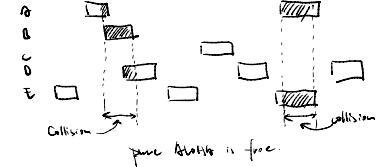


4.2 Multiple access protocols

4.2.1 ALOHA

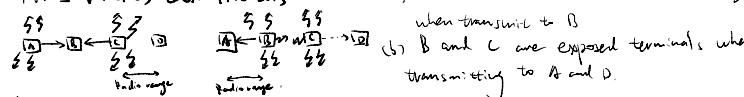
Pure ALOHA: send the frame when ready, resent if the frame is destroyed by user



Whenever two frames try to occupy the channel at the same time, there will be a collision (as seen in the figure) and both will be erased. If the first bit of a new frame overlaps with just the last bit of a frame that has almost finished,

both frames will be totally destroyed (i.e. have incorrect checksum) and both will have to be retransmitted later. The checksum does not (and should not) distinguish between a total loss and a near miss. Bad idea!

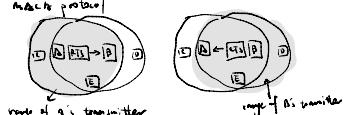
4.2.5 Wireless LAN Protocols



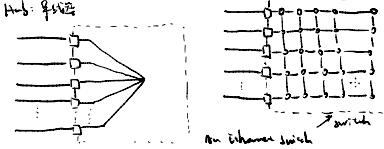
(a) A sends and when C immediately senses the medium, it will not hear A because C is out of its range. Thus C will falsely conclude that it can transmit to B if C does start transmitting, it will interfere at B, wiping out the frame from A.

(b) If C senses the medium, it will hear a transmission and falsely conclude that it may not send to B (shown as a dashed line). In fact, such a transmission would cause bad reception only in the zone between B and C, where neither of the intended receivers is located.

Media protocol

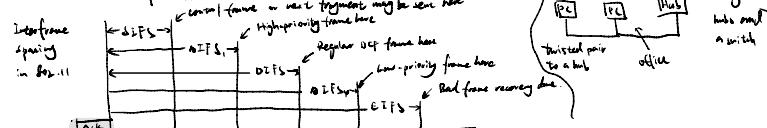


4.3.4 Switched Ethernet

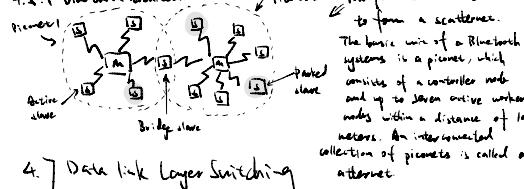


4.4.3 The 802.11 MAC sublayer protocol

Time slotting discipline: limits by extending CSMA/CA with carefully defined intervals between frames.

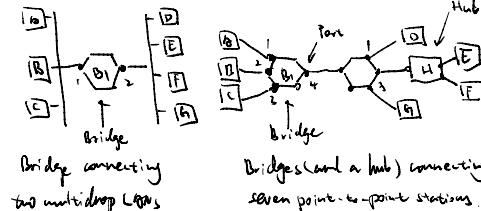


4.5.1 Bluetooth architecture



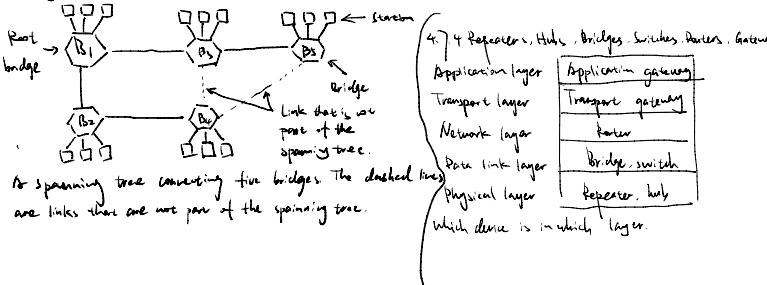
4.7 Data Link Layer Switching

4.7.1 Learning Bridge



4.7.3 Spanning-Tree Bridges

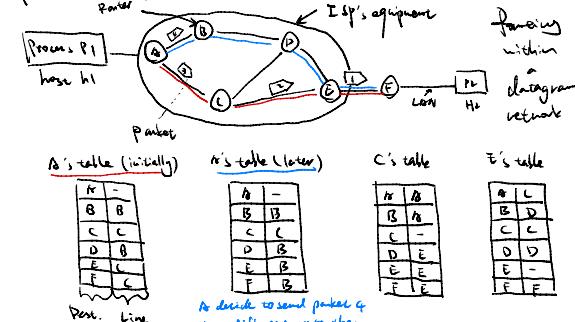
For the bridges to communicate with each other and overlay the actual topology with a spanning tree that reaches every bridge:



A spanning tree connecting five bridges. The dashed lines are links that are not part of the spanning tree.

