S38.3115 Signaling Protocols – Lecture Notes

Lecture 10 – Media transport over a packet network

Contents

This lecture gives a technical background to transporting media over the Internet and IP networks overall.

Introduction

Traditionally, voice services have been provided using circuit switched networks such as PSTN, ISDN and GSM networks. The volumes of *data traffic* in wide area networks have been growing at rates much higher than the volumes of voice traffic. The growth rates of voice traffic have historically been a few percent per year. The historical growth rates of data traffic in the Internet have been between 50 and 200% per year.

The result is that the volumes of data traffic have overtaken the volumes of voice traffic (this happened around 2000 – 2002) making the moving of voice services to data networks a reasonable option. If during the early years of data networking, data was carried over circuits taken from the PSTN, now it has become reasonable to do the opposite: allocate some data network capacity for voice services.

Packet networks such as the Internet were initially designed and built for data services. The Internet became a commercial service in 1995 and was an immediate success due to mainly 2 services, namely, World Wide Web and E-mail. These services can easily tolerate packet delays and losses. The latter are corrected using a reliable transport protocol, in practice TCP. The protocol eliminates packet loss by acknowledging received packets and resending what got lost on the way. I.e. *loss is eliminated at the cost of further delay and data overhead*.

The delay of a single packet across the Internet is in the order of tens to several hundreds of milliseconds end-to-end. Usually, it is considered that a user can wait for one to two seconds for a new screenful of data to appear without getting irritated. The best effort transport with the current level of delays and packet loss is quite adequate for creating such an experience to the end user.

What comes to packet loss, it is a normal condition of the Internet. The idea of TCP is to ramp up its sending rate until packet loss occurs. After that the sender cuts its rate to a half and ramps up once more. This is how the end-to-end TCP adjusts its sending rate to the capacity of the network. Packet loss at the peak rate takes place in the bottleneck link of the connection. One can also say that TCP/IP networks require packet loss to work properly.

From 1995 onwards the transmission rates of the links in the Internet as well as router forwarding capacities have been growing at a fast pace in order to keep up with the growth of traffic. Lately, introduction of higher capacities into the Internet has slowed down somewhat compared to the events of just after the turn of the new millenium. The historical trend of traffic growth in the Internet has been similar to Moore's law, i.e. exponential. The traffic volume has about doubled each year for a long time until recently. During the early years of the new millennium, the growth was attributed largely to peer-to-peer. However, the legal battles over copyrights have taken a toll on the traffic growth during peaks of legal activity over the past few years. Moving all video services to the Internet has now become the reason for traffic growth. Nevertheless, the traffic growth has slowed down from the historical trend of doubling every year but growth continues at a much faster pace than the growth of voice traffic (in the order of doubling each 2 years).

Like we mentioned, the volume of data traffic globally surpassed voice traffic probably in about 2002. The exact year depends on how to measure the traffic. In any case, at the moment the volume of data traffic far surpasses the volume of voice traffic that has been growing approximately at the rate of 5% pa.

With the growth of Internet capacity, link, node and end-to-end packet delays are coming down. Let us take three examples: a link at 2Mbit/s, 100Mbit/s and a link at 10Gbit/s all transmitting a packet of 1200 bytes, i.e. approximately of 10 000 bits. It will take

t = 10 000/ 2 000 = **5 ms**; t= 10 000/100 = **100 μs** or t = 10 000/ 10 000 = **1μs**

to carry that packet. From this we gather that from link speeds of 100Mbit/s upwards packet times on links are very low and the Internet has become capable of transporting not only data traffic, but much more.

At the moment, electronic link speeds in the Internet core reach 10 to 40 Gbit/s. An electronic link technology in the 802.1 series that reaches 100 Gbit/s has been developed by the industry and is becoming commercial. Initially it will be deployed in places like the data centers. 100GE will require high quality Fiber rather than just *any* single mode Fiber, so deployment in the wide area will take time. On a link between two large Internet nodes the total capacity can be higher than what one electronic interface can do. The total capacity can be multiplied in the optical domain and/or adding more Fibers.

It is not enough that the Internet core can transmit gigabits. Access network speeds have always been the bottleneck. The reason is that the cost of the access network far surpasses the cost of the core. Nevertheless, operators and vendors introduced ADSL and later ADSL2 and ADSL2+ technologies allowing offering access capacities of up to 24 Mbit/s to end users at homes. Now a VDSL2 technology has become commercial for symmetrical user access speeds of about 100Mbit/s (the max speed in xDSL technologies always depends on the length of the copper connection). Alternative access technologies using Cable TV networks and Fiber to the Curb (FTTC), Fiber to the building or basement (FTTB) or even Fiber to the Home (FTTH) are also available and are being deployed in several countries. The factor slowing down the deployment of Fiber is not the cost of communications technology but rather the cost of digging the trenches and doing other civil works. The latest mobile broadband technology, LTE (long term evolution) makes it feasible to transfer user data at about 100Mbit/s in a cell that is shared among all active users in that cell.

Figure 10.1 shows the Global penetration of ICT services among them the growth of wire line and mobile cellular broadband subscriptions. The numbers are per Millions of subscribers/users. One should also note that, additionally, many users have access to broadband Internet at work.

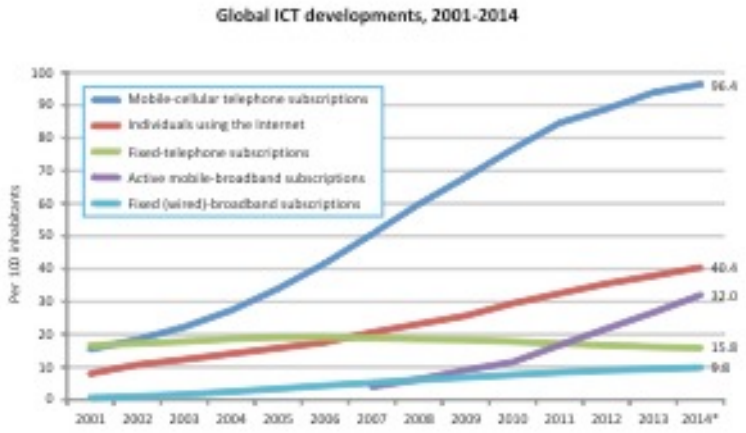


Figure 10.1 Growth of Global Adoption of ICT Services[1]

At the end of 2013, the share of different wire line broadband access technologies in OECD countries are approximately 52% for DSL, 31% for Cable TV, more than 16% for Fiber and less than 1% for other (in practice wireless fixed broadband)[2]. Over the past few years Fiber has been growing its share at the expense of DSL while Cable TV is holding up or even growing. On top of that, Figure 10.1 shows that over the past few years, mobile cellular broadband penetration has overtaken fixed broadband penetration. There are some 700M fixed broadband and more than 2 Billion mobile broadband users.

More statistics on broadband and Internet use is available in http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2013-e.pdf.

