Seminar 6. Wireless and Mobile Networks/Multimedia Networking

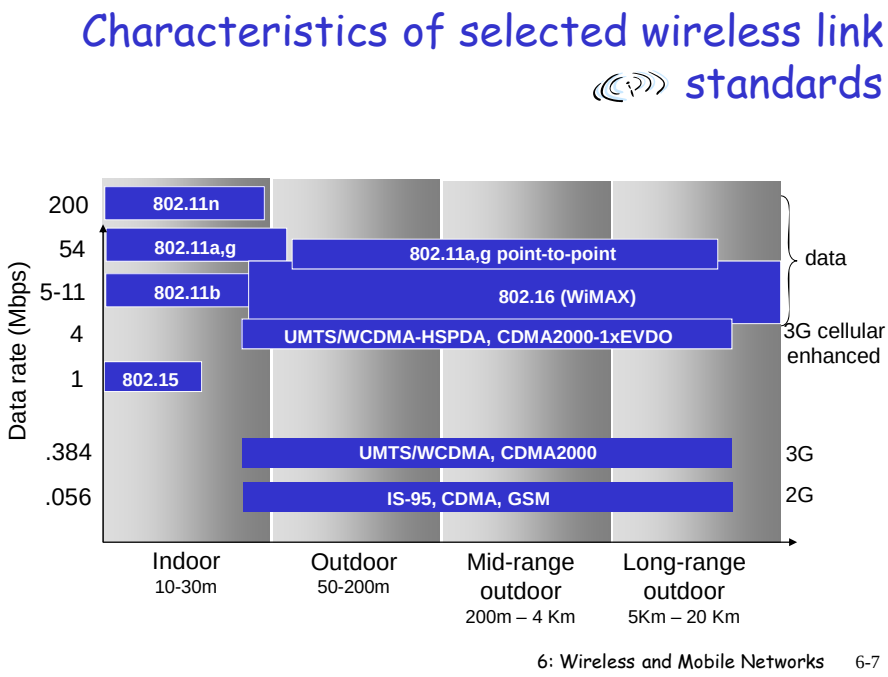
Wireless and mobile networks

The number of wireless (mobile) subscriber now exceeds the number of wired phone subscribers. Computer networks have now morphed into laptops, PDAs, palmtops, an Internet-enabled phones offering untethered Internet access. This week's lecture discusses two important (but different) challenges.

Wireless

Communications taking place over the wireless link now become the norm. There are several elements of a wireless network. Some of these include wireless hosts such as laptops, PDAs, IP phones, and other devices that run applications and maybe stationary (non-mobile) or mobile. It is important to remember that wireless does not always mean the mobility. In a wireless network, the base station is typically connected to a wired network in some fashion. The relay is responsible for sending packets between wired networks and wireless host(s) and asked area. A good example of this would include cell towers and 802.11 access points.

A wireless link is typically used to connect mobile devices to the base station, and it can also be used in the backbone link. This can involve multiple access protocol coordinating link access, as well as various data rates and transmission distances. The characteristics of selected wireless LAN standards are seen in the graphic below.



Also, when a wireless network operates in infrastructure mode, the base station connects mobile devices into a wired network of some sort. This results in a handoff where mobile devices change base stations based on signal strength. It is important to remember that the base station currently in use is still providing the connection into the wireless network.