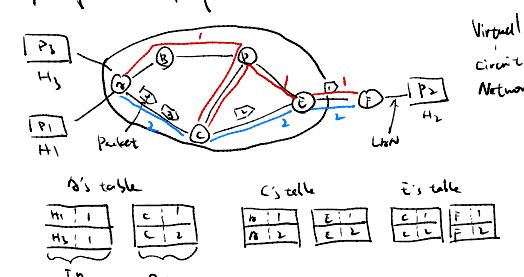
Chp 5 Network layer

5.1.3 Implementation of connectionless service

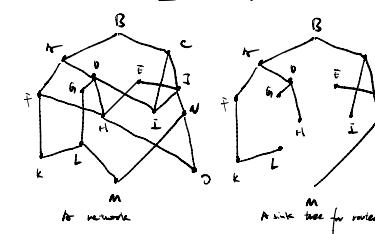


The algorithm that manages the tables and makes the routing decisions is called the routing algorithm.

5.1.4 Implementation of Connection-Oriented Service



5.2.1 The Optimality Principle



One can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the optimality principle.

e.g. If a route from node A to node B is

optimal, then every sub-path of that route must also be optimal. That is, if the path A to B is the best path from A to C, then the path from B to C must also be the best path from B to C. As a direct consequence of the optimality, the optimality principle, the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree

e.g. (b) represents a sink tree for node B, which means that for every node in the network, the path from B to that node in the sink tree is as short as possible. It's a subset of the network that highlights the optimal paths from the sink (or root) to all other nodes.

Distance Vector Routing: Although it converges to the correct answer, it may do so slowly. Link State Routing: Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them, and more bandwidth is needed to send status reports about them.

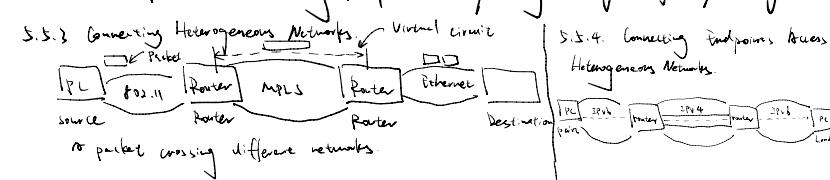
Hierarchical Routing within a network: There is a penalty to be paid increased path length. For example, the best route from 1c to 5c is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.

Demand-based Routing: Each router must have knowledge of some spanning tree.

5.2.8 Multicast Routing: sending a message to such a group is called multicasting; and the routing algorithm used is called multicast routing.

In anycast, a packet is delivered to the nearest member of a group. Such shortest paths are called anycast routing.

5.4.3 Packet Scheduling FIFO, Fair queuing, weighted fair queuing.



5.5.3 Connecting Heterogeneous Networks: Virtual circuit switching. 5.5.4. Connecting Endpoints Across Heterogeneous Networks: IPsec, SRV6, SDN, 2025, 2026, 2027.

No packet crossing different networks.

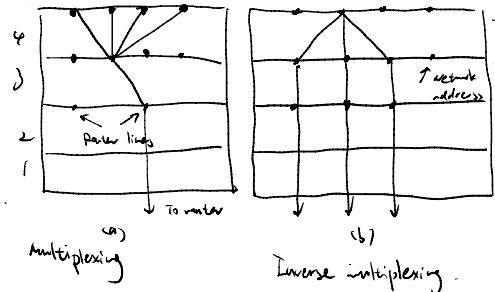
8.2.1 Transport Service Priorities

When a frame arrives, the data link layer processes the frame header and, if the destination address matches for local delivery, passes the contents of the frame payload field up to the network layer. The network layer similarly processes the packet header and then passes the contents of the payload up to the transport layer.

$\phi \geq 1$, addressing

To APs, NSPs, and transport connections
the transport addresses to which
processes can listen for connection
requests are called ports.

