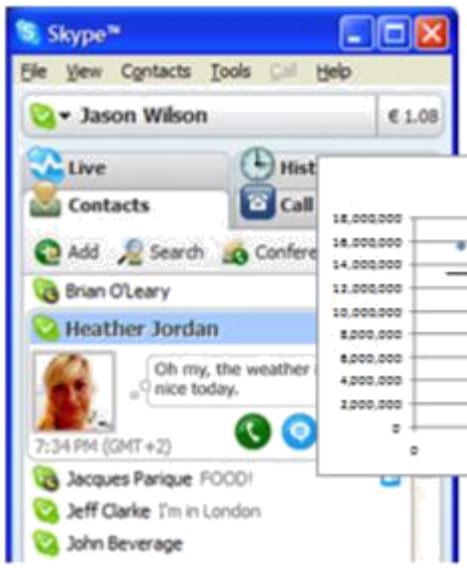
Outline

1. Introduction
2. The current telephone network: PSTN
3. Voice over IP: VoIP
4. Multimedia over IP

Skype: voice over the internet



Date	Online users	Days
2009-02-02	16,000,000	21
2009-01-12	15,000,000	84
2008-10-20	14,000,000	35
2008-09-15	13,000,000	209
		42
		84
		259
		82
		71
		155
		66
		92
2005-10-20	4,000,000	155
2005-05-18	3,000,000	93
2005-02-14	2,000,000	117
2004-10-20	1,000,000	418
2003-08-29	0	

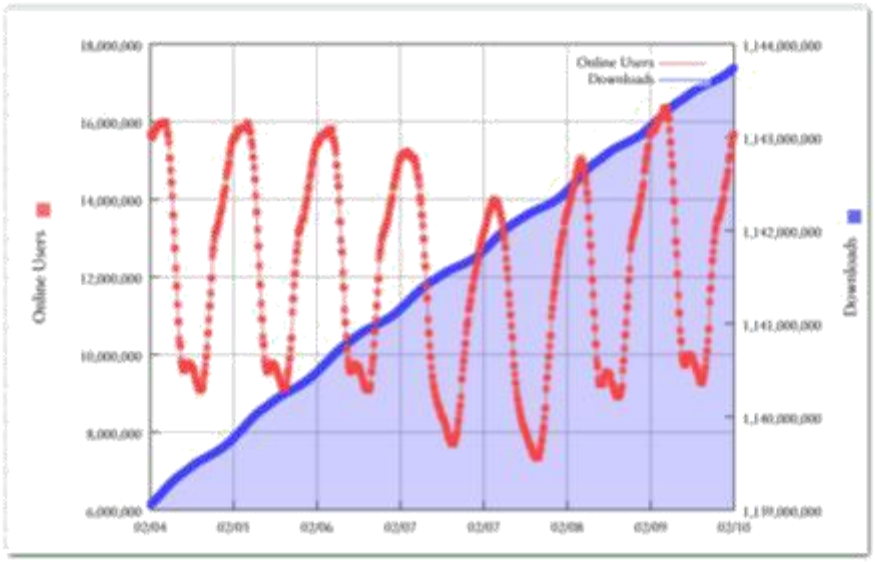
In Feb. 2009, the record of 16 million concurrently online users has been established.

Early Feb. 2009, the number Skype users that are online at the same time varies between 7 and 16 million (out of the over 400 million registered users).

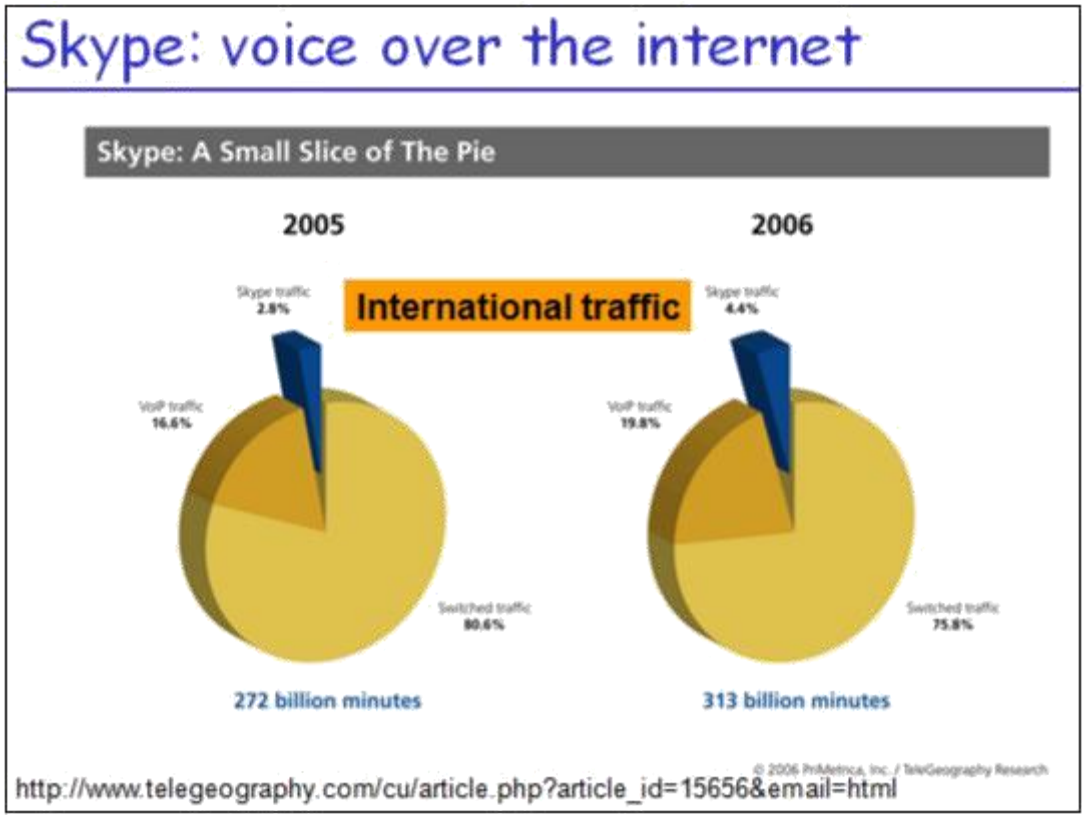
The past 5 years, the online user base has grown with approx. 8960 users/day or 3 million per year.