The final mode possible in wireless networks is called "ad hoc mode." In this scenario, there are no base stations but the actual nodes can only transmit to other nodes within link coverage. These nodes organized themselves into a network and routing takes place internally within the nodes themselves.
Mobile
Mobile IP is outlined in great detail in RFC 3344. This area has many features we have seen earlier in the text. Some of these include home agents, foreign agents, foreign-agent registration, and encapsulation. There are three components to this standard: 1) in direct routing of datagrams; 2) agent discovery; and 3) registration went home agents.
The home network is a network of cellular providers that offer subscriptions to users. Examples include Verizon Wireless and Sprint PCS. The home location register (HLR) is a database and home networks containing permanent cell phone numbers, profile information such as services, preferences, and billing, as well as information about current location (in the event the mobile device is currently located in another network.)
The visited network is the network in which the mobile device currently resides. In turn, the visitor location register (VLR) is a database with an entry for each user currently in a network. It is important to remember that this mobile device can actually still be in the home network.
Impact on Higher Layer Protocols
In wireless and mobility, the best effort service model remains unchanged. In fact, DTP and UDP can (and do) run over wireless and mobile devices. Therefore, the impact on higher layer protocols should be minimal. In relation to performance, however, packet loss/delay is sometimes evident due to bit-errors (discarded packets, delays for link-layer transmissions), and handoff. TCP interprets loss as congestion and will often decrease the congestion window unnecessarily. Other factors affecting this can include delay impairments for real-time traffic as well as limited bandwidth of the wireless links.

Present Internet technology uses the store-and-forward method to pass data between communicating applications. The store-and-forward technique results in undetermined delays in links, in addition to variable arrival packet rates at destinations. Although this transfer behavior is acceptable in data-oriented applications such as file transfers, it is not acceptable for phone conversation or TV broadcasting.
Currently, various techniques and architectures are being introduced to make the best of the available Internet technology. For example, new functions are being added to routers to manage traffic belonging to different applications. In the following we are going to discuss difficulties facing real-time applications and their proposed solutions.

Example Multimedia Applications:

1. Streaming S*tored* Audio and Video. In this type, multimedia files are stored in servers that clients download and play on their computers. Rather than waiting for the whole file to get loaded (and probably overfill the local hard disk), this type of application starts playing received parts while loading is still in progress. The process of loading parts while others are being played is called streaming. Streaming is mostly successful thanks to data buffering before playing, however problems might arise if the network becomes excessively slow.
2. Streaming of *Live* Audio and Video. Unlike the first application, this type broadcasts live data such as TV pictures. In other words, data is sent in real-time as they are produced. However, data packets arrive at receivers at variable speeds (depending on variable network load conditions).
3. Real-time Interactive Audio and Video: This form of application is becoming increasingly available on the Internet; many people are using Internet phones to communicate. Nevertheless, Internet phones are still not as good as regular phones in terms of clarity and availability (you still need to have light Internet traffic to make a call)

What is still missing?

Network layer still provides best-effort service, which does not guarantee enough bandwidth

during multimedia data interaction. In addition, there is no guaranteed *constant* packet rate from sender to receiver, which results in packet jitter. (How does packet jitter present itself in played multimedia data? How is it felt in phone conversations?).

Data compression as a partial remedy to the limited bandwidth problem:

Speech requires a bandwidth of 64 Kbps, while stereo music requires 1.411 Mbps.
Video data, on the other hand, requires a bandwidth of 1.5 Mbps. Today's modems can provide enough speed to deliver speech data but not stereo music or video pictures. Thanks to data compression, bandwidth requirements are significantly reduced. Data is compressed before being transmitted; subsequently, clients decompress data before playing them. Popular compression techniques for audio data are GSM, G.729, G.723.3, MP3 and for video data MPEG.

Accessing audio/video from a Web/Streaming servers

There are different models to establish multimedia client/server applications. Models that use web browsers are restricted by the HTTP and TCP limitations. For example, TCP implements data congestion techniques that change segment delivery rate to alleviate link congestion. However as we noted above, audio/video data is badly affected by variations in data rates. Therefore, Web servers are obviously not the most suitable way to function as audio/video servers.
Alternatively, HTTP can be used in an initial phase to request multimedia data from servers and, at the same time, to prepare a local player to receive data with certain specifications. Servers using UDP protocol (with constant transmission speed) allows direct communication with the multimedia player on the client's side.
A protocol called Real-time Streaming Protocol (RTSP) is used to send *out-of-band* signals that allow multimedia players to control the transmission of a stored media stream, such as fast forwarding, pausing, etc.