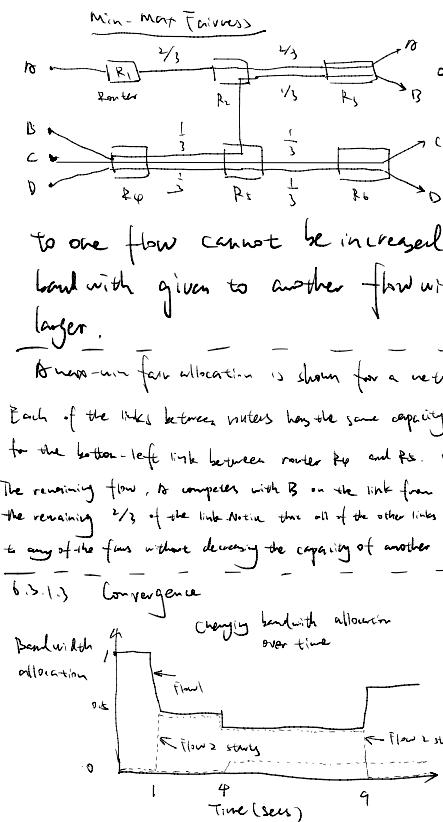
b. > 5 multiplexing



network paths that it can use. If a user needs more bandwidth or more reliability than one of the network paths can provide, a way out is to have a connection that distributes the traffic among multiple network paths on a round-robin basis. This is called inverse multiplexing.

6.3 Congestion Control: The only effective way to control congestion is for the transport protocols to send packets into the network more slowly.

6.3.1.2 Desirable Bandwidth Allocation



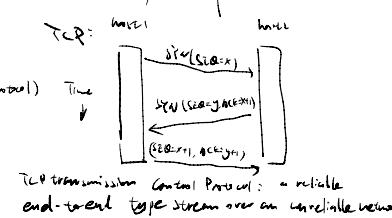
dividing the available bandwidth to each have 65% of the bandwidth. For 9 seconds the second flow leaves, and the third flow remains unchanged. The first flow quickly captures 85% of the bandwidth.

b. 3. 2. 1 Regulating the Sending Rate (AIMS)

b-4.1 Introduction to UDP

The connectionless protocol is UDP (User Datagram Protocol).

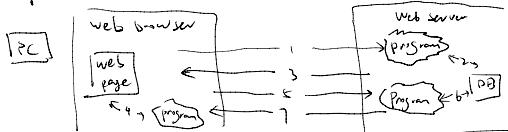
Source part	Destination part
UDP header	UDP checksum



7.3.4 HTTP and HTTPS | HTTP vs HTTPS: The protocol that is used to transport all the information between Web servers and clients is HTTP. Both protocols HTTP and HTTPS (Secure Hypertext Transfer Protocol) essentially retrieve objects in the same way. The HTTP standard to retrieve Web objects is evolving essentially independently from its secure counterpart, which effectively uses the HTTP protocol over a secure transport protocol called TLS (Transport Layer Security).

Static vs dynamic content or objects

Static objects: files sitting on a server, presenting themselves in the same way each time they are fetched and viewed, e.g. logos, style sheets, header and footer. Generally amenable to caching (dynamic). This is content that can change based on user interaction (location, permission, etc).



The requests and responses happen in the background; the user may not even be aware of them because the page URL and title typically do not change. By including client-side programs the page can present a more responsive interface than with server-side programs alone.

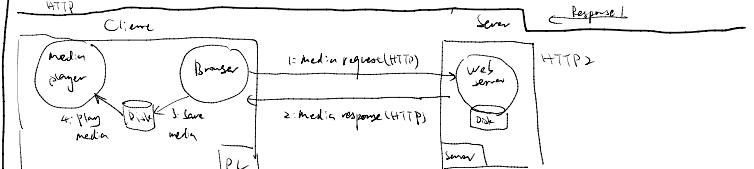
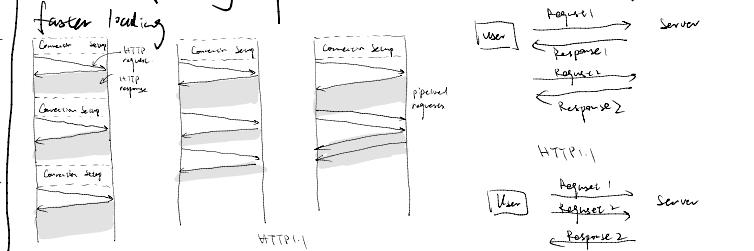
HTTP.1 vs HTTP.1: HTTP.1: supports only one request per connection and does not support persistence connections leading to more bandwidth waste and less efficient cache management.

HTTP/1.1, which supports persistent connections. With them, it is possible to establish a TCP connection, send a request and get a response, and then send additional requests and get additional responses. This strategy is also called connection reuse.