(See http://nyanyan.to/skype/40hr_chart.php for latest 2-day evolution, http://nyanyan.to/skype/7d_chart.php for last week)

Skype: voice over the internet



Sample week chart from http://nyanyan.to/skype/7d_chart.php, 4-10 feb. 2009



Dec 2006:

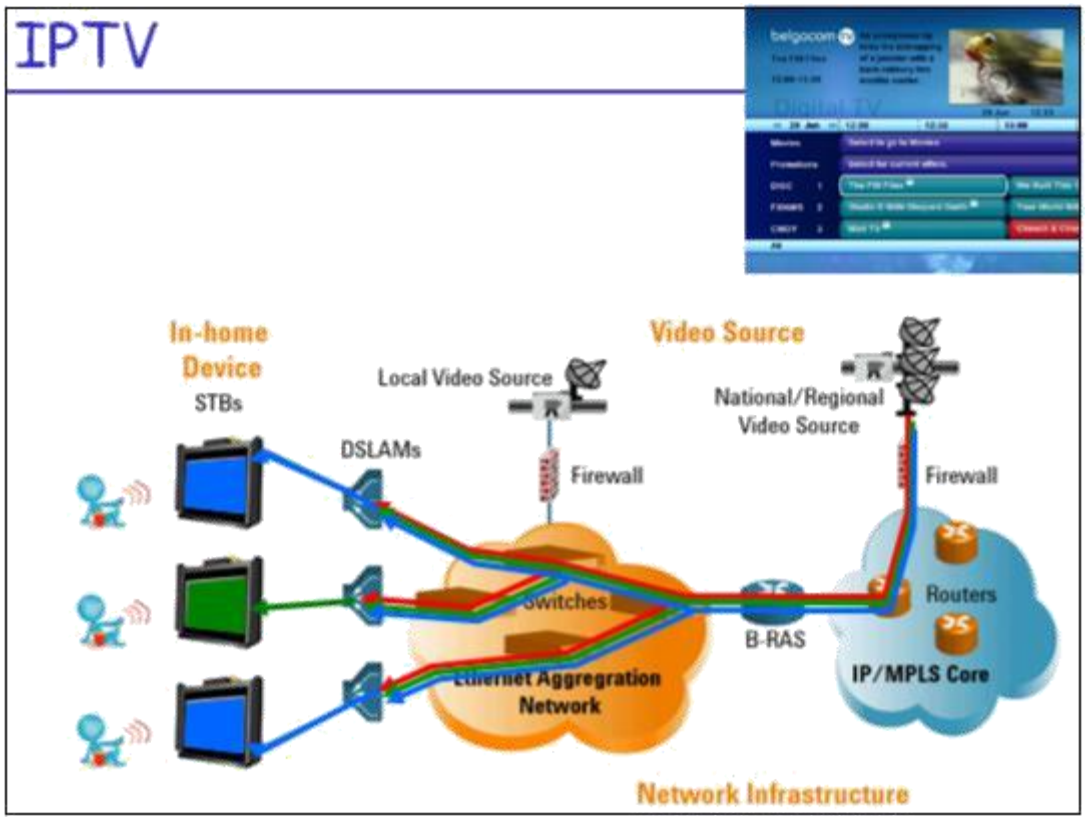
Computer-based Voice over IP (VoIP) is nothing new, but Skype is the first such service to break into the mainstream, attracting millions of users worldwide. Skype had a million simultaneous users within six months of the release of its first version for Windows in July 2004. By the end of the third quarter of 2006, it had 136 million registered users, and the number of users online now regularly exceeds eight million. These users generated about 6.6 billion minutes of traffic in the third quarter of 2006, and are on track to make over 27 billion minutes of PC-to-PC calls this year. About half of Skype's traffic is international.

This has prompted worries that Skype - and similar services - could undermine the viability of the international long-distance market. However, while the volume of international traffic routed via Skype is significant, the quantity is still small when compared to a global switched and VoIP traffic base of 264 billion minutes. Computer-to-computer traffic between Skype users in 2005 was equivalent to 2.9% of international carrier traffic in 2005 and approximately 4.4% of total international traffic in 2006. Furthermore, not all of Skype’s traffic is a net loss for international carriers. Skype also offers a paid 'Skype Out' service, which allows Skype users to place calls to traditional telephones. The service relies on wholesale international carriers, including iBasis, Cable & Wireless, and Level 3, to terminate this traffic to the telephone network.

Still, it’s clear that VoIP services will continue to gain in popularity. 'Someday, all calls will be routed over the Internet,' commented Stephan Beckert, Research Director at TeleGeography. 'But the numbers suggest that traditional international carriers aren’t going to disappear anytime soon.'

--

Aug. 2007, China: “IP calls accounted for 43% of lo ng distance calls last year” [http://voipguides.blogspot.com/2007/08/china-voip-growing-faster-than-pstn.html]



The figure shows a typical architecture used for the distribution of Digital Television over the Internet (IPTV).