The Internet phone example:

Before going into the Internet phone example, let us recall how a regular phone works. Regular phones maintain a direct circuit between the talking parties throughout the phone conversation. During the setup time a circuit is established, and stays open until it is terminated by ending the phone session. Thus, resources are allocated to the conversation session even if the talking parties pause for lengthy periods of time. Internet, on the other hand, uses the store and forward principle to pass data between parties. Resources from sender to receiver are always shared by many traffic flows at the same time. Accordingly, the Internet strives to allocate resources to other users while talking parties pause during conversation.
Today's Internet was not designed with Internet phones, or other interactive multimedia applications, in mind. A message from one application to another is segmented by the application layer, and then passed on to the transport layer (TCP or UDP protocols), which hands it over (after some processing) to the IP of the network layer, and so on. The network layer's IP does not promise any time limit to the delivery of the data segment to the destination, nor does it guarantee any constant speed for packet delivery. Meanwhile, UDP, the preferred transport layer protocol for this application (why?) does not promise safe delivery of packets. A brief discussion of the difficulties of using Internet phones:

1) Packet loss: This problem can be tolerated as long as the loss is less than 20%. The following techniques help to minimize the effect of lost frames:
a) Forward error correction: every stream's packet is piggybacked with a lower quality copy of previous packet. Lower quality packets are smaller in size than the original packets. If a packet is lost, then the received lower quality (attached to the next received packet) is played instead. See Figure 7.7 (3rd edition) or Figure 7.6 (4th edition and 5th edition) in the textbook. (What would happen if a number of consecutive packets were lost?)
b) Interleaving: Every packet is cut down into small pieces. Pieces from different packets are grouped together to form an interleaved packet (Figure 7.8 (3rd edition) or Figure 7.7

(4th edition and 5th edition) in the textbook). If any of these interleaved packets is lost, then a piece of each participating packet is lost. Losing a small part of each of several packets is better than loosing a whole packet. (Do you agree?). This method needs the use of a buffer for collecting the packets - which causes an additional delay.
c) Receiver-Based Repair of Damaged Audio Streams: This easy technique replaces a lost packet with the packet received next! No surprise. It has been observed that speech packets are similar within a short period of time (what is the duration of speech packet, btw?). An alternative to this method is to derive lost packets from previous and next packets.

2) End-to-end delay: This problem is felt when the delay is over 400 milliseconds. The effect becomes bothersome when you say a sentence that is not heard by the other party before about half a second (this happens even in regular phones at holiday times). Unfortunately, this problem is still to be solved. At the present time, you still need to hunt for a reasonably low traffic time to make an Internet phone conversation.

3) Delay jitter. Speech packets are generated by the sender every 20 milliseconds. However, these packets arrive at the receiver at a rate greater or smaller than the rate at which they were produced. Therefore, if packets are played as they are received then the quality can be unintelligible at the receiver. Following are techniques to remove jitter at the receiver:
a) Fixed Playout Delay: This is a natural and easy solution to the jitter problem and is done by collecting a number of packets before playing them. Once a number of packets are available, they can be spaced correctly before being played (Figure 7.6 (3rd edition) or figure 7.5 (4th edition and 5th edition) in the textbook). However, waiting for a number of packets to accumulate introduces a certain delay (which is ok as long as it is less than 400 milliseconds). It is also possible to reduce the playout delay by dropping packets that failed to arrive within a preset delay. The less the playout delay, the more the lost frames (we can afford to lose up to 20% of frames, however).
b) Adaptive Playout Delay: this is a more developed technique to remove jitter in received packets. This technique relies on compressing the pause between talk spurts, instead of a fixed playout delay. A new value for delay is estimated each time by relying on the past estimated network average delay and the actual last time delay (see mathematical derivation of the smoothed average delay in the text)

Real-Time Protocol (RTP)

We have already studied difficulties, as well as solutions, facing real-time (multimedia) streams. RTP is used to help overcome these difficulties through communicating necessary information between senders and receivers.

