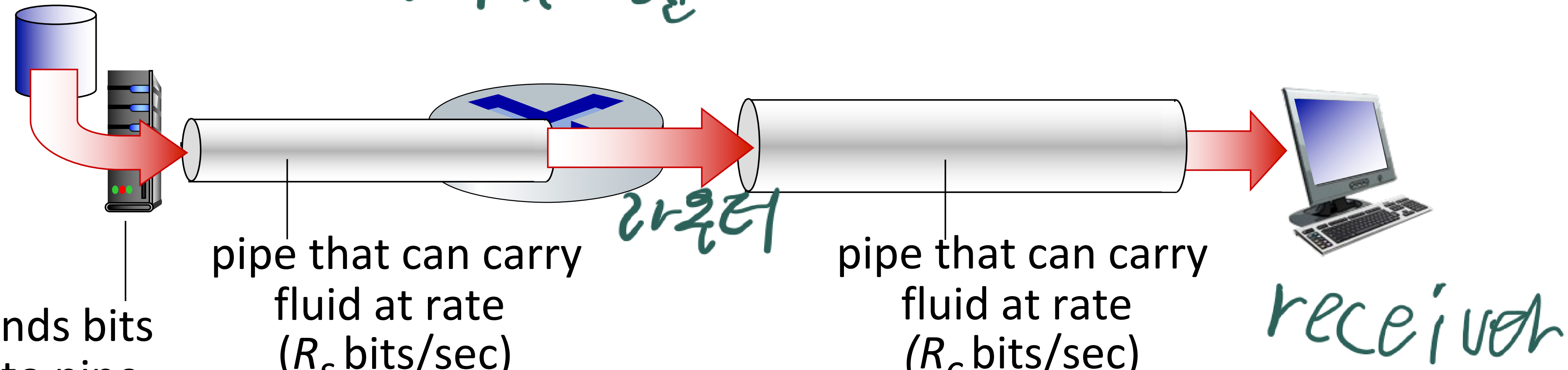
packet loss의 장점?

Throughput

sender와 receiver간 초당 전송량 = 전송 속도

- *throughput:* rate (bits/time unit) at which bits are being sent from sender to receiver → 특정 시점의 단기 관측 (순간 속도)
  - *instantaneous:* rate at given point in time
  - *average:* rate over longer period of time

→ 장기 관측 후 평균



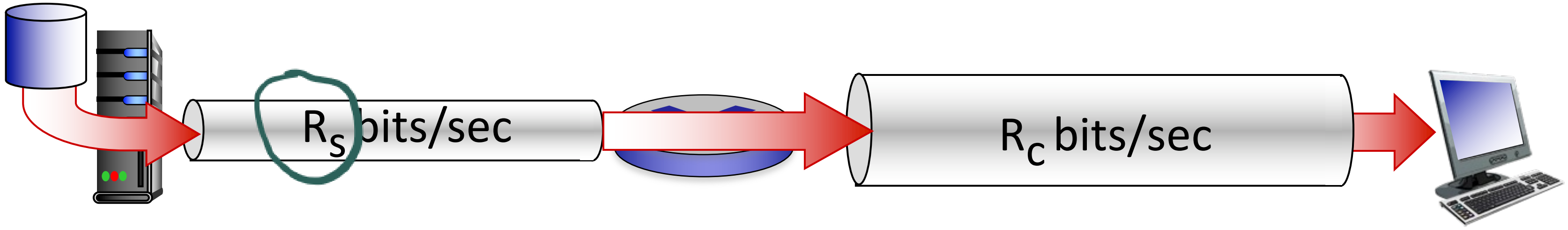
server sends bits (fluid) into pipe

pipe that can carry fluid at rate (R_s bits/sec)

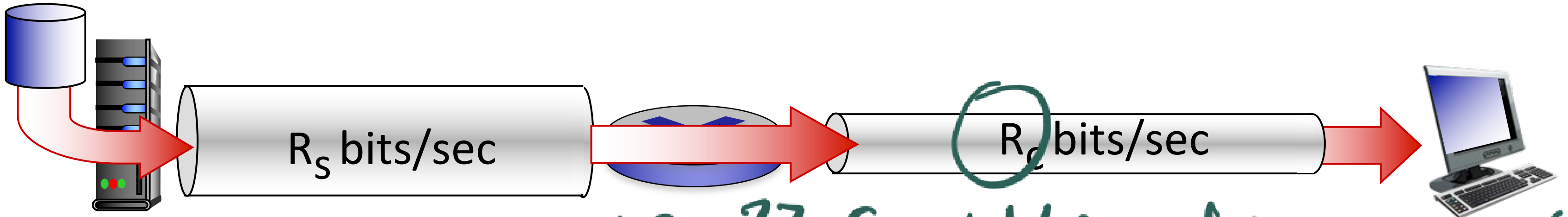
pipe that can carry fluid at rate (R_c bits/sec)

Throughput

$R_s < R_c$ What is average end-end throughput?



$R_s > R_c$ What is average end-end throughput?