HTTP1.1 vs HTTP2:

HTTP 1.1: It is characterized by loading resources one after the other, which can lead to one resource blocking another if there are issues with loading.

HTTP2: It allows a single TCP connection to send multiple streams of data at once, Server push, which sends content to clients before they request it, and HTTP/2 header compression for faster loading



- step 1: it sends an HTTP request for the movie to the Web server to which the movie is linked.
- step 2: The server fetches the movie (which is just a file in MP4 or some other format) and sends it back to browser

step 3: The browser then saves the entire movie to a scratch file on disk in WMP9.
step 4: Finally, in step 4 the media player starts reading the file and plays the movie.

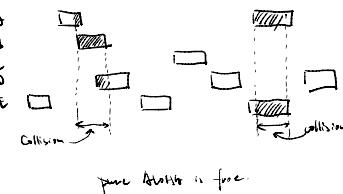
② Two hosts Because there are only 14

January falls and whether the year is a leap year. Thus, a Java program could easily contain all six code for and a small database of which year

with caterpillar

Five Atoms: send the frame when ready, resend if the frame is destroyed

User



pure Atoms is free.

Whenever two frames try to occupy the channel at the same time, there will be a collision (as seen in the figure) and both will be erased. If the first bit of a new frame overlaps with just the last bit of a frame that has almost finished, both frames will be totally destroyed (i.e. have no error detection) and both will have to be retransmitted later. The checksum does not (and should not) distinguish between a total loss and a near miss. Bad idea.

(a) A and C are hidden terminals to B, when transmitting to B.

If A sends and then C immediately senses the medium, it will not hear A because A is out of its range. Thus C will falsely conclude that it can transmit to B. If C does start transmitting, it will interfere at B, wiping out the frame from A. (We assume here that no CSMA-type scheme is used to provide multiple channels, so collisions garble the signal and destroy both frames.) We want a MAC protocol that will prevent this kind of collision from happening because it wastes bandwidth. The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the hidden terminal problem.

(b) B and C are exposed terminals when transmitting to A and D.

If C senses the medium, it will hear a transmission and falsely conclude that it may not send to D (shown as a dashed line). In fact, such a transmission would cause bad reception only in the zone between B and C, where neither of the intended receivers is located. We want a MAC protocol that prevents this kind of deferral from happening because it waste bandwidth. The problem is called the exposed terminal problem.

- One can make a general statement about optimal routes without regard to network topology or traffic.

This statement is known as the optimality principle.

e.g. If a route from node A to node B is optimal, then every sub-path of that route must also be optimal. That is, if the path A to B is the best path from A to C, then the path from B to C must also be the best path from B to C.

- As a direct consequence of the optimality, the optimality principle, the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree.

- Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist.

e.g. (b) represents a sink tree for route B, which means that for every node in the network, the path from B to that node in the sink tree is as short as possible. It's a subset of the network that highlights the optimal paths from the sink (or root) to all other nodes.

Distance Vector routing: Although it converges to the correct answer, it may do so slowly.

Link State Routing: Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them, and more bandwidth is needed to send status reports about them.

Hierarchical routing within a network: There is a penalty to be paid: increased path length. For example, the best route from 1c to 5c is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.

Broadcast routing: Each router must have knowledge of some spanning tree.

(a) If only one network address is available on a host, all transport connections on that machine have to use it. When a segment comes in, some way is needed to tell which process to give it to. This situation called multiplexing, is shown in Fig(a). In this figure, four distinct transport connections all use the same network connection (e.g. IP address) to the remote host.

(b) A host has multiple network paths that it can use. If a user needs more bandwidth or more reliability than one of the network paths can provide, a way out is to have a connection that distributes the traffic among multiple network paths on a round-robin basis. This is called inverse multiplexing.

- The form of fairness that is often desired for network usage is max-min fairness. An allocation is max-min fair if the bandwidth given to one flow cannot be increased without decreasing the bandwidth given to another flow with an allocation that is no larger.
- Each of the links between routers has the same capacity, taken to be 1 unit, though in the general case the links will have different capacities.
- A max-min fair allocation is shown for a network with four flows, A, B, C, and D. Each of the links between routers has the same capacity, taken to be 1 unit, though in the general case the link will have different capacities, taken to be 1 unit, though in the general case the links will have different capacities. Three flows compete for the bottom-left link between router R₂ and R₃. Each of these flows therefore gets $\frac{1}{3}$ of the link. The remaining flow, A, competes with B on the link from R₂ to R₃. Since B has an allocation of $\frac{1}{3}$, it gets the remaining $\frac{2}{3}$ of the link. Notice that all of the other links have pure capacity. However, this capacity cannot be given to any other flow without decreasing the capacity of another flow. If more of the bandwidth on the link between R₂ and R₃ is given to flow B, there will be less for flow A. This is reasonable as flow B already has more bandwidth. However, the capacity of flow C or D (or both) must be decreased to give more bandwidth to B, and these flows will have less bandwidth than B. Thus, the allocation is max-min fair.

