Class Notes

Any questions from the last lecture or the textbook chapter 1?

First sessional syllabus: First two chapters of the textbook

Textbook reading: Please finish whole chapter 1 by 31st August

Student presentations: We start in the next class.

When done reading section 1.1 of book, see video: https://youtu.be/74sEFYBBRAY
When done reading section 1.2 of book, see video: https://youtu.be/k8NmM-hImBU
When done reading section 1.3 of book, see video: https://youtu.be/f1nUcCdQJ8Y
When done reading section 1.4 of book, see video: https://youtu.be/hm1y4LsphQQ
When done reading section 1.5 of book, see video: https://youtu.be/IZ_PnVXtMeY
When done reading section 1.6 of book, see video: https://youtu.be/yukwBqSwAkg
When done reading section 1.7 of book, see video: https://youtu.be/l4q8iuqbuiQ
When done reading section 1.8 of book, see video: https://youtu.be/-YaGGf8C1A4

Connectivity within FAST: L2-switched or L3-switched:

Did you get a chance to conduct the experiment we discussed last time regarding FAST lab computers connectivity?

What was the result?

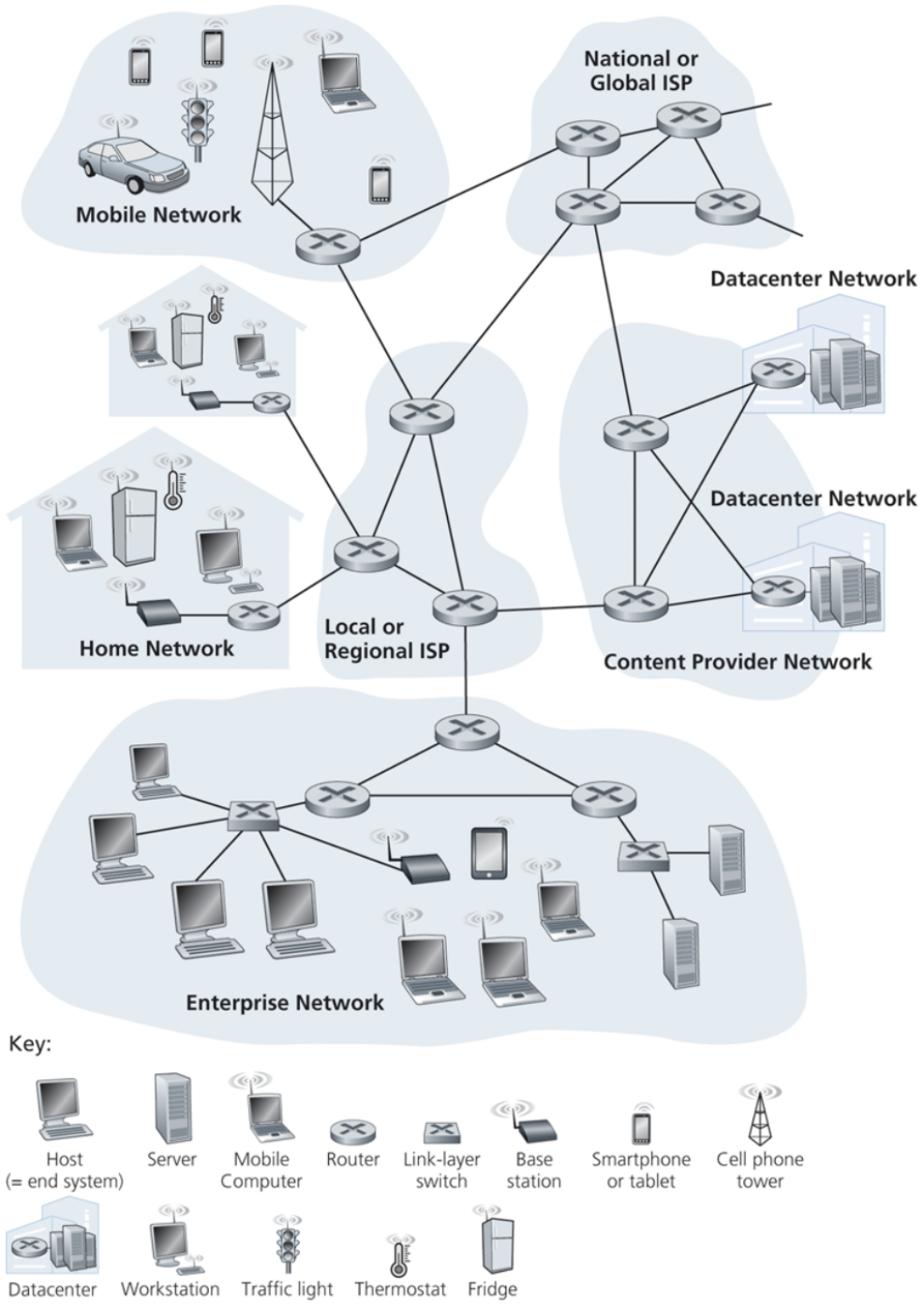
Revision From Last Time:

When we say, **"a hop"**, we usually mean an L3-hop. (When we will mean L2-hops, either it will be clear from the context or we will explicitly say so.) So, the responsibility of the network layer (the IP in the case of the Internet), is **host-to-host global delivery considering one L3-hop at a time**.