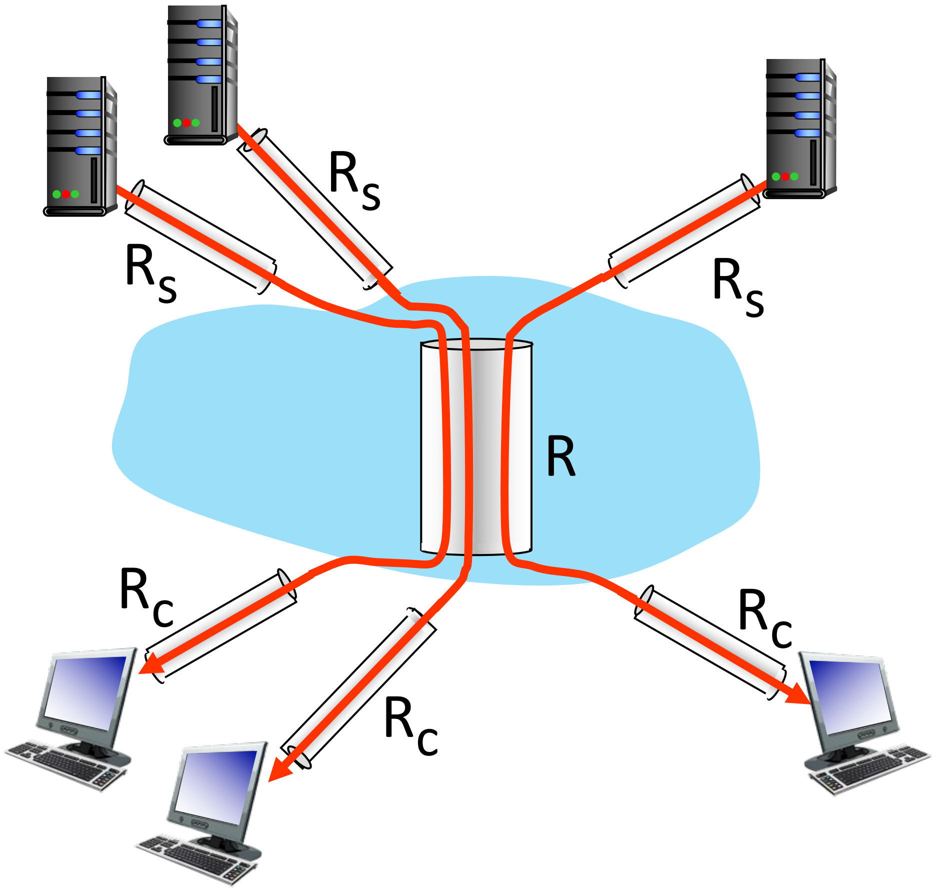
적은 쪽 (bottleneck)이 전체 속도 좌우

bottleneck link

link on end-end path that constrains end-end throughput

제한한다.

Throughput: network scenario



10 connections (fairly) share backbone bottleneck link *R* bits/sec

- per-connection end-end throughput: $min(R_c, R_s, R/10)$ → 각 연결당 대역폭
- in practice: R_c or R_s is often bottleneck

Should R > sum of all R_S

R > sum of all R_C ?

if $R < \Sigma R_s$ or $R < \Sigma R_c$ ⇒ 병목 현상 (R/10 throughput)

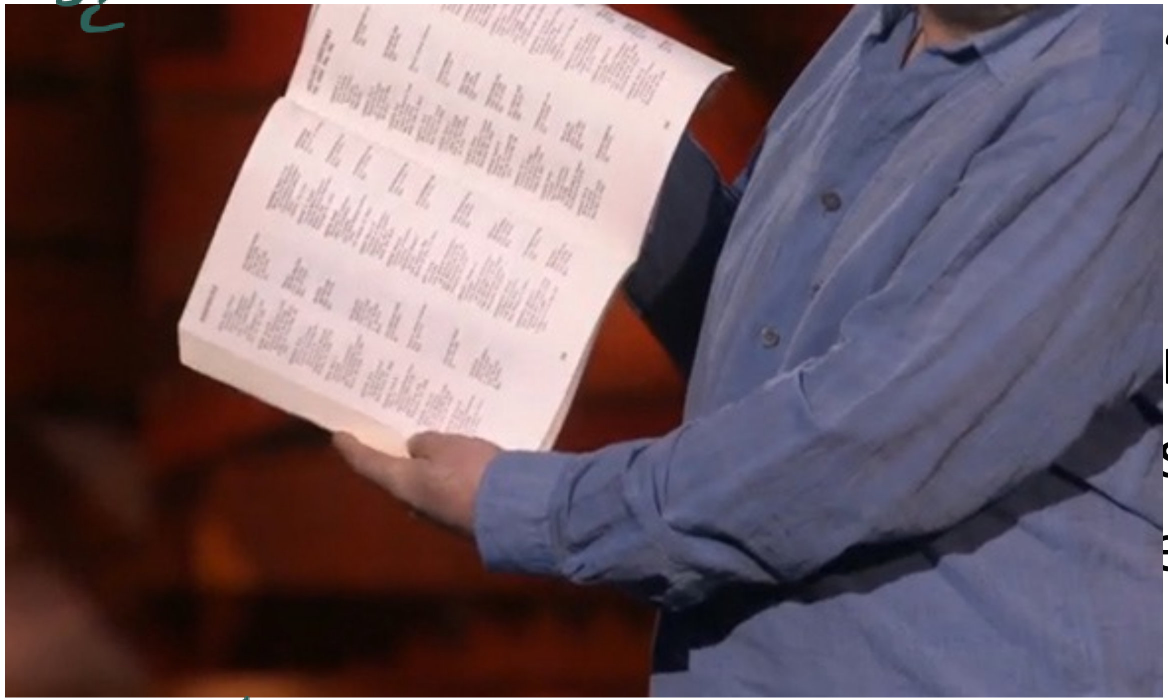
Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