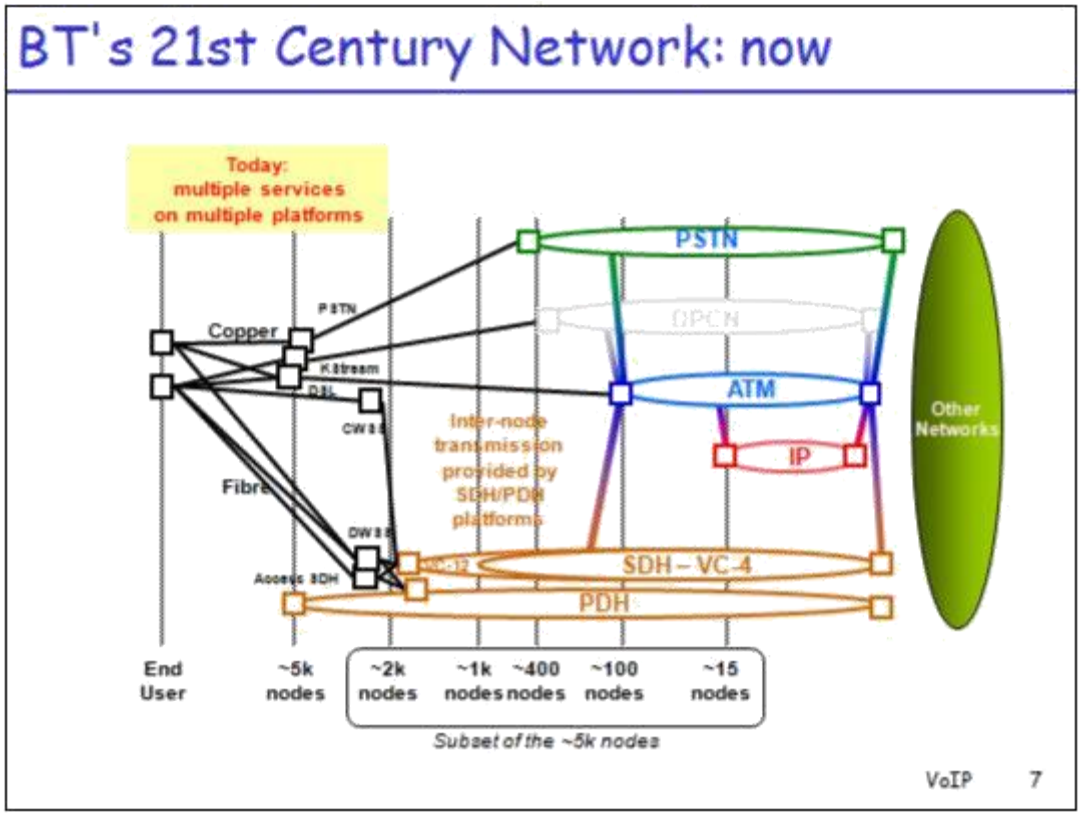
MPLS: Multi Protocol Label Switching

B-RAS: Broadband-Remote Access Server

DSLAM: Digital Subscriber Loop Access Multiplexer (used in e.g. ADSL or VDSL).

STB: Set-Top-Box

Dec.2007: Belgacom and Telenet have 305.300 resp. 391.000 digital TV subscribers.



This figure gives an impression of the current BT network. A lot of technologies are combined in order to offer telelephony, internet access and TV services to the end user.

PSTN: Public Switched Telephone Network

DPCN: Digital Private Circuit Network (= leased lines)