How will a packet traverse one L3-hop? It will do so using the local L2 network (that might use many L2 hops to deliver packet to the next L3 router). Usually, L2 switches do not change packet contents. They just observer it (for example to read the L2 destination for L2-level delivery). That is why it is usually quite hard to see an L2 switche like we were able to see L3-switches (routers) because each router does changes in the packet header (for example reducing the hop-count by one, and re-calculating some checksums because header changed due to hop-count reduction).

The transport layer can provide ordered delivery at the final destination (layers above network layer are not involved within the core of the network), and at the destination hands out the data to an appropriate process (running at the destination host) as per the directions of transport-peer (at the source of the data.).

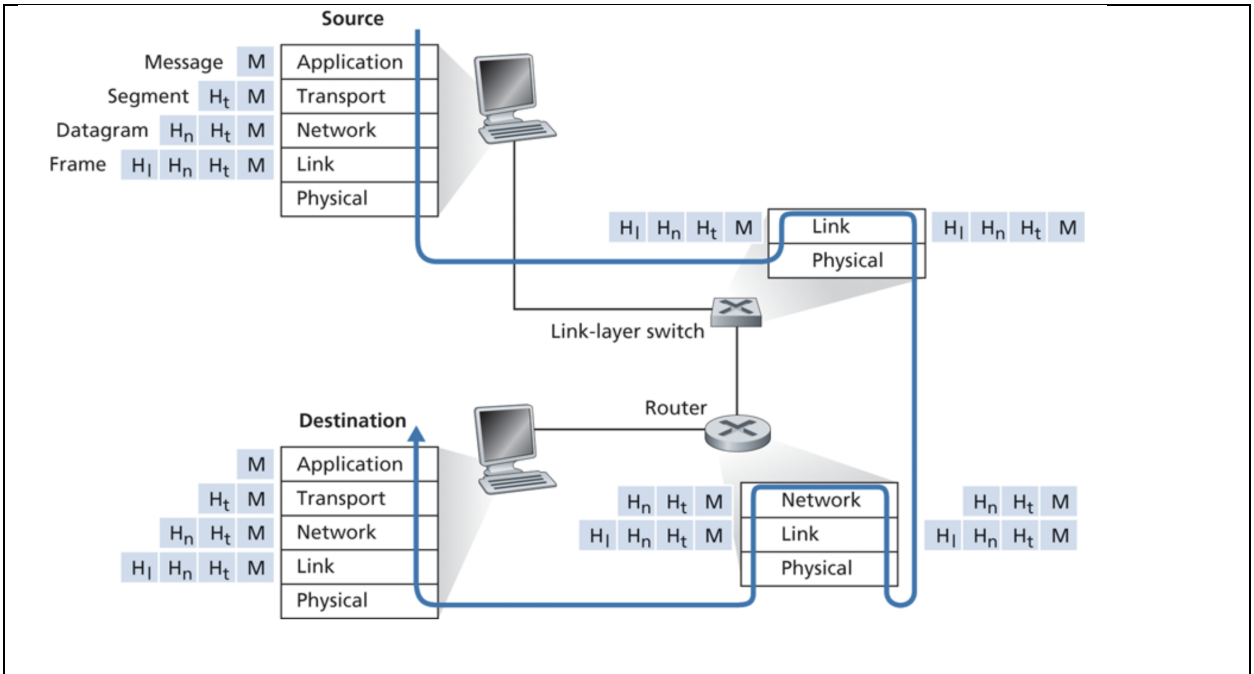
Layer-2 Switch VS Layer-3 Switch (Direct delivery VS Delivery via the IP):



Source: Figure 1.1 from the textbook

Encapsulation and its overhead:

Service model: To provide service to layer (n+1), layer n does some work, and uses the services provided by the layer (n-1).



Source: Figure 1.24 from the textbook

We are tagging on so many headers. Those headers add to the length of the overall data to transmit. But from the perspective of the application, headers (and any footers) are pure overhead (albeit a necessary one!). Generally, we would want this overhead to be as low as possible. For example, if we could send out 1500 Bytes packet as the largest packet on some link, and the sum of all headers and footers is 80 Bytes, then the overhead is about **~6% per packet**.

Some protocols like secure shell (ssh) usually send data as character by character (or whatever characters it has accumulated in a very short time) to the server, and server echoes it back to client (that is why at times when you ssh to a server far away, due to excessive propagation or queuing delays, it feels very slow and annoying!). The reason is that ssh does not know how to interpret characters or how many are needed at the server side. (You might be reading a remote man page and going through it line by line by pressing down arrow. So, if ssh did not send data as character by character, you will not be able to see new lines interactively!)

Network stack is not “turtles all the way down!”

CS-students (which are usually far away from the link-layer/physical layers), at times consider networking layers as “turtles all the way down”.

The “buck” stops at the physical layer! A physical layer must send or receive data. And that is why we are starting student presentations with the physical layer (so that you know its not turtles all the way down!).

Turtles all the way down

7 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

This article is about the saying. For other uses, see Turtles All the Way Down.

"**Turtles all the way down**" is an expression of the problem of infinite regress. The saying alludes to the mythological idea of a World Turtle that supports a flat Earth on its back. It suggests that this turtle rests on the back of an even larger turtle, which itself is part of a column of increasingly larger turtles that continues indefinitely.