Network security

- Internet not originally designed with (much) security in mind
  - *original vision:* “a group of mutually trusting users attached to a transparent network” ☺

투명한 상호 신뢰하는



“catch-up” (cat-and-mouse game)

모든 계층에서 보안 사항 고려

Is it good enough?

networks *공격 방법

st attacks *방어 방법

e immune to attacks *공격에 면역이 있는 구조 설계 방법

↳ 40년전, 인터넷 이용자들의 개인 정보 공유집 (일종의 전화 번호부)

Bad guys: packet interception

가로채기

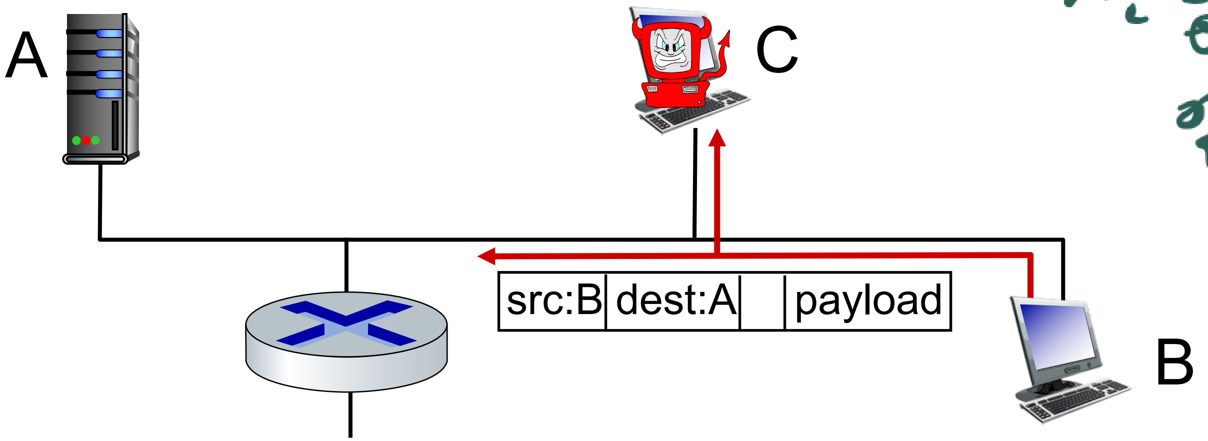
packet "sniffing":

- broadcast media (shared Ethernet, wireless)

무차별

- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by

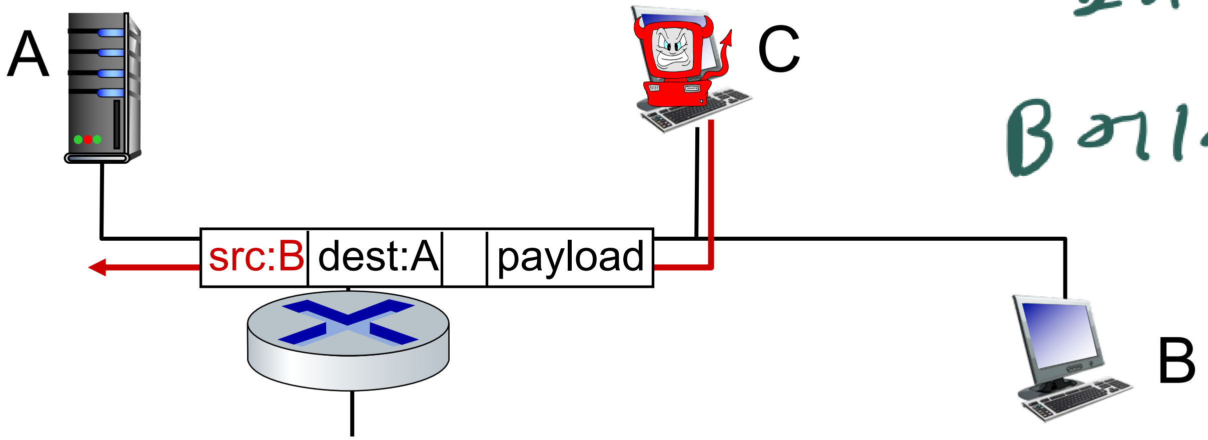
네트워크를 통해 전송되는 패킷을 가로채 읽기 & 저장



Bad guys: fake identity

IP spoofing: injection of packet with false source address

Source IP를 B로 수정한
패킷을 삽입해
B에서 요청을 보낸
것처럼 위조



Bad guys: denial of service

가짜 트래픽을 과도하게 보내

Denial of Service (DoS): attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