ATM: Asynchronous Transfer Mode

IP: Internet Protocol

SDH: Synchronous Digital Hierarchy

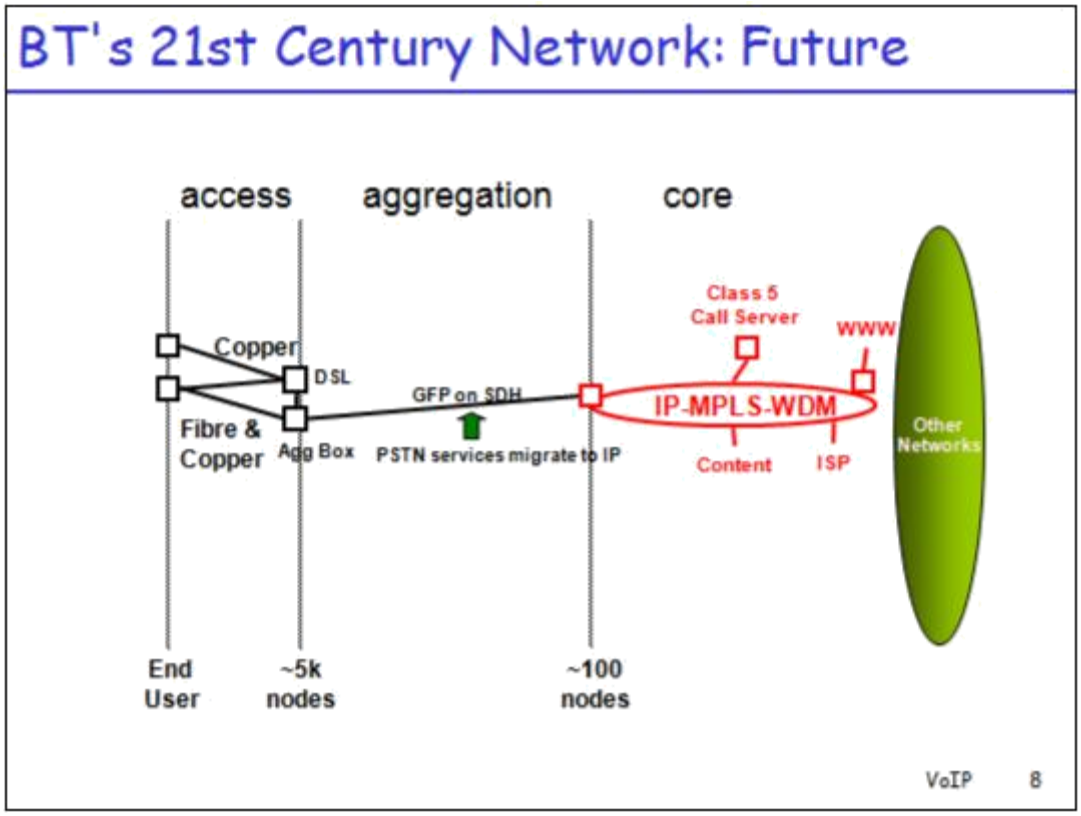
PDH: Plesiochronous Digital Hierarchy

DWSS: Digital Wideband Serving Section

CWSS: Customer Wideband Serving Section

VC-4: Virtual Container 4 (150 Mbit/s)

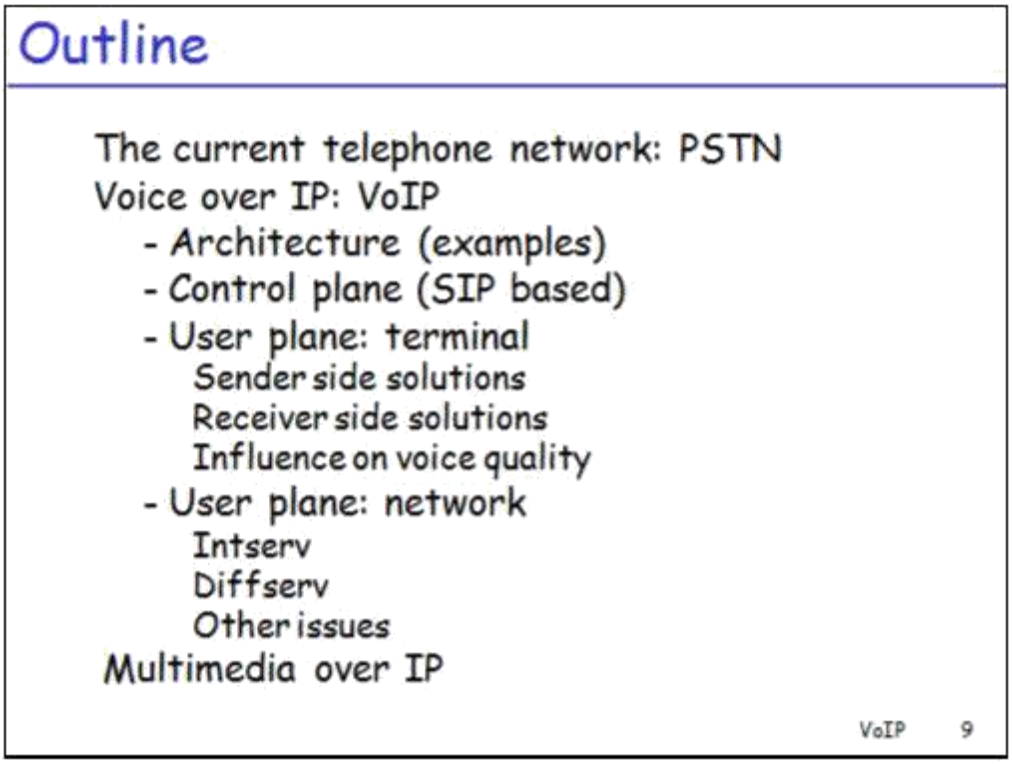
VC-12: Virtual Container 12 (2Mbit/s)



In this figure, the future BT internet is shown where a drastic convergence is observed. The core of the network is evolving towards an all IP based network (enhanced by MPLS and supported on a WDM transport network).

GFP: Generic Framing Procedure

MPLS: MultiProtocol Label Switching



The goal of this chapter is to provide an overview of how to provide voice signals (telephony) over an internet based infrastructure.

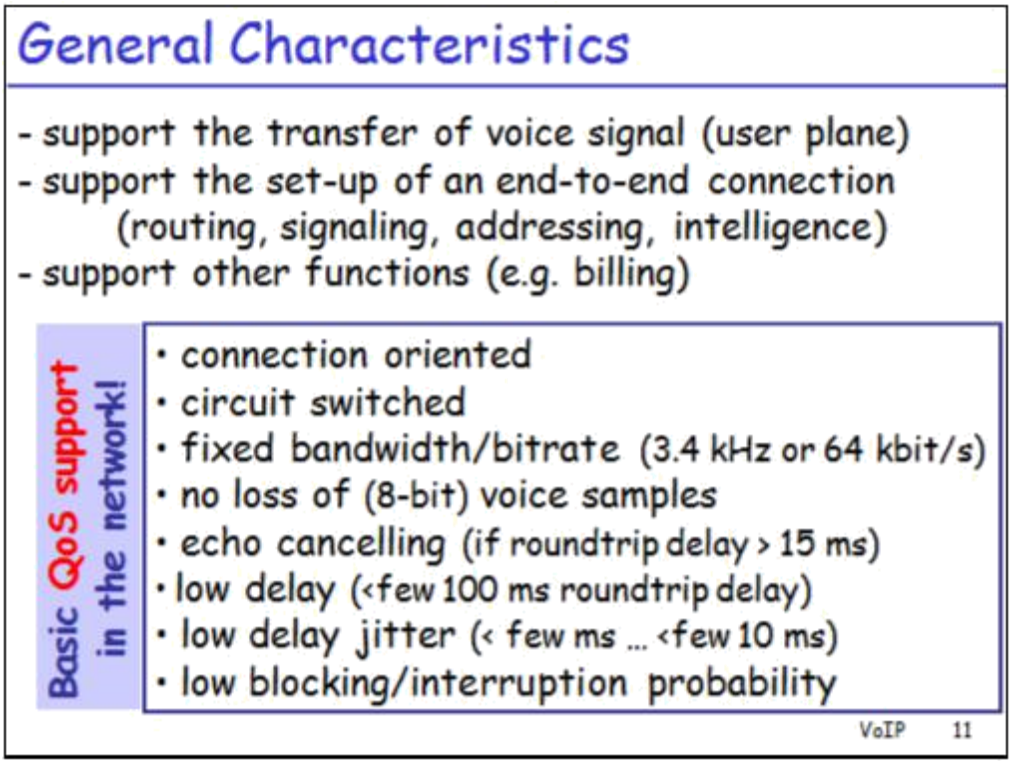
A first part will briefly recapitulate on the operation of the classical Public Switched Telephone Network (PSTN).