The saying holds that the world is supported by an infinite stack of increasingly larger turtles.

The exact origin of the phrase is uncertain. In the form "rocks all the way down", the saying appears as early as 1838.[1] References to the saying's mythological antecedents, the World Turtle and its counterpart the World Elephant, were made by a number of authors in the 17th and 18th centuries.[2][3]

The expression has been used to illustrate problems such as the regress argument in epistemology.

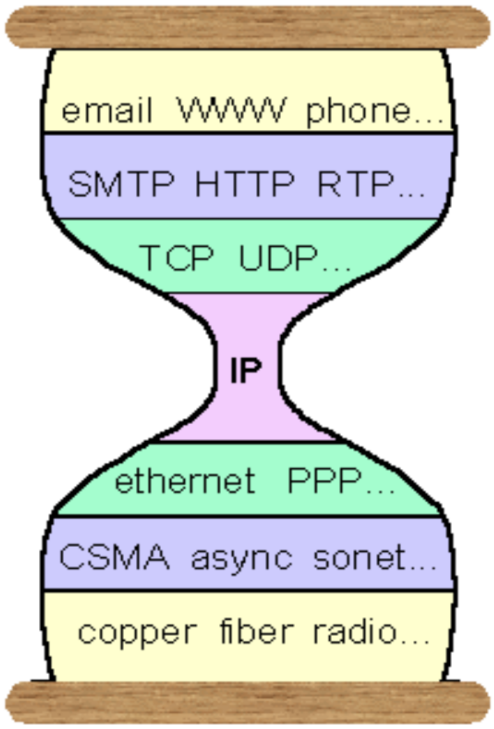
History [edit]

Background in Hindu mythology [edit]

The narrow waist of the Internet:

The IP layer (network layer) is special in the TCP/IP stack, and it is one of the major reasons for the Internet’s success.

- If you have a brilliant app idea, you don’t have to get anyone’s permission. You can just build your app (client/server etc.) and try it out. Internet will route your app’s data. No change needed in the network core!
- If you are a hardware vendor and making a new kind of network. Great you are free to do so, as long as your new network technology is able to carry through the IP packets.
- Huge innovation on both ends of the spectrum (apps, hardware) due to stability provided by the IP layer.



Source: Steve Deering Presentation: Watching the narrow waist of the Internet
https://www.ietf.org/proceedings/51/slides/plenary-1/sld002.htm

More about Delays (Latency)

Processing delays:

- Time taken by a router to decide on which egress (outgoing) port a packet should go
- Difference between **routing and forwarding**

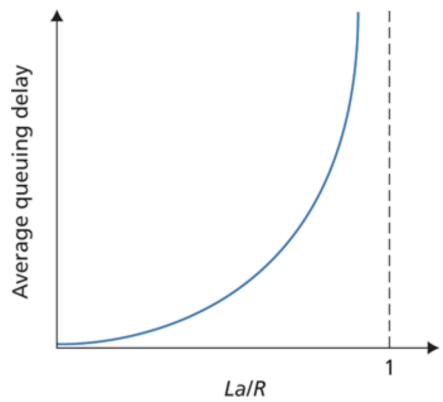
Queuing delays:

- A huge topic for research, and a body of study called **queuing theory**
    - Applicability at a lot of places in real life (supermarket check outs, bank tellers, roads, networks, etc.)
    - How big should be routers' buffers?

- Traffic intensity:
    - Let **a** be the average rate of packet arrival at some router in packets per second

- Let **R** be the transmission rate in bits per second
- Let's assume (for simplicity) that all packets coming at the router are of equal size, size of each packet being **L** bits
- Average rate of arrival of bits at the router = **L*a bits per second**
- Ratio: L*a/R is traffic intensity
  - If traffic ratio is consistently > 1 (meaning more data coming in than a router can process and transmit (push out)), it is a very bad situation! Buffers will fill up and data will be lost!
  - If L*a/R <= 1, then queuing delay depends if traffic is periodic (say a packet comes in every L/R seconds) or as a burst of traffic. A burst will experience more delays.
- In real life traffic is random and traffic intensity alone is usually not sufficient to understand or estimate delays. But still traffic intensity is a good approximation to understand queuing delay.

Figure 1.18 **Dependence of average queuing delay on traffic intensity**



Source: Textbook figure 1.18

Queueing delays shoot up rapidly as traffic intensity approaches 1

- In the above simple model, we assumed buffers at the router has infinite capacity. But router buffers are of finite size (again an area of research that what is an appropriate size for a router buffer). Once buffer fills up, packet loss is inevitable.

Space-time diagrams for end-to-end latency:

Problem: Compare the total delay in sending **x bits** of user data over a **k-hop path** in a circuit-switched network and a packet-switched network. Here, the total delay is defined to be the time from the data starts to be delivered to the network till all the data bits have been received at the end of the path. For the packet-switched network, it is assumed that the network is lightly loaded and the queuing delay at each switching node is ignored, and in addition, the packet header size is ignored. For both networks, there is no loss. The circuit setup time is **s second**, the **propagation delay is d second per hop**, each **packet contains p data bits**, and the **bit rate of the line on each hop is b bps**.

1) What is the total delay, if the message is sent over the circuit-switched network?

2) What is the total delay, if the message is sent over the packet-switched network?

3) Under what condition does the packet-switched network have a lower delay?