HTTP vs HTTPS: The protocol that is used to transport all the information between the servers and clients

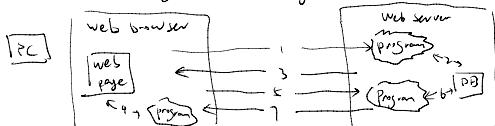
is HTTP. Both protocols HTTP and HTTPS (Secure Hypertext Transfer Protocol) essentially retrieve objects in the same way.

The HTTP standard to retrieve Web objects is evolving essentially independently from its secure counterpart, which effectively uses the HTTP protocol over a secure transport protocol called TLS (Transport Layer Security).

Static vs dynamic content or object:

static objects: files sitting on a server, presenting themselves in the same way each time they are fetched and viewed.
e.g. logo, style sheets, header and footer.
Generally amenable to caching

dynamic: This is content that can change based on user interaction, location, permission, etc.



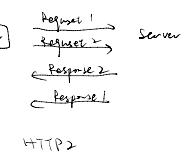
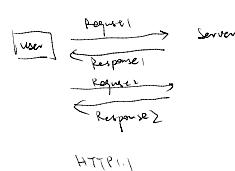
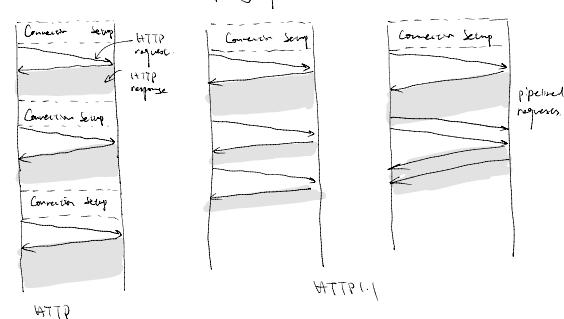
The requests and responses happen in the background; the user may not even be aware of them because the page URL and title typically do not change. By including client-side programs, the page can present a more responsive interface than with server-side programs alone.

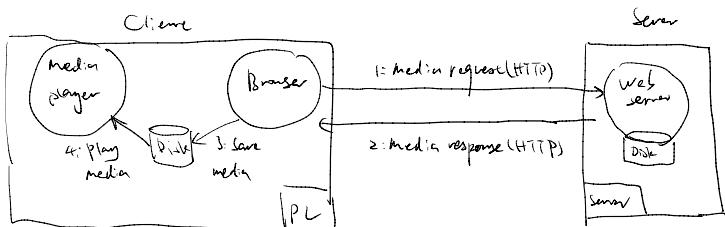
HTTP1 vs HTTP1.1

HTTP1: Supports only one request per connection and does not support persistent connections, leading to more bandwidth waste and less efficient cache management.
HTTP1.1: which supports persistent connections. With them, it is possible to establish a TCP connection, send a request and get a response, and then send additional requests and get additional responses. This strategy is also called connection reuse.

HTTP1.1 vs HTTP2:

HTTP1.1: It is characterized by loading resources one after the other, which can lead to one resource blocking another if there are issues with loading.
HTTP2: It allows a single TCP connection to send multiple streams of data at once, sever path, which sends content in chunks before they request it, and HTTP/2 header compression for faster loading.





- Step 1: it sends an HTTP request for the movie to the Web server to which the movie is linked.
- Step 2: The server fetches the movie (which is just a file in MP4 or some other format) and sends it back to browser
- Step 3: The browser then saves the entire movie to a scratch file on disk in steps.
- Step 4: Finally, in step 4 the media player starts reading the file and play the movie.

- ② JavaScript. Because there are only 14 annual calendar, depending on the day of the week on which 1 January falls and whether the year is a leap year. Thus, a Javascript program could easily contain all 14 calendars and a small database of which year gets with calendar.
- ③ PHP. This requires a large database. It must be done on the server by using PHP.
- ④ Both works, but JavaScript is faster.