It is curious to see that in leading broadband countries the competition has moved from increased penetration to quality and that Fiber to the home is gaining ground in those countries. See Figure 10.2. Over the next couple of years we will see how this trend will be impacted by LTE and mobile broadband in general.

This development has made it very attractive for operators to phase off the low capacity legacy voice networks and use the data network cores to transport also voice traffic. The emergence of broadband subscribers has made it possible to offer end-to-end voice services over the Internet.

[1] At the time of updating this document, World population was 7,3 Billion. Internet users: 40% ≈ 3Billion.
[2] This is information from about 2011.

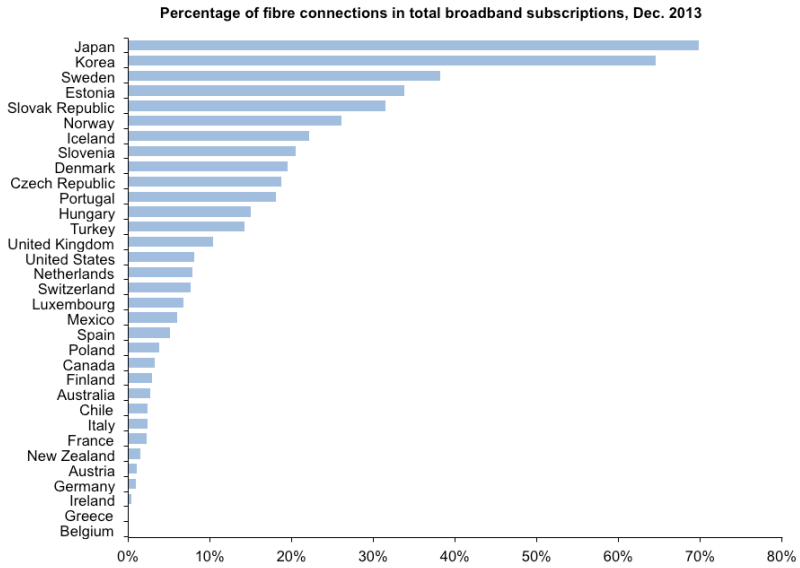


Figure 10.2. Share of Fiber in leading countries in broadband subscriptions

Data network technology facilitates the following deployment models for the purpose of phasing off the old circuit switched voice networks:

A. **TDM emulation**: PDH transmission links are replaced by data connections. For example 20 consecutive timeslots of one PDH channel are buffered and sent in one data packet using for example the 802.1 -technology for transport. Assuming 18 bytes for 802.1 framing and other data link protocol overhead and a few bytes for TDM emulation overhead, this solution reaches a payload to used capacity ratio of around 50%. Alternatively, if an operator can replace a whole PDH system with an Ethernet based link. In this case all timeslots from several frames can be packed into an Ethernet frame.

   In these solutions, the operator retains its TDM switching centers and all signaling and service systems that it has for providing voice services.

B. In addition to the transmission plant, an operator may choose to **replace its signaling and call control infrastructure** as well. The new signaling system may be a variant of an older signaling system from the digital era. For example, H.323 signaling inherited its framework and style from ISDN signaling. Alternatively, any new (native IP based) signaling system could be used. This is the approach taken by BT in the UK.

   The BT solution uses Multiple Services Access Nodes (MSAN) instead of DSLAMs that are providing ADSL access using the operator's copper access network. An MSAN can have ADSL, PSTN and ISDN line cards. Traffic from an MSAN is routed to different network systems based on the nature of the traffic. The BT solution uses the so-called Next Generation Network (NGN) that is based on the IP Multimedia Subsystem (IMS) for voice and multimedia signaling and

call control (session control). In the solution, PSTN customers may choose to keep their old telephones while the operator changes its circuit switching infrastructure into a new packet based infrastructure. The viability of this solution is yet to be proven. Investment decisions have been announced and testing is ongoing but wide scale deployment is slow.

C. The most advanced approach is to offer **end-to-end packet based voice services** over the broadband infrastructure. This means that users must have ADSL or another broadband modem at home and that they are likely to have Internet access in parallel with voice services using that broadband connection. Phones may look like phones or calling can take place using a computer. Compared to B, in this model packets are created and consumed in the terminal while in option B packetization and de-packetization of legacy analogue and TDM services take place in the MSAN.

   This option has several business models. The end-to-end voice services may be provided by the incumbent operator, an ISP, or for example, a global player like Google or Skype (e-Bay/…MS). Depending on the business model, different technologies for signaling and call control can be adopted. For example SKYPE uses proprietary signaling and the so-called peer-to-peer model.

From Circuit transmission to packet transport

In PSTN, ISDN and GSM networks, digital transmission was initially based on PDH systems. We have described them on Lecture 1. Synchronous Digital Hierarchy or SDH standardization started on the second half of 1980's and the systems were gradually deployed into real networks starting in early 1990's. SDH has two variants, SONET in ANSI markets and the SDH proper in the rest of the world. The principles are the same but transmission speeds are different. A lot of SDH systems are still in use.

SDH speeds start at 155Mbit/s allowing carrying some 64 PCM signals. The interleaving of PCM signals into one SDH signal is such that small variations in the clock rate in different PCM signals do not create transmission errors due to SDH framing because the variations can be compensated within the SDH framing. Up-to 256 basic SDH signals can be interleaved onto a higher order SDH signal. This means that the classical SDH supports electronic transmission speeds of up-to 40 Gbit/s.

SDH is a *carrier grade* transmission system. This means that it typically has a comprehensive management system that allows allocating transmission capacity for different traffic flows accurately (property 1). Operator defines for example a transmission pipe of 4 Mbit/s from point A to point B. The converse also applies: unless such allocation exists, nothing is transmitted (property 2). In addition, Carrier grade means that the transmission is reliable and in case of link failures, recovery takes place quickly (in less than 50 ms). In SDH this recovery is achieved for ring topologies.

SDH allows pointing PCM frames and timeslots in the SDH frame structure. The basic unit of transmission and switching in SDH is an octet i.e. the content of one PCM timeslot. When a physical SDH connection, e.g. STM4, is established between two sites, it means that irrespective of the actual use of network capacity, 622Mbit/s will flow between the sites at all times. Due to these properties SDH is rather costly. One benefit of SDH and PDH is that clocking signals for network nodes such as cellular base stations can be generated from the transmitted signals. For example, 3G WCDMA base stations need accurate clocks to work efficiently. One can reduce costs in a packet transport network by implementing only a subset of SDH functions on layer 1.

SDH transmission takes place in frames that are sent back-to-back. These frames do not carry addresses. Rather the frame must be split to octets for switching purposes. Due to lack of addresses in the frame, managing switching state in SDH nodes is based on network management systems that must install, monitor and change the state in the nodes based on the placement of a particular octet in the frame. Because of the lack of addresses, frames must follow each other back-to-back and thus transmission is synchronous. Each user gets an allocation of a link of constant speed.