Solution:

Some info to jog your memory about circuit and packet switching:

Circuit switching is a switching technique for communication networks. Circuit switching creates a direct physical connection/path between two devices. The transmission capacity on the path is exclusively reserved for the connection.

Packet switching is a switching technique for communication networks. In packet switching, each packet has a header providing an address to identify the destination. In the network, packets are switched in the **store-and-forward** manner, i.e., at each node, packets are received and stored, before being forwarded to the next hop.

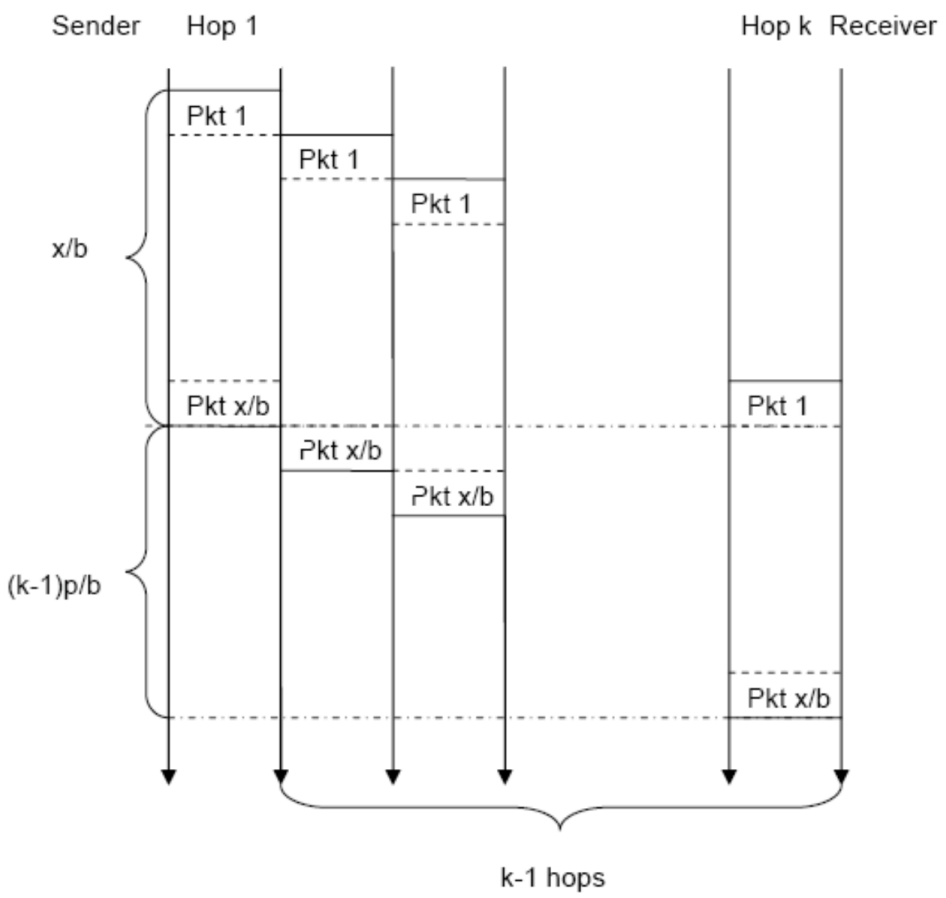
(i) Circuit switching requires a connection setup, while packet switching may not have the setup process.
(ii) Packet switching uses store-and forward transmission, while with circuit switching, the bits just flow through the path continuously.
(iii) Circuit switching is completely transparent to the sender and receiver, and they can use any bit format or framing method they want to. But, with packet switching, packets have special formats in assembling bits or frames.
(iv) In circuit switching, the transmission capacity on a path is dedicated to the corresponding connection, while in packet switching, the transmission capacity on a link is shared by packets from different connections.

Source: University of Utah: https://my.ece.utah.edu/~ece6962-003/exams/Midterm%201/practice%20exam%201-solutions.pdf

Time for circuit switching: s + kd + x/b in seconds

Time for packet switching: kd + x/b + (k-1)*p/b [Remember: Packet switches are doing store-and-forward]

(See https://my.ece.utah.edu/~ece6962-003/exams/Midterm%201/practice%20exam%201-solutions.pdf for details)



Note: In the above picture, propagation delay is not shown.

Food for thought:

(1) In above example, how end-to-end time will change if first bit haven't yet reached the destination but the last bit of data has been transmitted from the source?
(2) By using one or two hops, see does the general formulae above make sense?