This protocol adds header fields such as sequence numbers and time stamps to audio/video chunks. Obviously these fields help the receiving application to reconstruct the original stream. Although this protocol is incorporated inside applications and uses UDP sockets, it is still considered a transport layer protocol (see Figure 7.10 (3rd edition) or Figure 7.11 (4th edition and 5th edition) in
the textbook). RTP is a popular protocol because it is being used by many important applications. RTP includes the following fields in its header:

1. Payload type. This field indicates the type of audio encoding (PCM, for example). The sender might choose to change the encoding type on the run, to help alleviate network congestion for example.
2. Sequence number field. Receivers use this field to detect packet loss.
3. Timestamp field. Receivers use this field to remove packet jitter.
4. Synchronization source identifier. This is a random number chosen and assigned by the source to identify stream source (why random?).

RTP Control Protocol (RTCP)

This protocol is used in conjunction with RTP to help, among other things, in synchronizing different media streams (voice and picture, for example). In addition, every receiver and sender transmits periodic packets of this protocol to report quality of received/sent packet stream. RTCP packets include report about fractions of lost packets, last sequence number received/sent, the interarrival jitter, timestamp of the most recently generated RTP packet, etc.
Since RTCP packets are generated by every receiver to all other receivers, the number of RTCP packets becomes extremely large as the number of receivers increases. To deal with this problem, the number of RTCP packets per time period is reduced according to the increased number of receivers. For example, a small share is given to the sender's RTCP, while the rest is distributed equally among receivers (25% for the sender and 75% shared by all other receivers, for example).

Beyond Best-Effort

IP provides best effort to deliver data safely and on time from source to destination. However, real-time data still suffers from the problems we discussed earlier. The following architectural principles can be added to the network to provide a better service (still not perfect) for multimedia applications:

(1) Packet *marking* allows routers to distinguish among packets belonging to different classes of traffic. Thus, routers can be instructed to give priority to realtime streams.

(2) Packet *classification* allows a router to distinguish among packets belonging to different classes of traffic. This classification adds flexibility to router functions, so that important data other than multimedia can be given priority. Note that classification is more general than marking packets.

(3) It is desirable to provide a degree of isolation among traffic flows, so that one flow is not adversely affected by another misbehaving flow. This means that every flow gets a fixed share of link resources (bandwidth and buffering).

(4) While providing isolation among flows, it is desirable to use resources as efficiently as possible. Similar to the case of dividing a physical link among different flows using FDM or TDM, parts of resources are wasted if the allocated flow does not use them.

(5) A call admission process is needed in which flows declare their QoS requirements and are either admitted to the network or blocked.

Scheduling and Policing Mechanisms

Scheduling and policing mechanisms are used to implement the *architectural* principles listed above.

Scheduling Mechanisms

Routers receive packets from different applications. These packets are queued in router buffers then passed on to proper links. Currently, routers do not differentiate between these packets (no packet marking yet) and serve them on a First Come First Serve (**FCFS)** basis.

Multimedia packets should be given priority to reach destinations within an acceptable minimum end-to-end delay. Priority scheduling passes high-priority packets first, if all are gone, then lower

priority packets are allowed to pass. See Figure 7.26 (3rd edition) or Figure 7.24 (4th edition and 5th edition) in the textbook. Packet 2 arrived into router buffer earlier than packet 3. However, 3 is a higher priority packet and thus it is passed first. Packet 4, however, had to wait for the lower priority packet 2 because packet transmission cannot be interrupted once started. (Could starvation occur for lower priority packets?)

Round robin scheduling provides an equal share to several flows (streams). Unlike fixing each flow with an equal share of the link, Round Robin allows one flow to use the link to full capacity if other flow packets are not present. In Figure 7.27 (3rd edition) or Figure 7.25 (4th edition and 5th edition) in the textbook, packet 2 has to wait for packet 3, because packets 1 and 2 belong to the same stream. In other words, two flows alternate while using the link. On the other hand, packet 4 could pass after packet 2 because there are no other packets belonging to the other flows present.