A second part (the major part) will focus on VoIP (Voice over IP). Both the control plane and the user plane will be addressed. The control plane will focus on the SIP (Session Initiation Protocol) as developed by the IETF. For the user plane, a distinction is made between the terminal and the network.

Finally a short expansion to Multimedia over IP is discussed.

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
4. Multimedia over IP

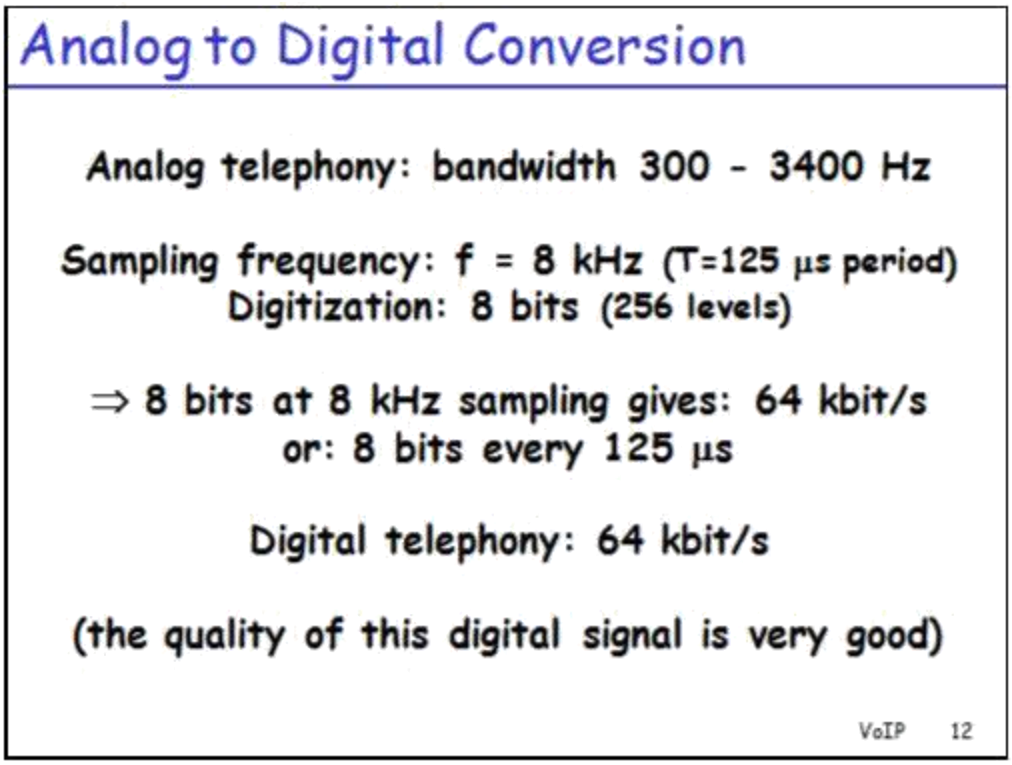


The general requirements of a telephone network are:

- support the transport of the voice signal
- support the set-up of the connection
- support other functions such as billing, management, etc.

Some basic features are:

- Connection oriented: before user data (= the telephone conversat ion) is exchanged one has to set up a call which will close the appropriate cross-points in the switches (or assign the correct time slots) resulting in an end-to-end circuit.
- The end-to-end circuit will support a constant bandwidth (3.4 kHz for analog telephony) or bitrate (64 kbit/s for digital telephony) during the call.
- Because the network is circuit switched, hardly any voice sample will be lost.
- The network will keep the delay as low as possible. Roundtrip delay values below 15 ms are required when trying to avoid echo. In general this will not be possible.
- Delay values below a few 100 ms are required to keep good interactivity d uring the call. This is a major problem when using satellites to interconnect phones (e.g. during trans-atlantic calls). A phone call via a GEO satellite results in a roundtrip delay of 500 ms.
- Delay jitter (=variation in the arrival time between differe nt voice samples) should be kept very low (typically below a few ms) because otherwise the voice quality is degrading very rapidly. In general this is not a real problem in telephone networks. Note that larger jitter is allowed when dejitter control is available (e.g. dejitter buffer)
- Blocking probability (due to network congestion) is in general very low.