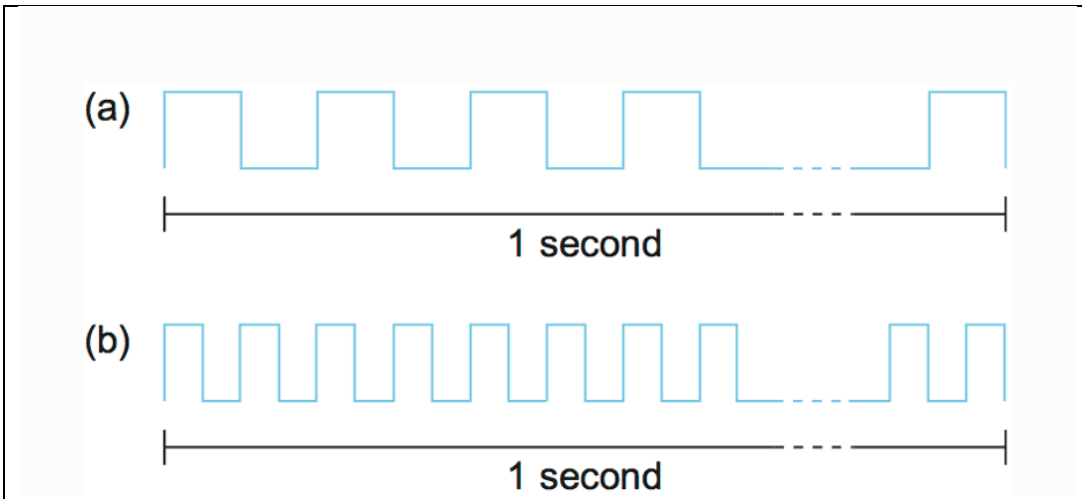


Figure 16 from Computer Networks: A systems approach. *Bits transmitted at a particular bandwidth can be regarded as having some width: (a) bits transmitted at 1 Mbps (each bit is 1 microsecond wide); (b) bits transmitted at 2 Mbps (each bit is 0.5 microseconds wide).*

Physical Link Multiplexing:

Multiplexing means that a system resource is shared among multiple users.

At an intuitive level, multiplexing can be explained by analogy to a time-sharing computer system, where a single physical processor is shared (multiplexed) among multiple jobs, each of which believes it has its own private processor.

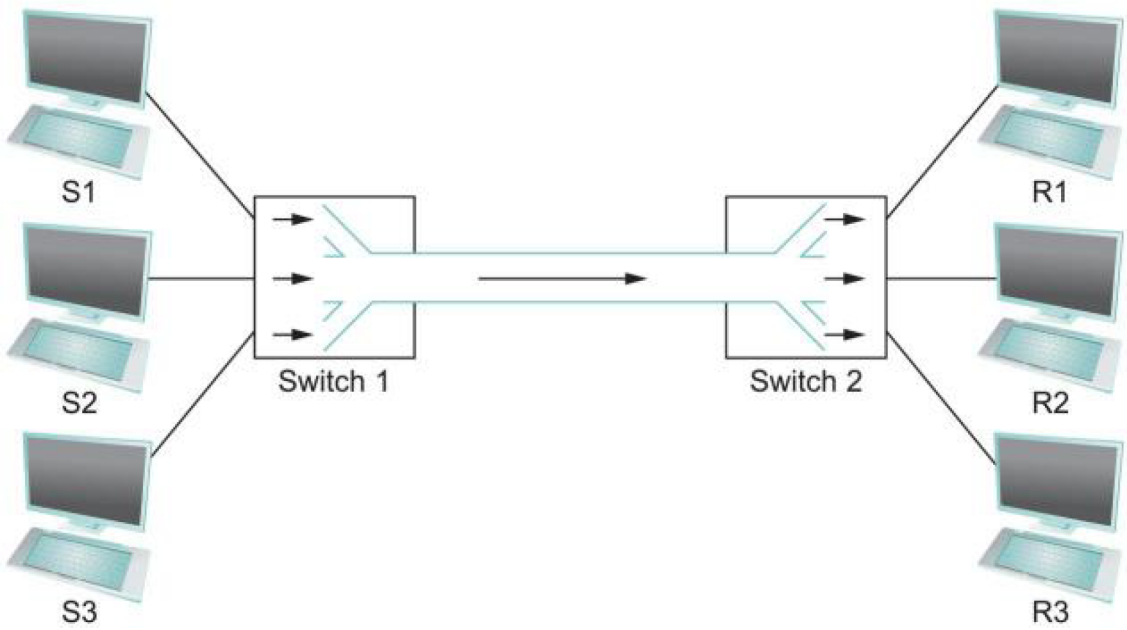


FIGURE 1.5 Multiplexing multiple logical flows over a single physical link.