In packet networks instead of transmission, we talk about *transport*. While transmission is about moving continuous byte streams of indefinite length, *transport is about moving addressed packets or frames of finite length*: usually there is a minimum length and a maximum. Transmission is a layer 1 service in OSI. In OSI terms, packet transport involves more than layer 1, at least layer 2, i.e. framing and link level addressing are needed. In case of native packet transport over “a wire” selling a layer 1 service is not an option anymore (such a thing can be emulated, though). Instead, as a minimum, the operator sells a layer 2 service. This is a fair reason not to talk about transmission anymore but about a *transport* service. Note, that in this context the word is used differently than in OSI. In *OSI the transport protocol layer* implements a means of data transfer (usually a reliable one) *end to end*. Packet transport is between 2 points in the operator’s network and reliable delivery is not targeted with the cost of additional delay like for example in TCP.

An industry trend is moving from synchronous byte oriented transmission to (native) *Carrier Grade Ethernet (CGE)* or *Carrier Grade Ethernet Transport* (CET) or any kind of packet transport such as *IP/MPLS* or MPLS-TP is used to bring the carrier grade control to the network. CGE means that IP is not used on the transport layer, rather all necessary control functions are implemented natively in the 802.1 framework.

A fundamental difference between synchronous SDH and packet-based transport is that frames in SDH do not contain addresses while packets contain them. Because of this, SDH can only provide constant rate bit streams (a node can know where to put a part of a frame only based on the placement of the part in the frame). In packet transport, a frame and a packet are the same thing and will always contain at least one address. Because of this packets in a flow do not need to be sent back-to-back or evenly spaced. A node receiving a packet can use the address in the packet to direct the packet onto the right outgoing port. Such forwarding decision requires knowledge of the topology. That

knowledge can either be stored in routing tables in the nodes or carried in the packet itself.

Ethernet is asynchronous. A frame starts with a *preamble* string for waking up the receiver. This is followed by a framing flag and the layer 2 protocol header information. Between two frames nothing needs to be transmitted. One result is that timing signals cannot be easily generated from an Ethernet signal. Solutions for this purpose are being investigated, standardized and designed. The winning standards will establish themselves on the markets in the near future.

The original corporate network Ethernet is not considered Carrier Grade. The reason is that corporate Ethernet does not need nor comes with a management system that would try to enforce the carrier grade properties defined above. In addition, an 802.1 switch broadcasts frames containing an unknown destination MAC address onto all outgoing links. This is part of the MAC learning algorithm. In CGE and CET, broadcast of unknown frames and MAC learning are disabled because they violate the fundamental properties of what the term “carrier grade” means.

The development of CGE is under way. We are likely to see new products and new more flexible ways of using 802.1 for packet transport in the future.

One milestone was reached in 9/2010 when the *Energy Efficient Ethernet* (EEE) for copper became an IEEE standard. An EEE connection has a fast sleep-wake-up algorithm meaning that power is turned off while there is nothing to send. This is similar to how mobile devices save power. It has been claimed that EEE will help to save hundreds of Millions of Euro each year in energy costs if and when it becomes widely deployed.

Packet transport differs from circuit transmission also in that the transport system tries to make use of *packet level statistical multiplexing* in order to achieve higher efficiency. The result is that in a transport node, at some point in time, more packets may be headed onto an outgoing link than what is the link capacity. *Queuing and scheduling* of packets onto the link are needed. This makes end-to-end delay unpredictable i.e. a stochastic variable. In circuit transmission, delay tends to be deterministic and can usually be calculated on an end-to-end connection. Under extreme conditions, even packet loss in a heavily loaded packet transport node is possible. It happens e.g. when a queue of an outgoing link overflows. We can avoid packet losses in a packet transport network by rate limiting the incoming traffic and by careful dimensioning, similar to how circuit networks are dimensioned.

We need to make use of statistical multiplexing on packet level because in data networks peak rates are many times higher than average data rates. The result is that we need a resource management system including some type of quality of service functions. These issues are discussed on a different course. Here we will look at packet transport from a voice application point of view.

To summarize: the driver for change from synchronous transmission to packet transport is cost. Investment cost for transmitting an asynchronous Ethernet bit is just a small fraction of the cost of transmitting the same bit over SDH. Common sense tells that OPEX is a function of the network topology and reach and to some extent of capacity. The latter relates to the fact that high bitrates require fast clock cycles in electronics and thus consume more power. The idea of power saving using sleep-wake-up algorithms is much easier to implement in packet transport than in continuous bit stream transport. Therefore, a data network with higher capacity can have lower OPEX/bit than an SDH network of the same reach.

Scaling a broadband packet network for voice

Modern voice codecs produce a bit stream at a rate ranging from a few kbit/s (e.g. 8 kbit/s or even less) till a few tens of kbit/s (e.g. some high quality codecs produce 80 kbit/s). In addition to the voice bits, packet overhead must be carried as well.

For a scaling exercise let us assume that the total packet stream for voice will use 64 kbit/s. Let us calculate how much capacity is needed to carry all simultaneous telephone calls in Finland on a single link. Let us assume that 70% of calls are local and will never be offered onto that link. If there are 4 million users, there can be no more than 2 million simultaneous conversations because the share of international calls is insignificant. A fair assumption is that only 10% of people are engaged a telephone conversation during the busy hour. Then the required capacity is

$$C = 0{,}3 \times 0{,}1 \times 2\,000\,000 \times 64 / 1000\,000 = 3{,}8 \text{ Gbit/s}$$

We can see that probably with the link capacities (10Gbit/s) that have been available off the shelf for many years on reasonable prices replacing the PSTN with a data network is not a problem.

One must assume that there is competition in the core and even one operator will want to have some redundancy in the core. Consequently, the required overall capacity of some 4Gbit/s is split among several links.

Model of packet transport for media

Broadband packet networks can carry data, voice and video. We talk about voice transport and media transport over packet networks. Media includes voice and video either in an interactive or in a streaming mode. In an interactive setting, two or more users are communicating in real time. Streaming is used to carry for example radio or TV channels or large video files over the packet network. In streaming, data flow is mostly unidirectional (with the possible exception of some asynchronous feedback for control).

Figure 10.3 shows a model of voice transport over a packet network. Voice is sampled, quantized and encoded in the terminal using some coding standard. The coding standard (for example G.711 used in PSTN) may produce samples that are just 8 bits in length. Obviously, sending one octet in one packet would lead to an extremely inefficient use of

the network. Therefore, coded samples are buffered in the sending terminal into *voice frames*. It may be that one or several frames are packetized into one IP packet for transport over the packet network. For example, one packet may carry 10 to 80 ms of voice.