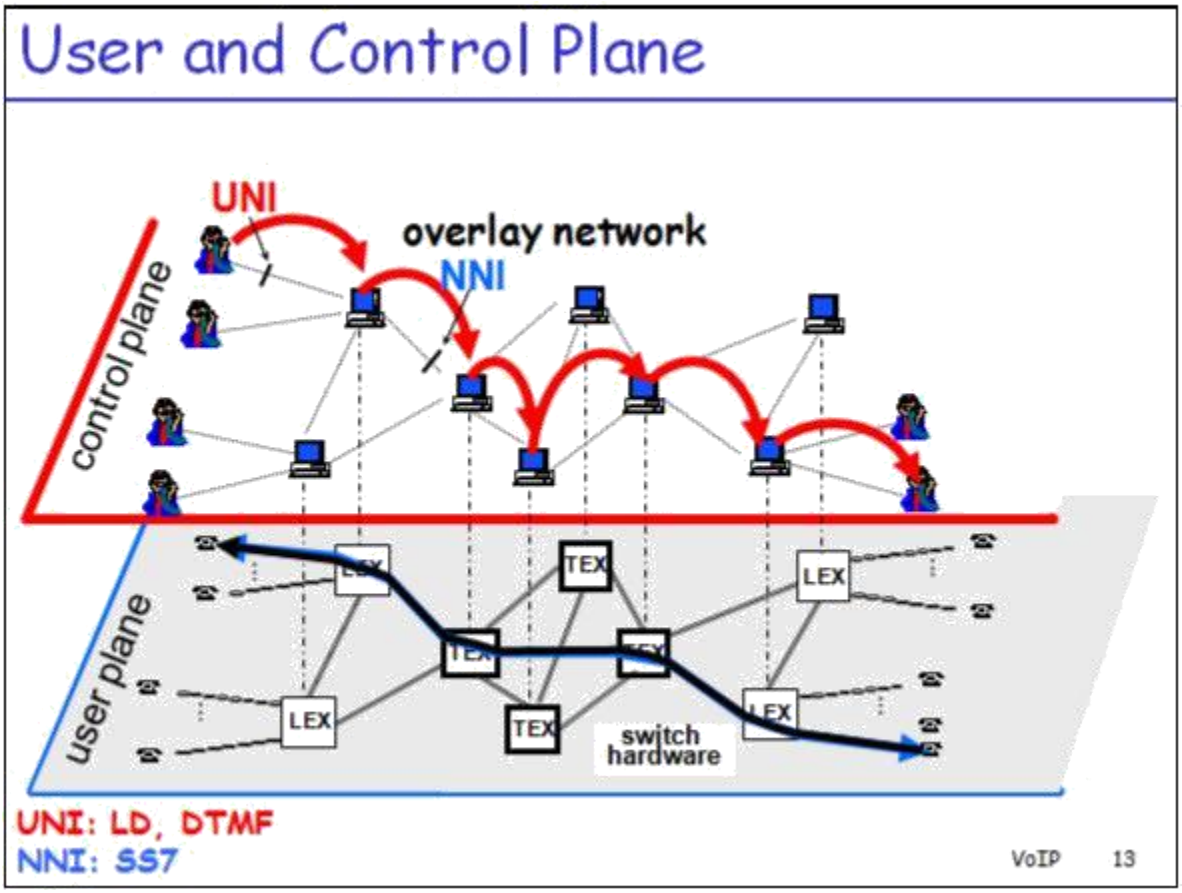


Today's telephone networks are fully digital inside the network and o nly the access to the network (from the home telephone to the local telephone switch) is still in most cases analog.

We consider an analog telephone signal which is converted to a digital signal: the analog voice telephone signal (300 - 3400 Hz) will be converted to a digital signal of 64000 bits/s.

We will first sample the signal in time. One has to sample the signal with a frequency which is at least two times the maximum frequency of the signal (cf. Nyquist-Shannon sampling theorem). For telephony this maximum frequency is 3400 Hz so we have to sample at least at 6800 Hz. In practice a sampling frequency of 8000 Hz is used (that is one sample every 125 μs). The next step is to digitize the amplitude of the signal. In telephony 8 bits (or 256 levels) per sample are used, since this suffices to obtain good voice quality. As a result 8 bits of information are generated every 125 μs, corresponding to 64000 bits every second: 64 kbit/s (= 8kHz x 8 bit).

(NB: CD is 2 channels x 44,100 samples per second per channel × 16 bits per sa mple = 1,411,200 b/s = 1,411.2 kb/s)



An important question still remains: how does the switching element know when it has to close? Each switch will use a computer (a controller) in order to close the appropriate switching elements (interface between user plane and control plane) but also in order to talk to other controllers and to the end user.

The use of the switch controllers and the signaling between a user and a switch controller or between the different switch controllers results in a kind of overlay network. This overlay network is not carrying the information the users want to exchange (=voice signal) but carries signaling messages required to control the network (to set up or tear down a call). Otherwise stated: during the call, the voice signal travels through the switch hardware, without entering the switch controller.

This overlay network is commonly described as the control plane and the network carrying the voice information is called the user plane (or data plane). This is more clearly shown in the figure.