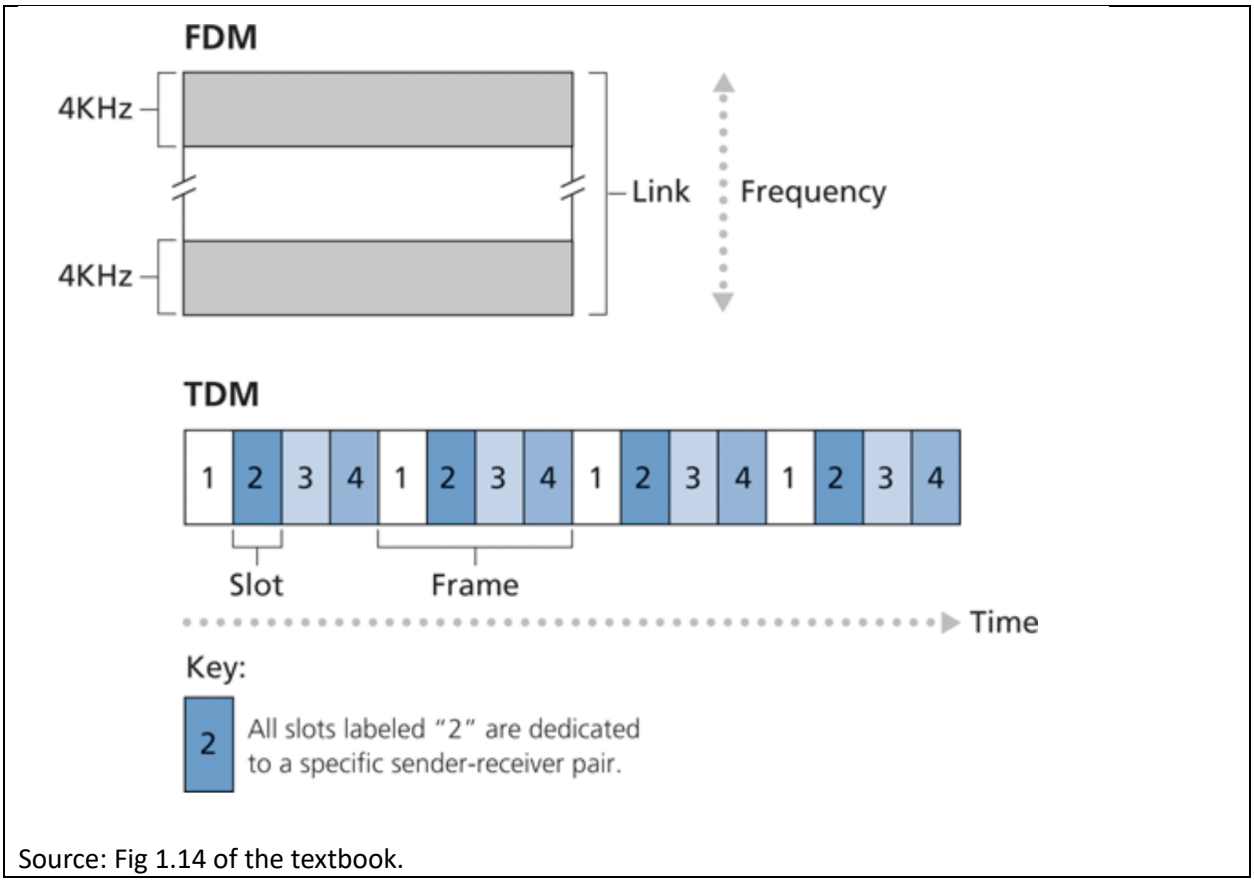
Source: Computer networking: a systems approach by Peterson and Davie

“

There are several different methods for multiplexing multiple flows onto one physical link. One common method is ***synchronous time division multiplexing*** **(STDM)**. The idea of STDM is to divide time into equal-sized quanta and, in a round-robin fashion, give each flow a chance to send its data over the physical link. In other words, during time quantum 1, data from S1 to R1 are transmitted; during time quantum 2, data from S2 to R2 are transmitted; in quantum 3, S3 sends data to R3. At this point, the first flow (S1 to R1) gets to go again, and the process repeats.

Another method is ***frequency-division multiplexing*** **(FDM).** The idea of FDM is to transmit each flow over the physical link at a different frequency, much the same way that the signals for different TV stations are transmitted at a different frequency over the airwaves or on a coaxial cable TV link.