Weighted fair queuing (WFQ) discipline gives different weights to different flows. A flow is guaranteed a *(R. w_i/©w_j)* of the link transmission rate, *R*. Where w_i is the weight assigned to flow *i.* In other words, WFQ is a multi-priority mechanism.

Policing: The Leaky Bucket

Other than controlling the transmission rate of different flows, policing provides a mechanism to control other aspects of transmission behavior such as: *peak rate, and burst size.* Peak rate is the maximum rate achieved during a short period of time, while burst size is the maximum number of packets transmitted in a short period of time. The leaky bucket mechanism is an abstraction used to explain the controlling mechanism for transmission behavior. Figure 7.29 (3rd edition) or Figure 7.27 (4th edition and 5th edition) in the textbook, explains this mechanism. Tokens arrive at this bucket at a rate of *r* tokens/sec*.* If the bucket is full (*b* tokens), no more tokens are accepted. Now, every packet takes a token when departing the packet queue (thus creating room for a token to enter the bucket); packets are not allowed to leave the queue as long as the bucket is empty. Convince yourself that this mechanism restricts the flow rate to *r* ***packets****/sec.* Also, restricts the burst size to *b* ***packets*** over a short period of time.

WFQ coupled with the leaky bucket policy, Figure 7.30 (3rd edition) or Figure 7.28 (4th edition and 5th edition) in the textbook, provides a provable maximum delay in queue $d_{max} = (b_i)/ (R. w_i/©w_j)$. (See text for proof).

Integrated Services

To guarantee QoS to a session, necessary resources should be allocated from source to destination including all routers' resources on the way. Requests for resource reservations follow route establishments (done by network layer), and are passed from router to router. Every router, upon receiving a request, determines whether it has enough resources (by looking at commitments to other flows). Accordingly a router either forwards the request to the next-on-theroute router after adjusting its scheduling mechanism, or sends an apology in the form of *denial of admission (*analogous to busy signal on telephone lines)*.*
Requested QoS is characterized with the pair (*r,b*) of leaky bucket parameters that we studied earlier (how many quality parameters does this pair imply?)

RSVP

RSVP is a *signaling* protocol used to reserve bandwidth from source to destination. Interestingly, the receivers not the senders initiate this RSVP. Multicast trees are built to establish links between senders and receivers. Because receivers have different abilities to receive data at certain rates, receivers send variable requests for the same stream.

Differentiated Services

The differentiated services architecture (the word *architecture* implies that details are left to future implementers to realize) is still in the early stages of its development and is rapidly evolving.

Integrated services have serious drawbacks, namely lack of scalability and flexibility. As far as scalability is concerned, Intserv (integrated Services) results in a large number of reservation requests made by receivers (as the number of receivers become increasingly large). In the same token, reservations are made per flow from source to destination. Moreover, Intserv serves a limited number of flow classes. As a result, differentiated services (diffserv) are being developed as an innovative mechanism to provide scalable and flexible traffic flow guarantees to requesters.

In Diffserv, most of the calculations are done on the network edges, as opposed to doing them in routers. Packets of each flow are handed passes with a certain category. Clearly, packets of different flows can belong to the same category. The pass category determines the quality of service offered to the holder in intermediate routers. For example class A holders might have twice the bandwidth of class B holders. Unlike Intserv, routers do not need to keep state information of individual flows; they simply treat packets according to their categorization. Details on how to treat each packet are left to routers.

Readings

For the 3rd edition of the textbook
Read chapter 7, and

Skip pages 582-584
Skip pages 608-610
Skip pages 634-636

For the 4th edition of the textbook
Read chapter 7, and

Skip pages 606-608
Skip pages 637-638

For the 5th edition of the textbook
Read chapter 7, and

Skip pages 667-669

Note: always read skipped pages at your leisure.