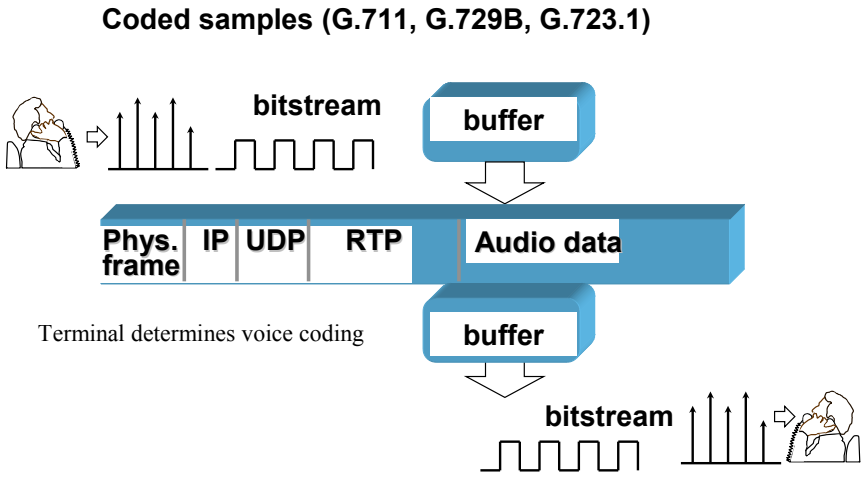


Figure 10.3: Model of voice transport over IP.

The encoding terminal can choose to use any (even proprietary) coding standard that is also understood by the receiver. The network will see packets and it normally has no reason to even look at the payload. Only in a hybrid packet to circuit network or circuit to packet network call, there may be a media gateway in the network on the technology boundary. Then that gateway needs to terminate/originate the packet voice stream and needs to agree to use a particular coding standard possibly suggested by the terminal. This is different from PSTN where only G.711 either μ-lay or A-law coding is used.

For voice transport, an IP network needs to use some end-to-end transport protocol. In case of interactive voice TCP is really not suitable because some voice packets would be delayed due to retransmission used by TCP to correct loss. Such delayed packets are useless and only degrade the quality. Instead, usually UDP, a connectionless unreliable protocol, is used for transporting voice. However, UDP does not carry any timing information. When a voice packet is carried over the IP network, it will experience unpredictable delay. To reproduce the same voice signal that the talker uttered, we must be able to play the received voice frames at the same pace they were produced.

The receiving terminal and the sending terminal have independent clocks. Without timing information, the receiver would not know when to play a received frame. The transport system cannot assume that each packet will contain exactly the same amount of voice, nor that the packets are sent strictly periodically. To overcome these problems several methods can be used. The voice coding standard itself can contain timing information (for example MPEG signals contain timing). In an interactive session, we often use the Real-time Protocol (RTP) in pair with the Real-time Control Protocol (RTCP) to carry timestamps and order numbers of the samples.

Figure 10.4 gives further details of voice carriage over a packet network.

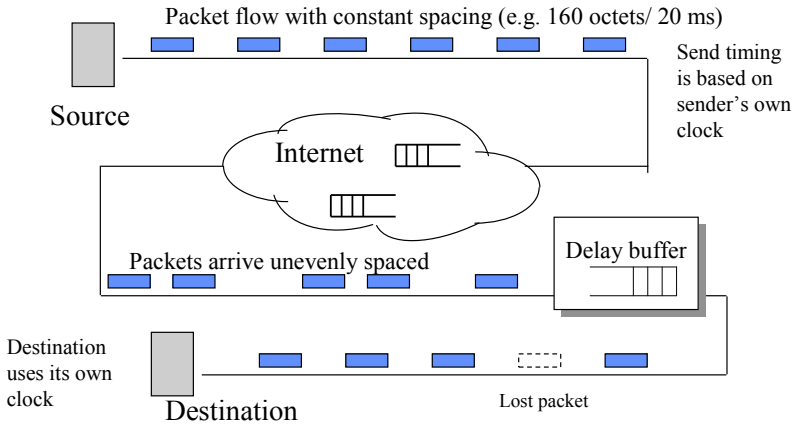


Figure 10.4: Timing of voice transport over IP

Timing information carried in the packet is based on the sender's clock. Stability of clocks for example in PCs is far worse than for example the stability of clocks used on PSTN nodes. The IP network itself is a network of queues resulting in unpredictable delay on an end-to-end connection. The call can, however, succeed and have even high quality than in PSTN, if a sufficient portion of the packets arrive within a limited network delay of e.g. 60ms. Packets that take longer than the current limit are as good as lost. The limit is lower for interactive voice than for streaming.

The receiver initially places the received voice packets into a *play-out buffer*. Filling of this buffer takes place at the pace the packets arrive from the network. The purpose of the buffer is (1) to ensure that while the sender is talking the receiver always has something to play. In addition, (2) the order of the received packets needs to be checked. If a packet is missing in a sequence, it may be that the packet has been lost or arrives later than a packet that was sent after it. If a late packet still arrives within a time bound, the receiver will try to push it into the right place in the play-out buffer. For play-out (3) the buffer is read at the pace of the receiver's clock. The length of a talk spurt is in the order of one second. During that time the timing difference between the sender and the receiver will have little effect on the quality of voice.

Delay and voice quality

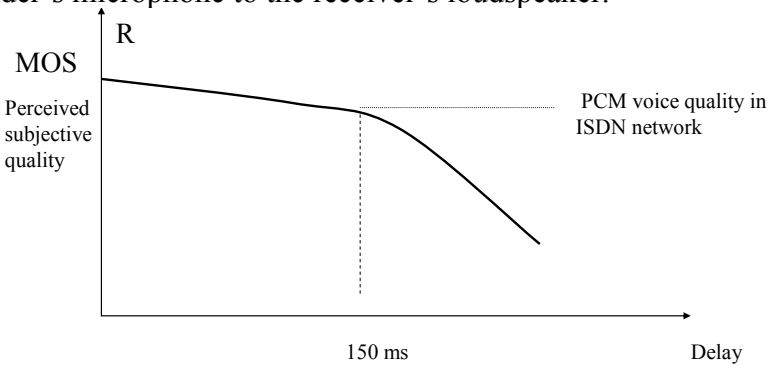
Voice experiences different *impairments* when it is carried over a packet network. Impairment is any factor that reduces voice quality. One of them is delay. Figure 10.5 shows qualitatively, how the quality of the conversation changes as a function of end-to-end delay. The vertical axis shows the R-value that is an *objective measure* of voice quality according to the E-model.

Alternatively, for voice quality assessment, *subjective testing* can be used. In subjective tests a group of people listens to a set of voice samples and each gives his or her opinion of the quality. The result is a Mean Opinion Score (MOS) ranging from 1 to 5. Measuring MOS for interactive voice and a large number of samples is obviously very cumbersome

and costly. For this reason and for cases when such tests need to be repeated often in an R&D process, methods for objective measurement of voice quality have been developed. An objective measurement of voice quality tries to guess or estimate how humans would perceive the quality. Current objective measurement results give reasonably high correlation when compared to MOS measurements with a group of human listeners. Nevertheless, in this context one cannot say that objective is better than subjective. For sure it is less expensive to obtain than a subjective measurement. But "true human perception" can be measured only by humans.

The figure shows that quality degrades slowly until the end-to-end delay reaches 150…170 ms. Above this threshold, the conversation gradually changes to half-duplex. This is not what ordinary users want, nor are used to. Half-duplex communications is still used for example by the military. Half duplex means that each talk turn ends with a word "over" or some such "talk protocol" must be used by the humans themselves.