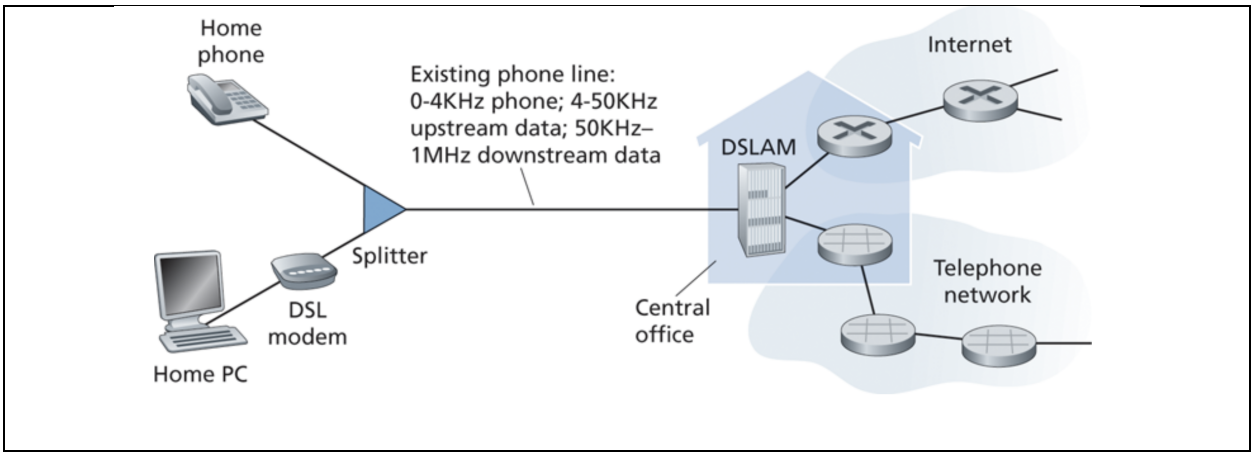
“



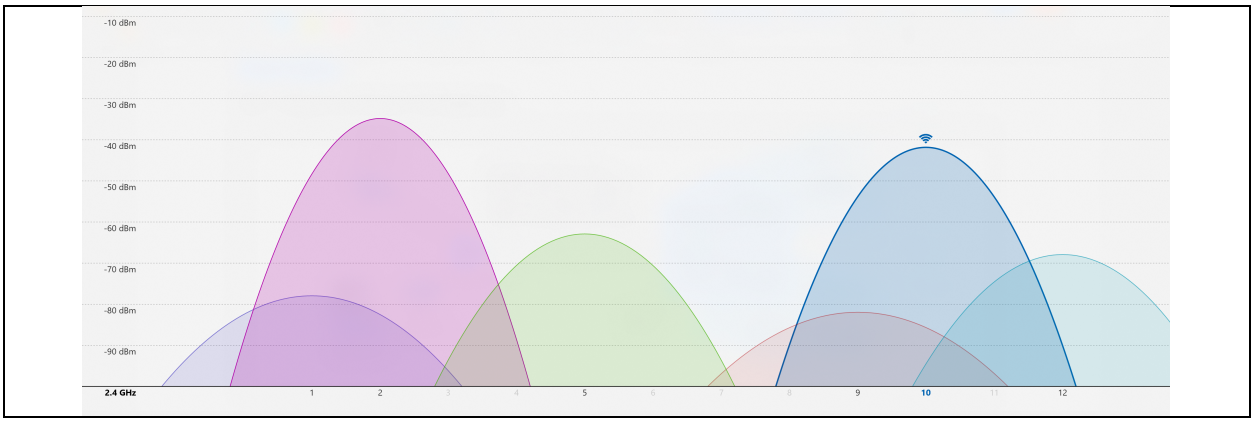
Source: Fig 1.14 of the textbook.

Examples:

DSL: Frequency Division Multiplexing



WiFi: Frequency Division Multiplexing



What is HZ (Hetz)?

In the context of signals, Hertz (Hz) represents the frequency of the signal. Frequency is a key characteristic of a signal and refers to how many cycles of a periodic waveform occur in one second. It is a measure of bandwidth.

Time division multiplexing:

4G LTE (Long-Term Evolution) uses a combination of multiple access techniques, including Time Division Multiplexing (TDM) and Frequency Division Multiplexing (FDM), but it is more specifically based on **Orthogonal Frequency Division Multiple Access (OFDMA)** and **Single-Carrier Frequency Division Multiple Access (SC-FDMA)**.

DOCSIS (Data Over Cable Service Interface Specification) uses Time Division Multiplexing (TDM) as part of its overall strategy for managing data transmission over cable networks, but it primarily utilizes a combination of **TDM** and **Frequency Division Multiplexing (FDM)**.

Shortcomings of FDM and TDM:

(1) If one of the flows (host pairs) does not have any data to send, its share of the physical link—that is, its time quantum or its frequency—remains idle, even if one of the other flows has data to transmit.
(2) Both STDM and FDM are limited to situations in which the maximum number of flows is fixed and known ahead of time. It is not practical to resize the quantum or to add additional quanta in the case of STDM or to add new frequencies in the case of FDM.

Statistical Multiplexing:

(1) The form of multiplexing that addresses these shortcomings, and of which we make most use in the packet switched Internet, is called *statistical multiplexing*.

(2) It is like STDM but data from each flow is transmitted on demand. This avoidance of idling the link gives packet switching its efficiency.
(3) For how long a flow can go on the link is necessary so that everyone could get its turn on the link.

Packet Switching VS Circuit Switching:

Why is packet switching more efficient? Let's look at a simple example. Suppose users share a **1 Mbps** link. Also suppose that each user alternates between periods of activity, when a user generates data at a constant rate of **100 kbps**, and periods of inactivity, when a user generates no data. Suppose further that a user is active only 10 percent of the time (and is idly drinking coffee during the remaining 90 percent of the time).

Thus, the circuit-switched link can support **only 10 (=1Mbps/100kbps) simultaneous users**.

With packet switching, the probability that a specific user is active is 0.1 (that is, 10 percent). If there are 35 users, the probability that there are 11 or more simultaneously active users is approximately 0.0004. (Homework Problem P8 outlines how this probability is obtained.)

When there are 10 or fewer simultaneously active users (which happens with probability 0.9996), the aggregate arrival rate of data is less than or equal to 1 Mbps, the output rate of the link. Thus, when there are 10 or fewer active users, users' packets flow through the link essentially without delay, as is the case with circuit switching. When there are more than 10 simultaneously active users, then the aggregate arrival rate of packets exceeds the output capacity of the link, and the output queue will begin to grow. (It continues to grow until the aggregate input rate falls back below 1 Mbps, at which point the queue will begin to diminish in length.) Because the probability of having more than 10 simultaneously active users is minuscule in this example, packet switching provides essentially the same performance as circuit switching, but does so while allowing for more than three times the number of users.