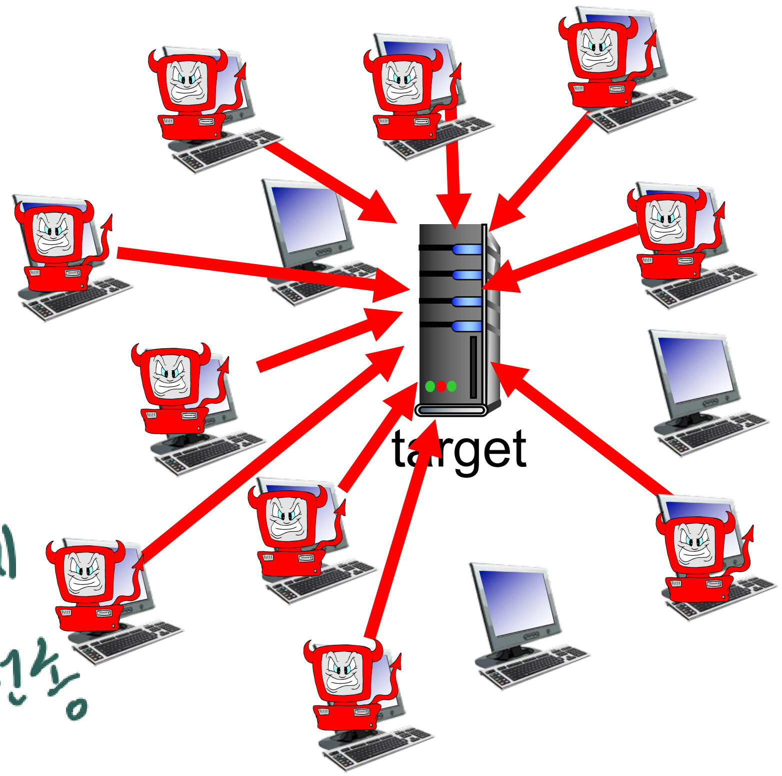
진짜 트래픽이 할당할 리소스 대신 차지

타겟 설정

1. select target
2. break into hosts around the network (see botnet)
3. send packets to target from compromised hosts

네트워크 근처의 host 침입

손상된 host를 이용해 타겟에 패킷 전송
(해킹당한)



Lines of defense:

인증

- authentication: proving you are who you say you are
  - cellular networks provides hardware identity via SIM card; no such hardware assist in traditional Internet ⇒ 셀룰러 네트워크의 SIM 카드가 [illegible]

기밀성

- confidentiality: via encryption 암호화

무결성 검사 변조

- integrity checks: digital signatures prevent/detect tampering

접근 제한

- access restrictions: password-protected VPNs

방화벽

- firewalls: specialized "middleboxes" in access and core networks:
  코어 네트워크 및 엑세스 전에 특별한 middlebox 추가
  들어오는 패킷을 필터링 ⇒ 송·수신자, 어플리케이션 제한
  - off-by-default: filter incoming packets to restrict senders, receivers, applications
  - detecting/reacting to DOS attacks

... lots more on security (throughout, Chapter 8)

Chapter 1: roadmap

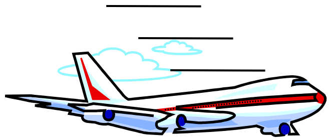
- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

Protocol "layers" and reference models

Networks are complex, with many "pieces":
- hosts
- routers
- links of various media
- applications
- protocols
- hardware, software

Question: is there any hope of *organizing* structure of network?
- and/or our *discussion* of networks?

Example: organization of air travel



end-to-end transfer of person plus baggage

ticket (purchase)	ticket (complain)
baggage (check)	baggage (claim)
gates (load)	gates (unload)
runway takeoff	runway landing
airplane routing	airplane routing

airplane routing

How would you *define/discuss* the *system* of airline travel?

- a series of steps, involving many services

Example: organization of air travel

layers: each layer implements a service

- via its own internal-layer actions
- relying on services provided by layer below

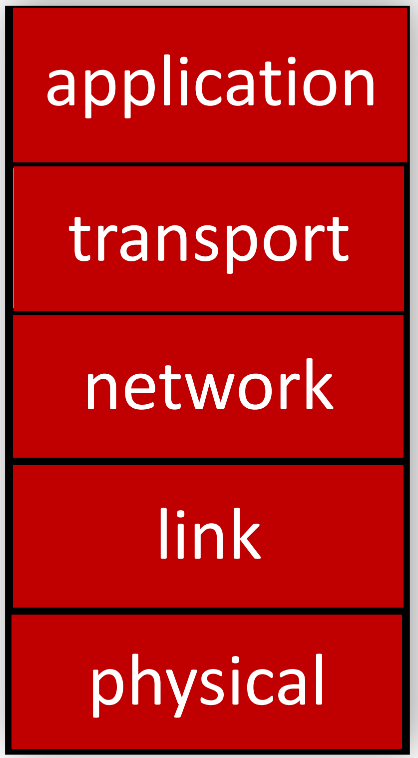
Why layering?

Approach to designing/discussing complex systems:

- explicit structure allows identification, relationship of system’s pieces
  - layered *reference model* for discussion
- modularization eases maintenance, updating of system
  - change in layer's service *implementation*: transparent to rest of system
  - e.g., change in gate procedure doesn’t affect rest of system

Layered Internet protocol stack

- *application:* supporting network applications
  - HTTP, IMAP, SMTP, DNS
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi), PPP
- *physical:* bits "on the wire"



Services, Layering and Encapsulation

source
application
transport
network
link
physical

source

M

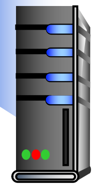
Application exchanges messages to implement some application service using *services* of transport layer

H_t M

Transport-layer protocol transfers M (e.g., reliably) from one *process* to another, using services of network layer