The end-to-end delay is the delay from mouth to ear. In practice we measure it from the sender's microphone to the receiver's loudspeaker.



Quality can be measured e.g. based on the E-model or using MOS –measurements.
MOS - Mean Opinion Score.

Figure 10.5: Voice quality as a function of end-to-end delay.

Components of packet delay in voice transport

Table 10-1 shows the components of packet delay in a practical voice over IP system. The values are based on measurements in a PC environment.

Delay component	ms	Explanation
Audio HW &device driver	0-100	Buffering
Algorithm	20-37.5	Sample length + lookahead time
Operating system	0 - 30	Depends on load and implementation
Coder	<5	Predictable delay in coding algorithm
Decoding	<1	Typically an easy process
Framing and packetization	<1	A small software delay
NIC and device driver	<5	Has some signifigance especially in WLAN
Network	0 - 500	In LAN about 1 ms, Dimensioning Issue!
Play-out buffer	0 - 100	At reception,depends on the state of the network
Synchronization	0 - 30	Audio device requests for data at constant intervals that can not be synchronized with packet arrivals. Avg = half a packet time

Source: M.Sc thesis by Jari Selin

Table 10-1: Components of end-to-end delay in a VOIP system.

From the table, we can conclude that network delay is a matter of network engineering but that PCs are usually lousy phones[3]. Network delay over a single Ethernet link is in the order of 1ms and usually less than 10ms in a campus network. Network delay grows as a function of the number of routers and links between the sender and the receiver. Unfortunately, unlike PSTN, the Internet has no strict engineering rules that would tell how many routers a packet may pass on its way between the sender and the receiver on the globe. Increases in network performance have been accompanied by flattening of network topologies. We say that Internet diameter is decreasing. This helps to keep the number of nodes a packet will have to cross within certain bounds. Unfortunately, other trends tend to increase the number. An example is the application level gateways on administrative boundaries that have been emerging lately in order to tackle network trust concerns.

The acoustic properties of PCs leave a lot to be desired. A PC may introduce noise from fans and the disk and if the operating system is not tuned to run real-time applications, the PC may be unsuitable for VOIP. Due to the popularity of Skype and other voice apps, modern PCs and MACs have become suitable for interactive voice.

One should also note that synchronization of the audio device with the CPU usually requires that the audio device reads a buffer in the main memory using the clock of the audio device. This may introduce a time shift between the CPU and the audio device. The maximum value of this time shift equals to the length of the sample read by the device at a time. By a well-engineered implementation of the interface between the audio device and the CPU, this time shift can be minimized. A similar situation exists between the microphone and the CPU.

[3] The measurement was done when VOIP first appeared. Modern PCs can be expected to do better.

The second conclusion from the table is that the total delay in a PC terminal is quite high compared to the delay budget of 150ms that we should be targeting. This means that there is room for dedicated phone-like devices for VOIP on the markets and that the networks need to be really high capacity and well engineered in order to trust them to perform well in voice applications.

NB: The measurements we refer to were done in late 1990's. Modern PCs and their software are much better with voice. However, it is good to be aware of the possible pitfalls.

Play-out algorithms

Figure 10.5 shows how delay impairs voice quality. Besides the delay itself, one should note that the delay varies over time. The variability is not smooth, rather the delay changes abruptly, i.e. exhibits spikes. *Jitter* is a measure of the variability of delay. We can for example define *jitter as the difference between the maximum and the minimum delay*. A more practical measure of the variability is *packet spacing difference* (*d*). We define

$$d = (t_{r1} - t_{s1}) - (t_{r2} - t_{s2}) = (t_{r1} - t_{r2}) - (t_{s1} - t_{s2}) \qquad (10.1)$$

Where by 1 and 2 we mark consecutive packets and by s, time of the source at the instant of sending and by index r the time of reception by the receiver using its own clock.

The right hand form of the formula shows that the time shift between the clocks of the sender and the receiver has no impact on the value of *d*. Moreover, a receiver can calculate *d*, provided that each packet carries a timestamp from the sender.

The formula gives an instant value of packet spacing difference. We can easily smooth the value by some form of averaging. The more we average, the more information about the spikes is lost. The task of optimizing play-out for quality involves averaging *d* to such an extent that not too many packets will be lost due to spikes within the play-out delay.

The play-out may be based on a constant delay of wall-clock time between the sender and the receiver. In practice, this may be difficult because the value of time shift between the sender and the receiver is not available at the receiver. The play-out delay should compensate for the maximum packet spacing difference capped at e.g. the 99% fractile. This would mean that we dimension the system for example to loose 1% of voice packets in order to keep the end-to-end delay within a limit.

Choosing the right play-out delay limit is not trivial because there are two types of loss. Some voice packets are really lost and some are just delayed more than a limit. A receiver may try to increase play-out delay in order to reduce loss. The idea is to trade delay impairment for better quality due to more packets received. If the real loss in the network is rather high, it may be that the receiver ends up with unnecessary delay

impairment without gain. If some of the "lost" packets were actually just delayed more than the current play-out delay limit, the result would be an optimization task between improved quality due to more packets received vs. delay impairment due to higher play-out delay.

Because connections over the network are of variable length, the average delay over different voice calls may vary significantly. The delay also varies during a single call. For a high quality interactive conversation, the E-model shows that end-to-end (mouth to ear) delay should be kept under 150ms (Figure 10-7). It is possible to change several parameters at the sender if there are problems with voice quality due to, for example, high network delay or jitter. One thing that the receiver can do is to *adjust the play-out delay dynamically*. Numerous algorithms have been developed for that purpose. The challenge is that network delay does not vary smoothly. Instead, spikes in the delay are a norm. Defining an adaptive algorithm that will adapt quickly to a signal with spikes is not easy. Many algorithms will measure the signal for an extended period before making any adaptation. The measurement period may be even longer than the whole conversation. A well working adaptive algorithm will shift the play-out time only during a period of silence. Adjusting timing during a talk spurt would introduce additional voice impairment instead of improving voice quality.

Another challenge is to choose the time for playing out the first sample of the conversation or a talk spurt. This challenge is created because the wall clocks of the sender and the receiver can have a time shift the value of which is unknown. One approach is to choose a safe value at the beginning of the call and adjust the timing dynamically based on the measured packet spacing difference and the play-out buffer state.

Dealing with packet loss

Packet loss is another factor of voice impairment in packet networks. The receiver must compensate for packet loss in calculating the packet spacing difference. More importantly, the receiver can try to compensate for the lost content in the play-out. For the purpose of detecting packet loss, the sender needs to number all sent packets with consecutive numbers. The receiver, after reordering incoming packets within a time window can then easily detect loss from the gaps in numbering.