Internet: Network of networks:

Figure 1.15 **Interconnection of ISPs**

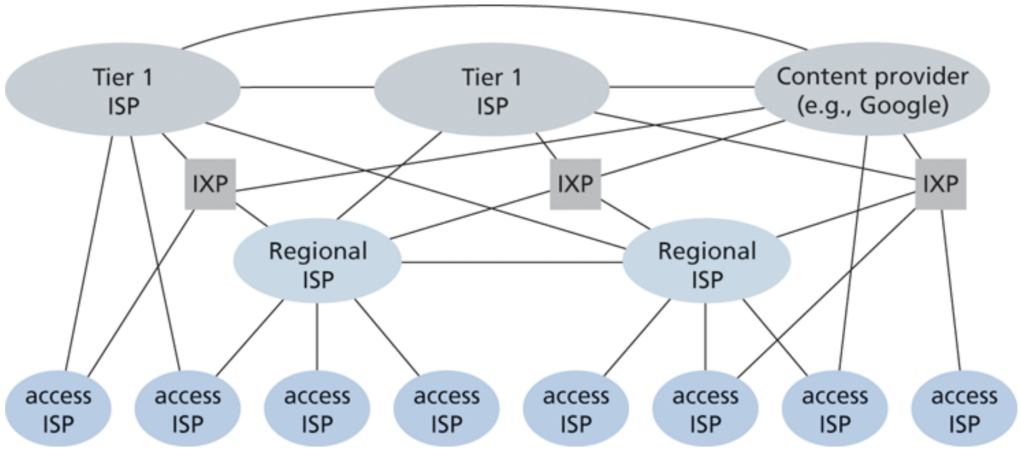


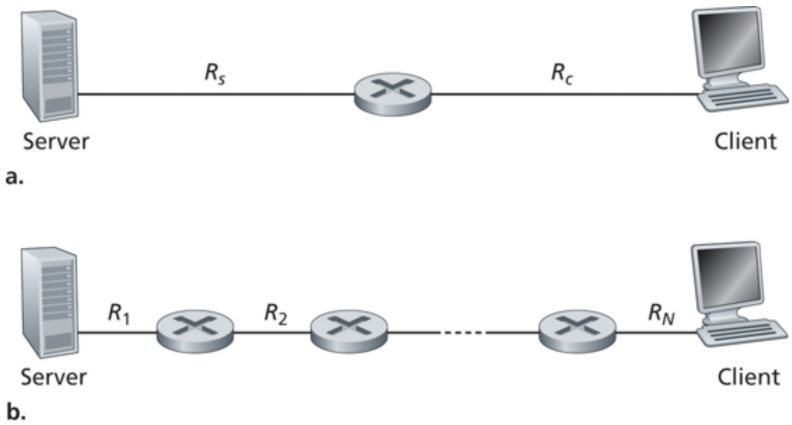
Figure 1.15 Full Alternative Text

Source: Figure 1.15 from the textbook

End-to-End Throughput:

The concept of bottleneck link: The effective throughput depends on the bottleneck link on the path from the source to the destination.

Figure 1.19 **Throughput for a file transfer from server to client**



Source: Fig 1.19 of the textbook

How big is a "mega"?

There are several pitfalls you need to be aware of when working with the common units of networking, such as MB and Mbps. The first is to distinguish carefully between bits and bytes. Throughout this book, we always use a lowercase *b* for bits and a capital *B* for bytes. The second is to be sure you are using the appropriate definition of giga (G), mega (M), and kilo (K). *Mega*, for example, can mean either 2^{20} or 10^6. Similarly, *kilo* can mean either 2^{10} or 10^3, and *giga* can mean either 2^{30} or 10^9. What is worse, in networking, we typically use both definitions. Here is why.

Network bandwidth, which is often specified in terms of Mbps, is typically governed by the speed of the clock that paces the transmission of the bits. A clock that is running at 10 MHz is used to transmit bits at 10 Mbps. Because the *mega* in MHz means 10^6 hertz, Mbps is usually also defined as 10^6 bits per second. (Similarly, Gbps is 10^9 bits per second.) On the other hand, when we talk about a message that we want to transmit, we often give its size in bytes. Because messages are stored in the computer's memory, and memory is typically measured in powers of two, the *K* in kB is usually taken to mean 2^{10}. (Similarly, MB usually means 2^{20} and GB usually means 2^{30}.) When you put the two together, it is not uncommon to talk about sending a 64-kB message over a 100-Mbps channel, which should be interpreted to mean $64 \times 2^{10} \times 8$ bits are being transmitted at a rate of 100×10^6 bits per second. This is the interpretation we use throughout the book, unless explicitly stated otherwise.

The good news is that many times, we are satisfied with a back-of-the-envelope calculation, in which case it is perfectly reasonable to make the approximation that 10^6 is really equal to 2^{20} (making it easy to convert between the two definitions of mega). This approximation introduces only a 5% error. We can even make the approximation in some cases that a byte has 10 bits, a 20% error but good enough for order-of-magnitude estimates.

While we are making quick-and-dirty calculations, 100 ms is a reasonable number to use for a cross-country round-trip time—at least when the country in question is the United States—and 1 ms is a good approximation of an RTT across a local area network. In the case of the former, we increase the 48-ms round-trip time implied by the speed of light over a fiber to 100 ms, because there are, as we have said, other sources of delay, such as the processing time in the switches inside the network. You can also be sure that the path taken by the fiber between two points will not be a straight line.

Source credit: Computer Networks by Peterson and Davie

Security:

We define security as:

(1) Need for confidentiality
(2) Need for integrity
(3) Availability of the system

Bad actors can target all three above by sniffing network data, and Distributed Denial of Service Attacks (DDoS). As David Clark tells in the following paper, security was never a top-level goal of the initial Internet design. Security has been retrofitted over time.

The design philosophy of the DARPA internet protocols by David Clark

Source: Wikipedia, https://en.wikipedia.org/wiki/David_D._Clark