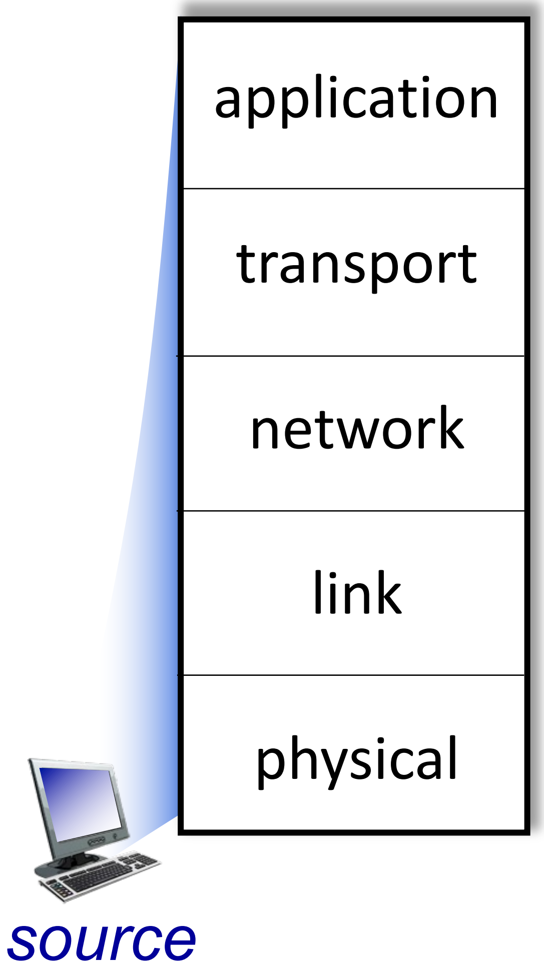
- transport-layer protocol encapsulates application-layer message, M, with *transport* layer-layer header H_t to create a transport-layer segment
  - H_t used by transport layer protocol to implement its service

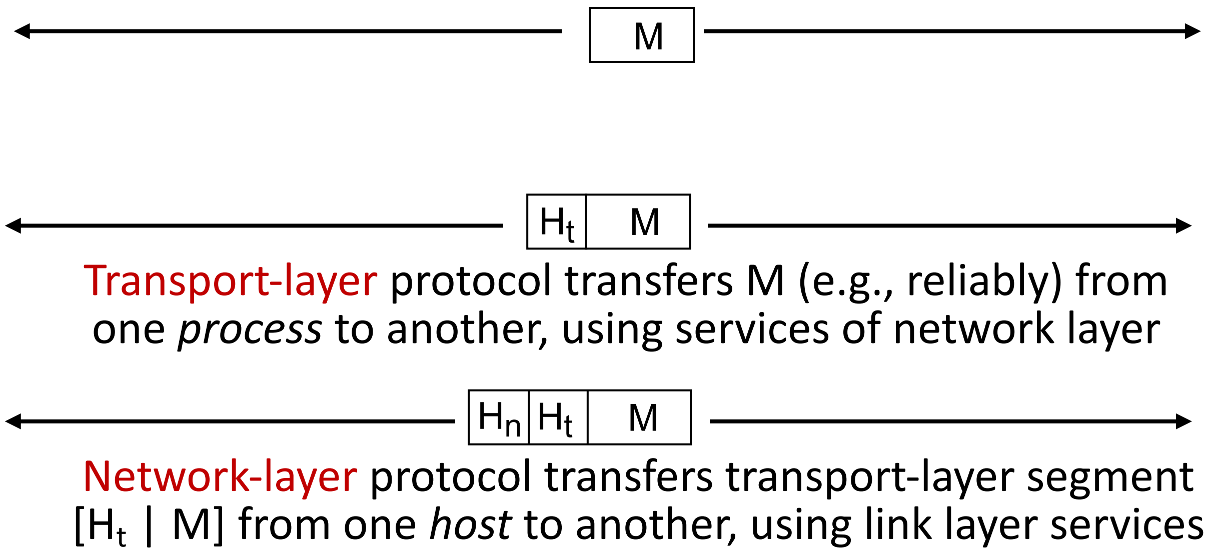
destination
application
transport
network
link
physical