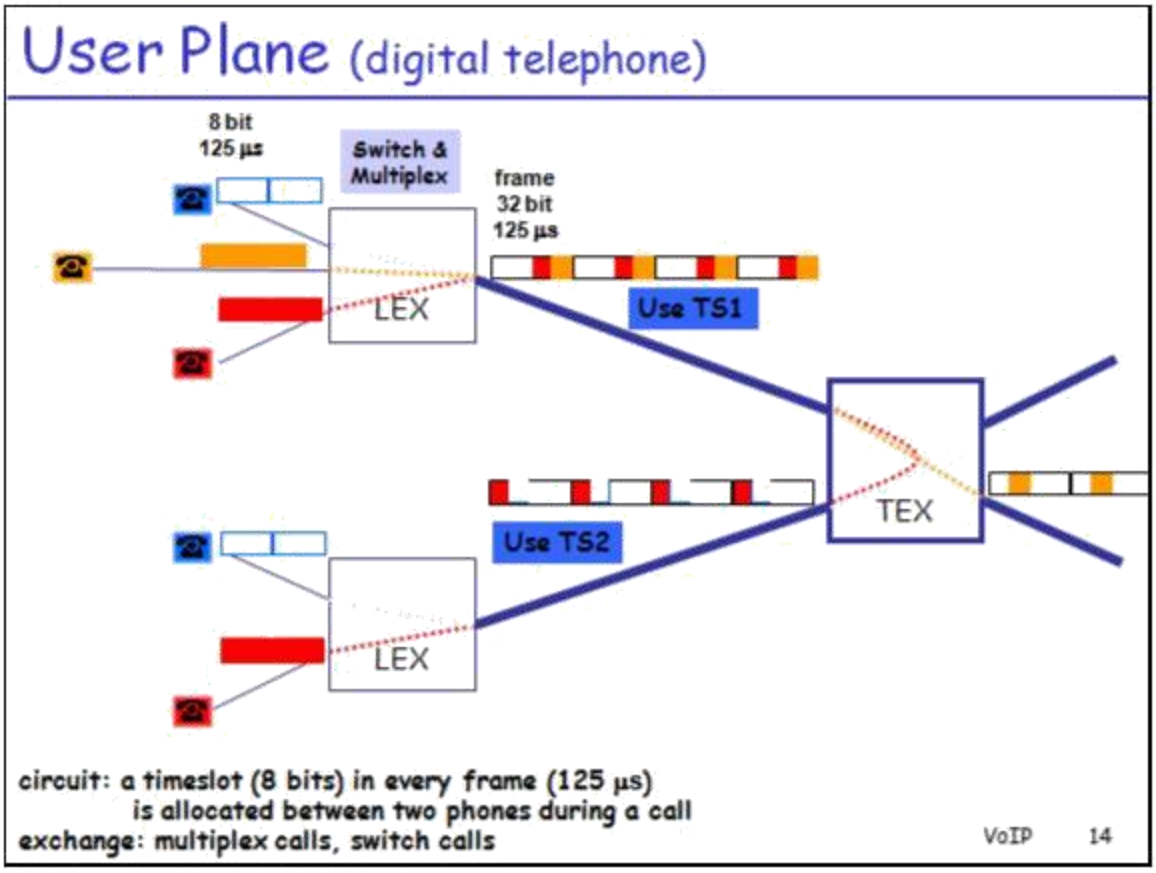
One can observe two types of interfaces in the control plane: one internal interface (NNI or **Network Node Interface** or **Network Network Interface**, between the switch controllers) and one external interface (UNI or **User Network Interface**, between the user terminal and the Local Exchange or LEX switch controller).

One of the important goals of the control plane is to support the call set-up. This call set-up will make use of signaling (both at the UNI and NNI interfaces) and routing. It will distribute the necessary information to know the switching elements that should be closed in order to set- up a phone call.

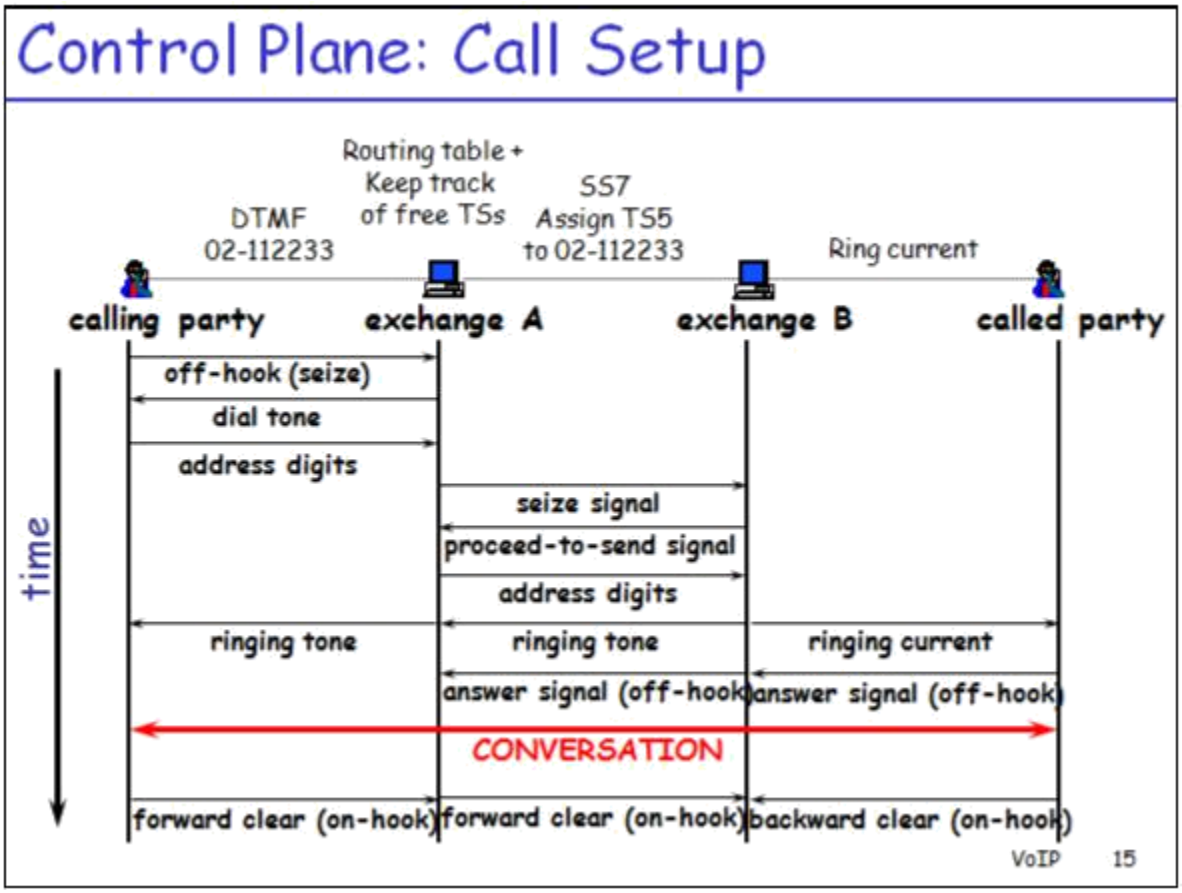
Note: as indicated in the animation, a one-way speech channel in the upstream direction is gradually built up. This is important to carry the ringing tone from the called party to the caller (in-band signaling).