One easy method for *compensating the lost voice content is to replace it with the contents of the previous packet*. If the conversation did not take place in real time, one could also make use of the next packet. Another method is adding redundant voice data into the media packets at the sender. One could for example interleave the samples to an extent. This method increases the use of network capacity for the media. However, it may be that we do not care because there is a lot of capacity and many other applications are even more capacity hungry. This aggressive approach assumes that packet loss is rarely higher than e.g. 10% and goes on e.g. to encode each short voice sample and place each short voice sample into two consecutive packets. Each single packet loss can be compensated

due to redundancy. Only when several consecutive packets are lost, the receiver will see data loss in the voice stream.

Quality of service

Instead of dealing with packet loss and delay in the application, one could try to improve the packet network itself in order to make it more suitable to carry all kinds of traffic. Under this flag, a lot of work has been done in IETF and in the academia. The results for IP networks include two QoS architectures. The first was the Integrated Services (IntServ) architecture. When that had been specified, most people felt that it had serious scalability problems. A new effort was launched immediately. The result was the Differentiated Services (DiffServ) architecture.

Modern routers support DiffServ but the capabilities are often not used. One reason is that if DiffServ routers are not properly tuned, QoS of some users will become worse than in a Best Effort IP network. The reason is that DiffServ partitions the network into a small number of parallel networks called classes. Capacity usage limits and rules are set for each class on each link. Consequently, if significantly more traffic of certain class arrives on a link than was expected, the quality of that class may be less than what the target was. It is even possible that a higher class will have lower quality than a lower class because of the unexpected split of traffic into the classes.

The traditional method to deal with QoS in the Internet is over-provisioning. When building a new network to replace an old one, one dimensions the network, for example, by multiplying the expected traffic by say 13. This sounds like an expensive method. However, the cost of transmission grows less than linearly as a function of link capacity and the operator avoids creating operational expenses due to the need to manage the network continuously.

Nevertheless, the question of using some QoS features is a business decision that the operator and the corporation having its own network must make.

A model of media processing

Figure 10.6 shows what functions are required in a terminal and also in a media gateway for media processing. The sender and the receiver are similar except that the receiver has the functions related to the play-out buffer.

Each device has two separate cards and the processor. One of the cards is the network interface card (NIC) and the other interfaces the real world. The CPU usually runs the TCP/IP stack and the drivers in the kernel mode and the audio application in the user space. Modern laptops and desktops have no difficulty in running a real time media application for one or even several users. If more capacity is needed for example in the media gateway, digital signal processing hardware can be used to process media. On the other hand, software for low end portable devices may need to be carefully optimized to

run an interactive voice application smoothly. Over the past few years even this problem has been solved by Moore's law, i.e. ever more cpu power in small devices.

Real implementations have buffers also at the interfaces between the CPU and audio devices.

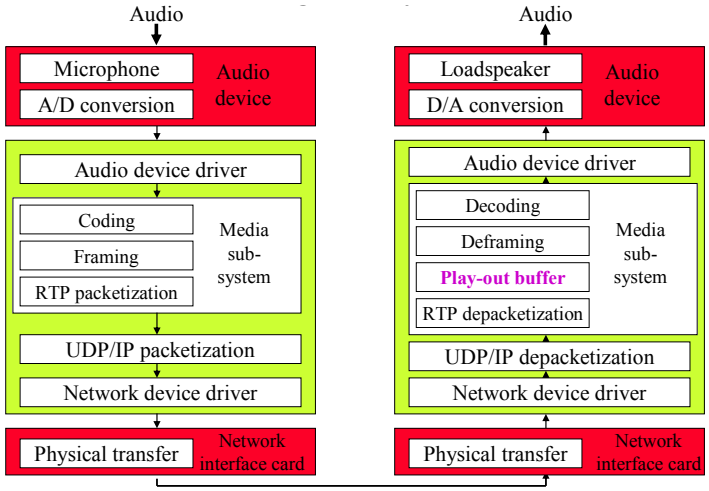


Figure 10.6: Media processing in terminals and media gateways.

Transport of media flows over IP

Basic protocols

IP needs a layer 2 protocol for framing the packet i.e. for showing where the packet starts and where it ends. An example is the Ethernet (or IEEE 802.1). The minimum Ethernet overhead per packet is 18 octets long containing two MAC addresses, a checksum and the length/Ethertype field (a preamble of 7 octets and a start of frame mark may be needed as well). Additional layer 2 features such as VLANs make the header longer. Another example is ATM. It uses 5 octets for a cell header and a fixed payload of 48 octets. An adaptation layer will use one or more octets from the cell payload.

The Internet uses mainly IPv4 at the moment. IPv4 packets have a variable length header and a payload. The IPv4 header is presented in Figure 10.7

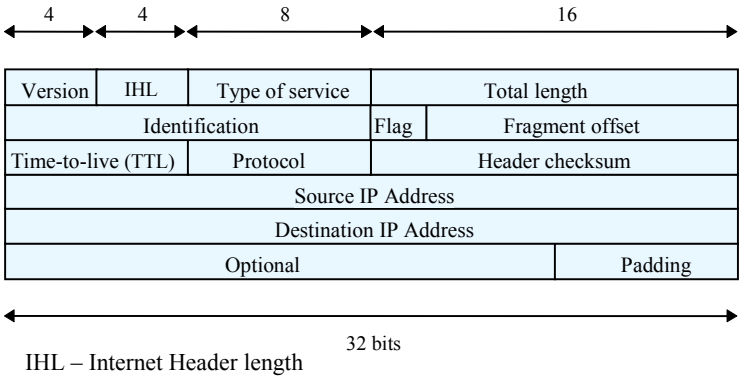


Figure 10.7: IPv4 header.

The minimum length of the IPv4 header is 20 octets. If the underlying network requires a minimum length for the packets that it carries (e.g. 802.1 in full duplex form), padding may be needed at the end of the packet. One should also note that the IPv4 header has a header checksum but it does not help to detect bit errors in the payload in any way.

Media transport over IP most often uses the User Datagram Protocol. UDP header is presented in Figure 10.18.

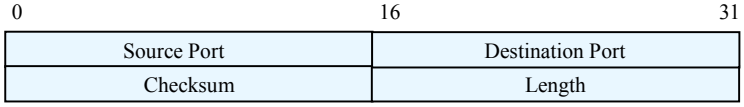


Figure 10.8: UDP header.

Usually the port numbers are used to identify the application that the host uses to process the packet. On this level, the checksum covers also the payload so bit errors in the payload can be detected. The length of the UDP header is constant and equals to 8 octets.

UDP is an unreliable transport protocol and has no acknowledgements, so the reliability of packet delivery is left for the application. The most common *reliable transport protocol* in the Internet is TCP. TCP traffic in the Internet is usually more than 80% of all traffic. TCP is not particularly suitable for real time media transport because it uses acknowledgements to ensure reliable delivery at the cost of increased delay[4]. In case of real time media, it is better to deal with packet loss at the application level and strive for as fast delivery as possible. This is due to the need to keep the delay within bounds, like we have discussed.

Media transport over IP often uses RTP, the Real-time Protocol. We will discuss it in the next section. For now, let us just note that RTP introduces a header of 12 octets (as a minimum) into each media packet.