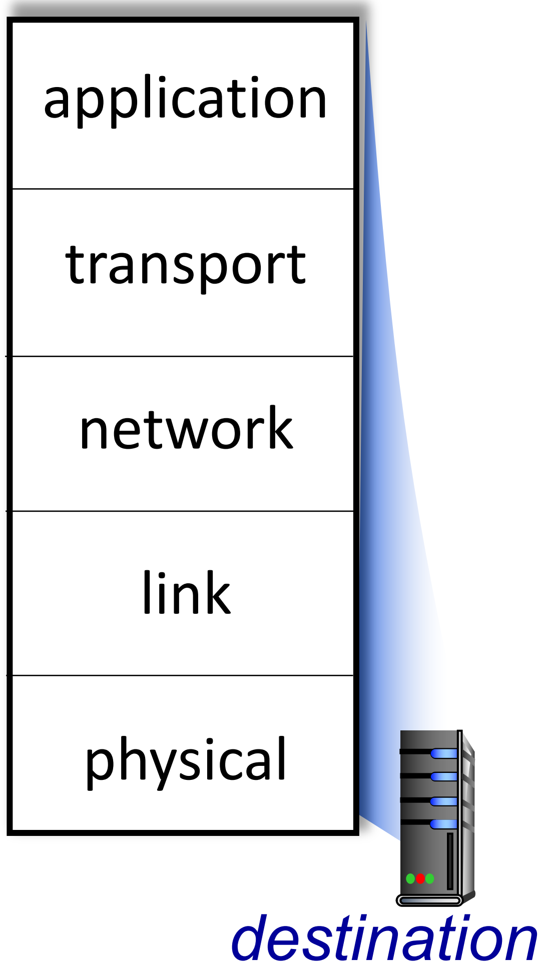


destination

Services, Layering and Encapsulation

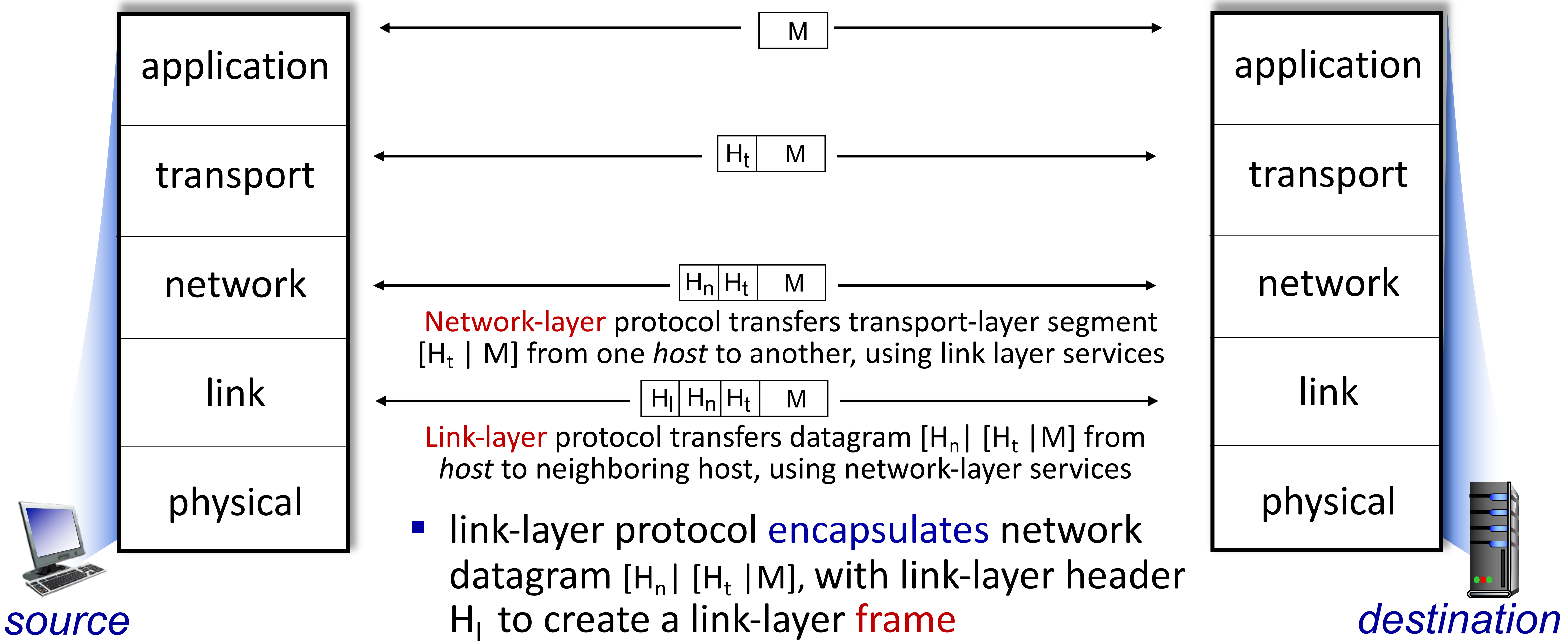






- network-layer protocol encapsulates transport-layer segment [H_t | M] with network layer-layer header H_n to create a network-layer datagram
  - H_n used by network layer protocol to implement its service