[4] The RTCweb architecture that is under development in IETF can carry voice also over TCP.

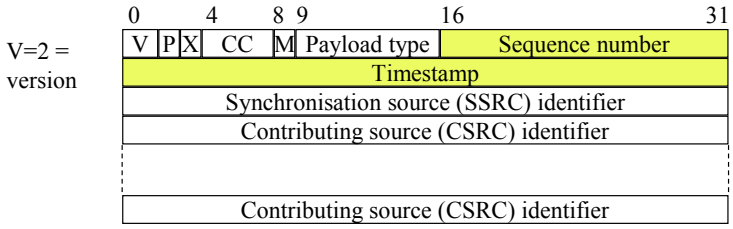
Real-time Protocol (RTP)

RTP supports the carrying of unidirectional media flows across a network. For a conversation two RTP flows are needed in opposite directions. RTP framing can also be used to store media e.g. on disks of some sort and play it out later.

From the previous discussion of packet delay, jitter and packet loss, we conclude that the sender needs to send its timestamp and number the media packets. These are the main features of RTP. RTP header is presented in Figure 10.9.

An RTP flow is based on the model that there is a *sender* or several senders, there may be an *optional processing element* on the way and there is at least one *receiver*. RTP has one message type and RTP messages are usually sent over UDP, so there are no acknowledgements reporting correct delivery of each packet. RTP supports multicast sessions in which case there are many receivers and at least one sender. Multicast without intermediate processing elements would lead to each session participant sending media to each other session participant. Clearly, this does not scale. With the growth of the number of participants, the access capacity of a participant might be exhausted and this approach is hard on the processing power of a participating host computer.

RTP overcomes this scaling problem to a larger number of participants by introducing the media processing element. It can for example sum media streams coming from several participants and thus it is possible to deliver a single stream to each receiver.



P - Padding - indicates that last octet of payload = nrof preceeding padding octets
X - Extension - there is an experimental extension header
CC - CSRC count - Nrof CSRC identifiers following the fixed header
M - Marker - e.g. End of video frame, Beginning of talk spurt
Payload type - format of RTP payload.
Seq. nr - each source starts at a random nr and =+1 for each packet -
determines order of packets with the same timestamp
Timestamp - value of local clock at source at generation of first octet of payload
SSRC and CSRC identifiers are generated at random

Figure 10.9: RTP header

In RTP all elements that have contributed to the media content are identified by 32 bit *random identifiers* (SSRC). If the session has only 2 participants, only the single sender's id is present in every media packet.

The sender initiates the *sequence number* at the beginning of the media session and keeps incrementing it for each packet. The sequence number is used to detect packet loss. It is necessary because at the level of media transport we do not assume that the sample length

in each packet is the same, nor that they are sent at regular intervals (although most times these would be valid assumptions).

The *timestamp* is generated using the sender's clock. It is used to calculate the play-out time. Like we discussed earlier, the timestamp is also needed for calculating the value of jitter. Jitter is used to adjust the play-out delay. RTP by default calculates jitter by averaging packet spacing difference (formula 10.1) over consecutive measurements:

$$J(i) = 15/16 \times J(i\text{-}1) + d(i) \qquad (10.2)$$

Moreover, the RTP header has the *payload type* field for the purpose of identifying what codec is used for the media. The marker (M) may indicate a specific event depending on the *profile*. For example, M may indicate the beginning of a new talk spurt. Finally, the CC –field gives the number of random source identifiers in the packet.

Exercise: What is the probability that among 16 sources with uniformly distributed random IDs of 32 bits, two IDs have the same value? What is the probability if there were more participants (100, 1000 or 10 000) in the session? For how many users does this kind of randomly generated ID scale? Hint: look up Birthday paradox in Wikipedia.

RTP Control Protocol (RTCP)

RTCP usually uses the next port value to RTP. Figure 10.10 shows the main message types used in RTCP.

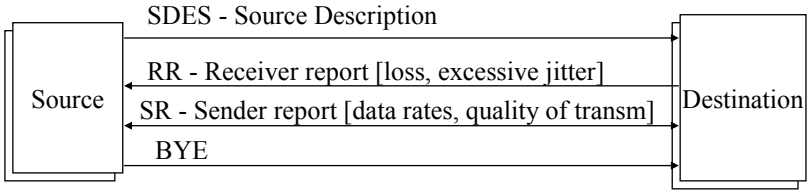


Figure 10.10 Main RTCP message types.

A receiver, that does not also send RTP packets, sends RR –receiver reports and a participant that both sends and receives RTP packets sends SR reports. SDES and BYE are used at the beginning and the end of an RTP session respectively.

Each RTCP message has a fixed header that gives the RTCP message type and e.g. the number of RR blocks etc. The useful control information is carried in the RR and SR blocks.

The SR block has the NTP wall clock time for round-trip delay measurements, the RTP timestamp for relating the SR report to the RTP stream and finally, the sender's packet and octet counts from the beginning of the session.

The RR report block (Figure 10.11) first identifies the sender in the same way that is used in RTP. The block gives the fraction of lost packets since the previous report and cumulative number of packets lost in the session.

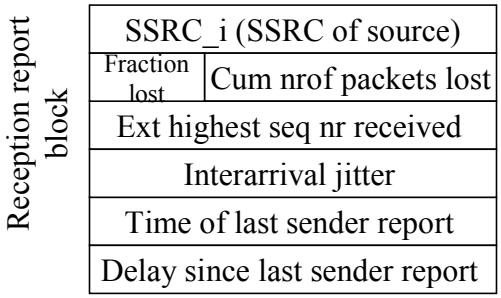


Figure 10.11: RTCP Receiver block contents.

The RR relates the report to the sequence numbering and time of the previous SR. The value of the calculated jitter is also sent to the sender. If loss is high or has increased, the sender can for example move to use a lower rate. This is possible because many codecs have numerous modes of coding with different rates. If jitter is high and the round-trip time is also high but loss is low, it may be a good idea to move to shorter samples. This will increase overhead and the summary bit rate including the overhead but this may be tolerated if the result is better voice quality.

Efficiency of voice transport over IP

In this section we will discuss the overhead created when voice is carried in packets. *Packet overhead* includes the *header overhead*, *framing overhead* and *padding*. Packet overhead contains all bits carried in a packet that are not used for media encoding. The media encoding bits are also called the payload.

Summing up the header overhead

Over a corporate Ethernet network, we have so far learned that voice packet overhead is:

802.1	18 octets
IPv4	20 octets
UDP	8 octets
RTP	12 octets
	58 octets

If voice encoding takes place at the GSM rate of 13 kbit/s, 58 octets corresponds to 35,7 ms of voice. Consequently, if voice samples are less than 36ms long, voice transport efficiency is less than 50%. If Ethernet is not switched but shared media is used, the Ethernet frame may have a minimum length that may be longer than 100 bytes. In that case padding needs to be added further decreasing efficiency. More padding is needed in faster networks. Fortunately, the share of voice traffic in a fast network is likely to be low because most of the capacity is used by data traffic.

An operator network may use for example 802.1ah. In that case MAC headers put together would use some 30+ octets (4 MAC addresses and 4 VLAN tags from 12 to 32 bits each).

In many networks, there is a minimum packet size. In case of ATM, the transport packet or cell size is fixed. It is impractical to expect that voice frames would fit into one cell exactly. Functions in networks are arranged into layers and one must assume that the layers are independent. ATM cell payload is 48 octets and AAL5 takes one octet.

So, in case of ATM the header overhead calculus looks like this:

ATM cell	6 octets
IPv4	20 octets
UDP	8 octets
RTP	12 octets
	46 octets

The first cell allocated to a voice frame would have to carry 40 octets of IPv4, UDP and RTP headers leaving only 7 octets to a voice sample. Consequently, most likely each voice sample is carried in 2 or more consecutive cells.

Let us take an example. Let us assume a case of coding at 13 kbit/s and 20ms voice samples (same as GSM):

The voice frame takes	13 * 20/8 = 33 octets.
The first cell carries	7 octets of voice
The 2nd cell carries	26 octets of voice + 21 octets of padding

The payload to total bits ratio is

$$r = 33/(2*48) = 33/96 = 34\%.$$

If we place 2 frames of 20ms into one IPv4 packet, the efficiency is

$$r = 65/(3*48) = 65/144 = 45\%.$$

We can conclude that ATM scales poorly for voice transport using low bit rate codecs.

Ways to improve voice transport efficiency

If the share of voice traffic in an IP network is for example 10% of all traffic, we may ignore the fact that transporting voice over IP is inefficient. However, for example, when we are using wireless access, this may not be economically feasible.

In an access connection, we may apply *header compression*. This means that the sender will maintain flow state for each voice flow. The algorithm might send the first packet in full but for the following IPv4 packets move to sending a header difference plus the payload. The difference between two consecutive headers including IP, UDP and RTP in a voice flow might be possible to pack into a single octet.

In order to use IP address information for routing, decompression of Mobile Originated voice packets must take place before the first router. In a fixed IP network we might choose not to care about the overhead. If we are using fiber transport and the share of data is significant, changing one compression method (IP + UDP + RTP) → (UDP + RTP) is probably not worth the trouble.

In a packet network, another method that we can use to reduce the amount of data in a voice flow is *silence suppression*. The idea is that half of the time (up to 60% of time) any of the two parties in a telephone conversation is silent. During silence, it is possible to send fewer bits than during a talk spurt. This method helps in the core of the network where statistical multiplexing of packets works efficiently. From traffic theory we know that if there are hundreds of voice flows on a packet data link, with high probability under silence suppression, the total data rate can be cut close to half of the rate that is needed without silence suppression provided that the flow rate during silence is close to zero. It is also possible to replace silence be generating so called *comfort noise* in the receiver. The idea is to avoid the listener getting an uneasy feeling that the connection has been lost.

It is worth noting that silence suppression, although it is applied end to end, does not really help in the access. The reason is that even in wireless access only a few users are sharing the link at the same time. The probability that every one of them is talking at the same time is still rather high. Consequently, the wireless link needs to be dimensioned by assuming that (almost) every user is talking at the same time.

Amount of overhead as a function of the sample size and bit rate

Let us plot the efficiency of voice transport as a function of the sample size. For a classical 802.1 network, this is presented in Figure 10.12. The figure assumes 18 bytes for the Ethernet framing.

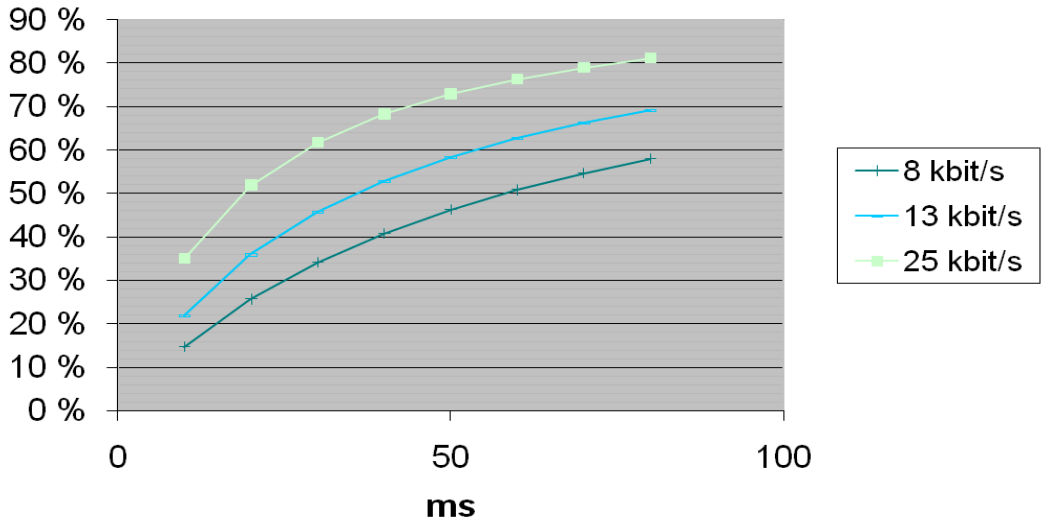


Figure 10.12: Voice transport efficiency over Ethernet as function of sample size.

The figure demonstrates that the higher the voice bit rate the less important is the question of efficiency. A high bit rate codec achieves a reasonable level of efficiency with a practical sample size. The figure also demonstrates that it does not make sense to try to do the utmost to reduce the bit rate by creating "more efficient" codecs. Even if one reduces the bit rate for a given level of quality with a clever coding method, only marginal gain is achieved. This logic was well understood by the designers of the first widely successful VOIP service, namely SKYPE. The application assumes that most users have a broadband connection and that a voice stream will use only a small portion of its capacity even if the coding is highly redundant and uses for example 80 kbit/s.

Comparison of voice telephony over IP and circuits

Table 10.2 sums up key differences between circuit networks and the IP network in terms of media transport.

Table 10.2: Comparison of circuit and IP networks in terms of media transport.

Criteria	**Circuit networks**	**IP network**
Coder placement	Network, may be also in terminal	Terminal and gateways
Coding standard	G.711, A-law and μ-law	Any standard that parties agree to use
Sample size	8 bits	Negotiated, ~10…80ms
Network engineering	ITU-T rules give max nrof hops	No rules.
Delay	Deterministic (more or less)	Random. A challenge for multi-hop connections and congested networks.
Jitter	Insignificant or none.	Varies. (random)
Loss	Very low	Needs to be compensated by

		the receiver.
Terminals	Dumb in fixed networks. In cellular have important role in managing mobility etc.	Intelligent. Mass market economies of scale help to make the best use of intelligence in terminals.
Relation to signaling	Media and signaling follow the same path hop by hop	Media and signaling use independent paths.
Overhead	Very low. Network is optimized for voice.	Varies. Can be high. May need to be reduced by header compression and silence suppression.