Services, Layering and Encapsulation

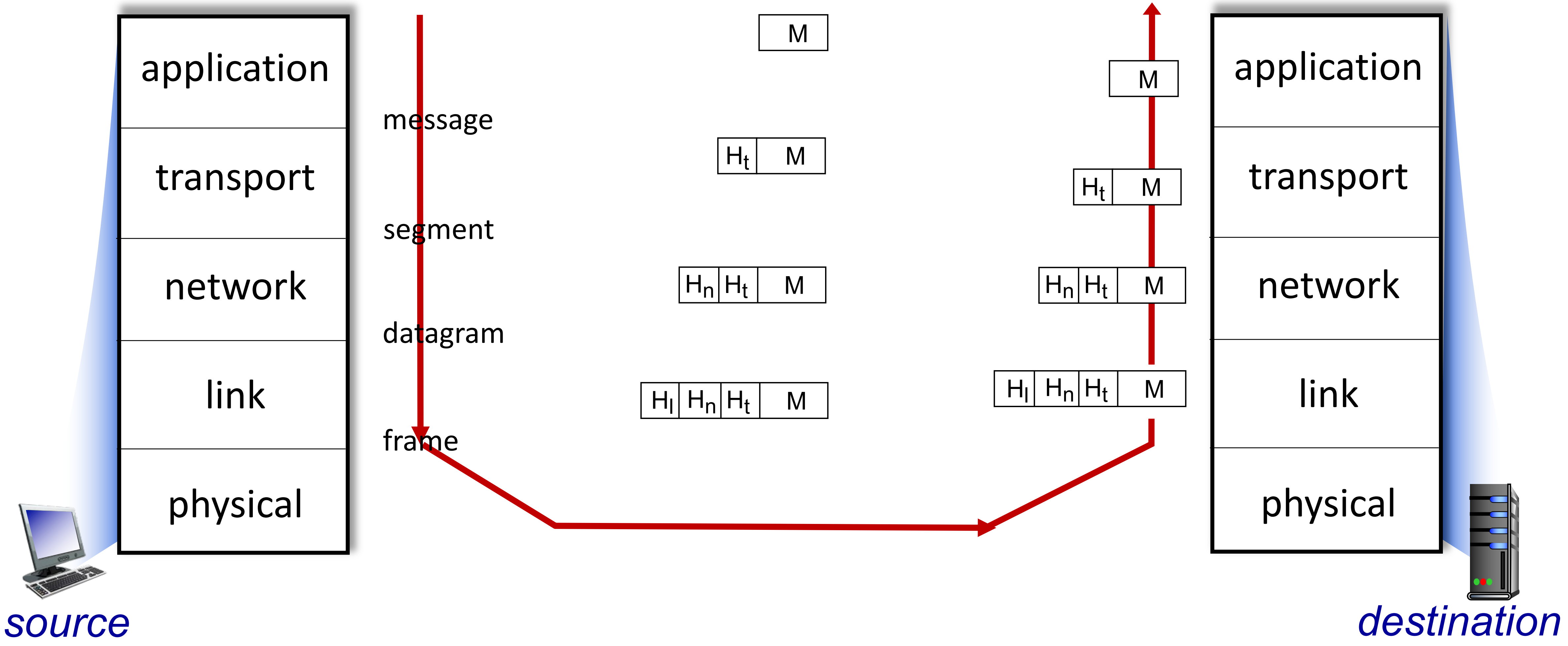


Network-layer protocol transfers transport-layer segment $[H_t \mid M]$ from one *host* to another, using link layer services

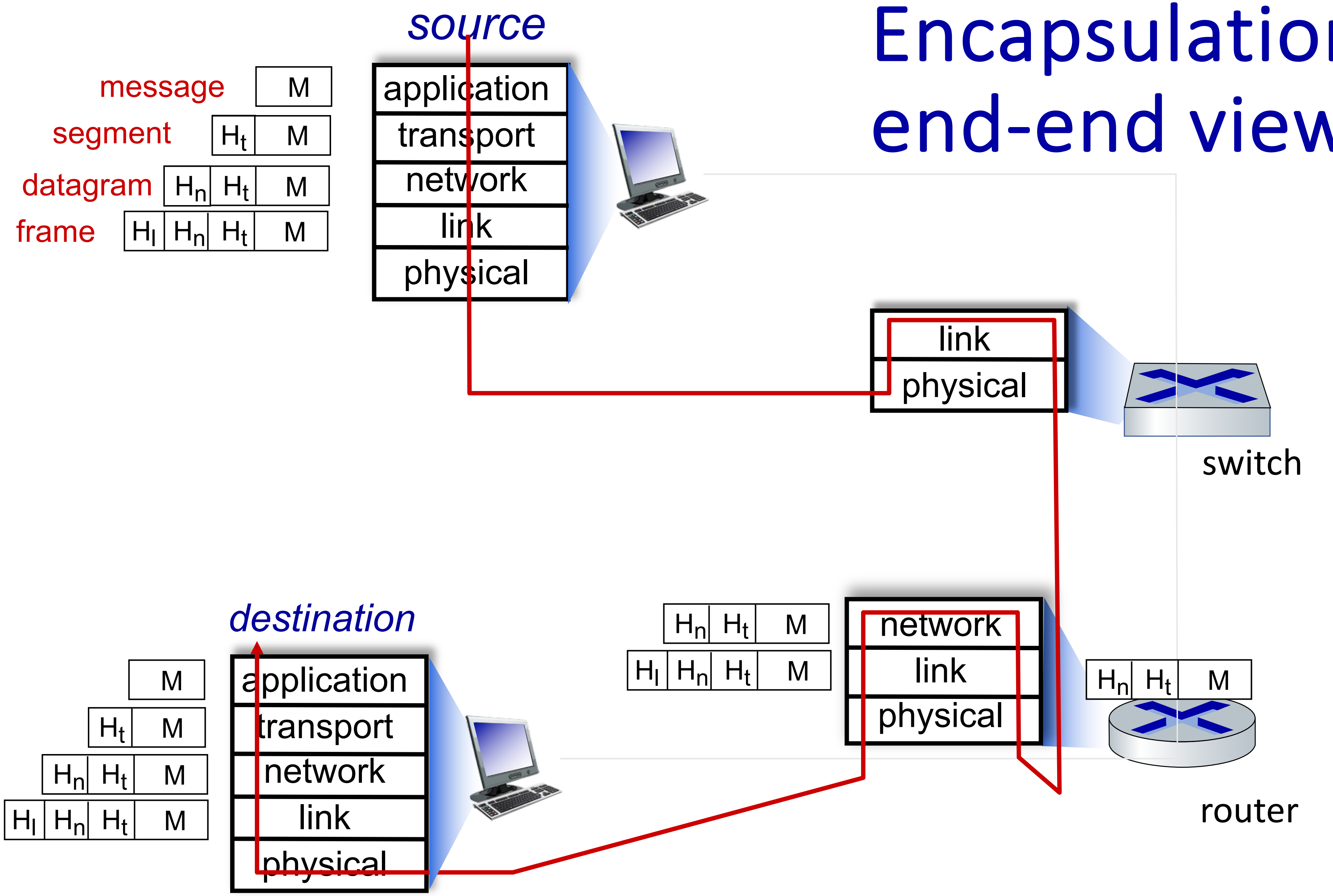
Link-layer protocol transfers datagram $[H_n| [H_t |M]$ from *host* to neighboring host, using network-layer services

- link-layer protocol encapsulates network datagram $[H_n| [H_t |M]$, with link-layer header H_l to create a link-layer frame

Services, Layering and Encapsulation



Encapsulation: an end-end view



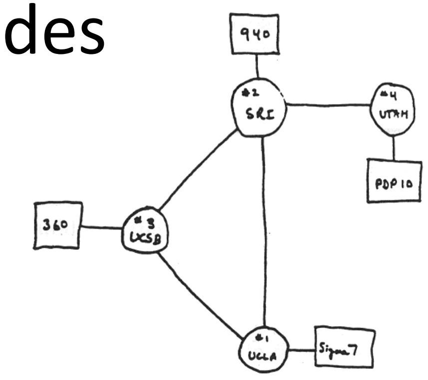
Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Security
- Protocol layers, service models
- History

Internet history

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational
- 1972:
  - ARPAnet public demo
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes



THE ARPA NETWORK

Internet history

1972-1980: Internetworking, new and proprietary networks

- 1970: ALOHAnet satellite network in Hawaii
- 1974: Cerf and Kahn - architecture for interconnecting networks
- 1976: Ethernet at Xerox PARC
- late70's: proprietary architectures: DECnet, SNA, XNA
- 1979: ARPAnet has 200 nodes

Cerf and Kahn's internetworking principles:

- minimalism, autonomy - no internal changes required to interconnect networks
- best-effort service model
- stateless routing
- decentralized control

define today's Internet architecture

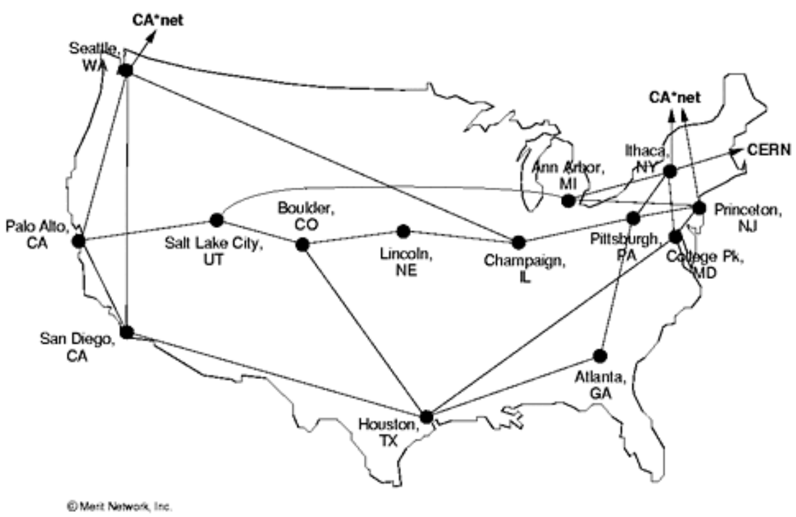
Internet history

1980-1990: new protocols, a proliferation of networks

- 1983: deployment of TCP/IP
- 1982: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1988: TCP congestion control

- new national networks: CSnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks

NSFNET T1 Network 1991



Internet history

1990, 2000s: commercialization, the Web, new applications

- early 1990s: ARPAnet decommissioned
- 1991: NSF lifts restrictions on commercial use of NSFnet (decommissioned, 1995)
- early 1990s: Web
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
  - late 1990s: commercialization of the Web

late 1990s – 2000s:

- more killer apps: instant messaging, P2P file sharing
- network security to forefront
- est. 50 million host, 100 million+ users
- backbone links running at Gbps

Internet history

2005-present: scale, SDN, mobility, cloud

- aggressive deployment of broadband home access (10-100's Mbps)
- 2008: software-defined networking (SDN)
- increasing ubiquity of high-speed wireless access: 4G/5G, WiFi
- service providers (Google, FB, Microsoft) create their own networks
  - bypass commercial Internet to connect "close" to end user, providing "instantaneous" access to social media, search, video content, …
- enterprises run their services in "cloud" (e.g., Amazon Web Services, Microsoft Azure)
- rise of smartphones: more mobile than fixed devices on Internet (2017)
- ~18B devices attached to Internet (2017)

Homework

인터넷의 역사+보안
⇒시간 될때 보기

TED Ideas worth spreading

WATCH

– 인터넷의 아버지

1,414,329 views | Danny Hillis • TED2013

Like (42K) Share Add

The Internet could crash. We need a Plan B

Read transcript

The Internet connects billions of people and machines; it's the backbone of modern life. But tech pioneer Danny Hillis thinks the Internet just wasn't designed to grow this big -- and he fears that one big cyber-attack or glitch could shut it down and take civilization with it. To head off a digital dark age, he sounds a clarion call to develop a Plan B: a parallel system to fall back on if -- or when -- the Internet crashes.

- https://www.ted.com/talks/danny_hillis_the_internet_could_crash_we_need_a_plan_b

Chapter 1: summary

We’ve covered a “ton” of material!

- Internet overview
- what’s a protocol?
- network edge, access network, core
  - packet-switching versus circuit-switching
  - Internet structure
- performance: loss, delay, throughput
- layering, service models
- security
- history

You now have:

- context, overview, vocabulary, “feel” of networking
- more depth, detail, *and fun* to follow*!*

Next…

- *Chapter 2.1 Principles of Network Applications*
- *Chapter 2.2 The Web and HTTP*
- *Chapter 2.3 Electronic Mail in the Internet*