LD: Loop Disconnect

DTMF: Dual Tome Multi-Frequency

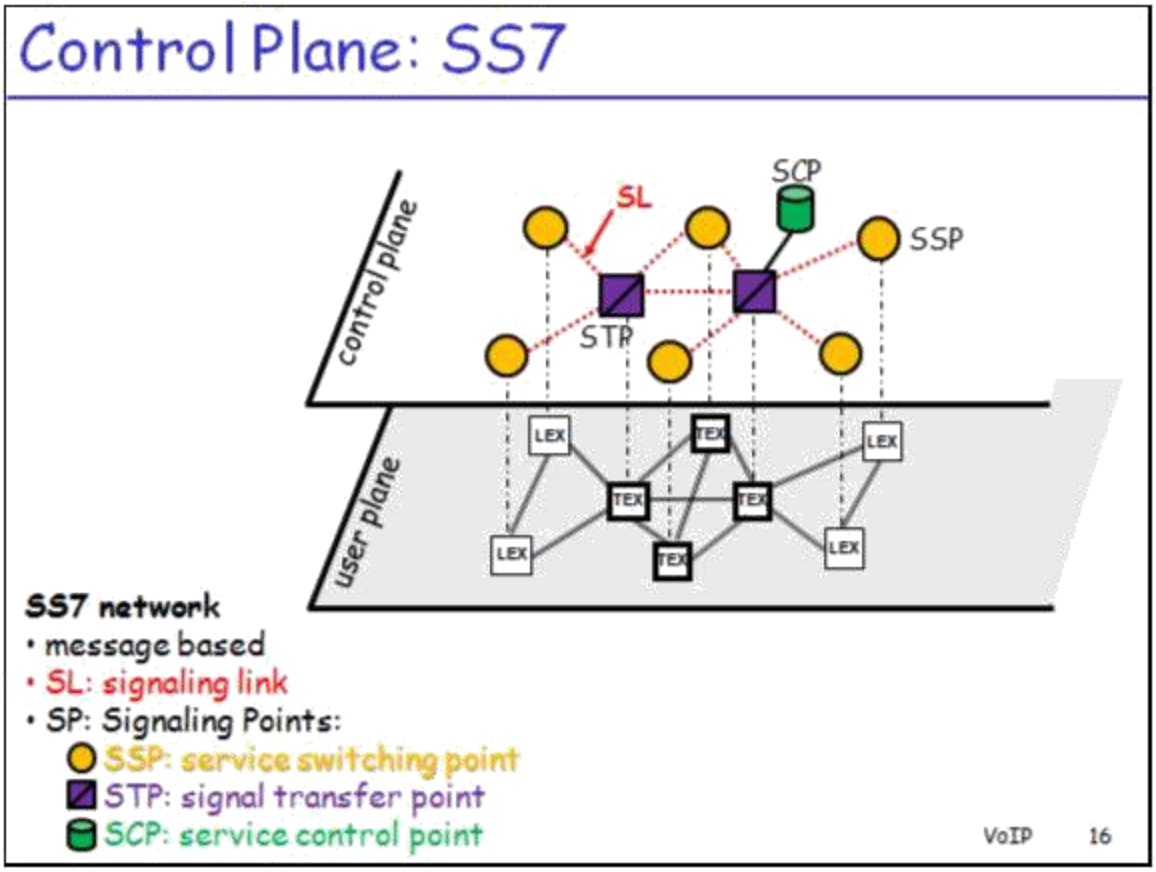


This figure gives a summary of the user plane operation (fully digital network).
During a phone call, a telephone is generating samples of 8 bits every 0.125 ms. This continues until the call has ended. These voice samples will be transmitted over the twisted pair access network towards the local exchange (LEX). In the LEX, a number of these samples from different phones will be multiplexed (in our fictitious example multiplexing in a 4 timeslot frame). In the transit exchange, these calls are switched. We observe that each frame will transport one voice sample from a sending telephone to a receiving telephone (and of course also the other way around, not shown in the figure). The same timeslots will be reused to transport different phone calls (not overlapping in time). Phone calls that overlap in time and use the same line systems will use different timeslots. If no timeslots are available anymore one observes a call blocking.



In order to better understand the control of the network, it is interesting to analyse a call set-up procedure. The different call set-up steps are:

- The caller lifts the handset (off-hook) which alerts the exchange that he wants to place a call.
- The exchange A will identify the subscriber and prepare itself to receive the digits.
- The exchange A will then provide a dial tone to the caller.
- Now the caller can dial the digits of the phone number he wants to reach. This is done using DTMF (Dual Tone Multi Frequency) signaling.
- The exchange A will wait till all the digits are dialed and then look up its routing information to see if the call is local (we consider here a non-local call via two exchanges).
- The exchange A will send a seize signal to the exchange B and sends the digits (after receiving a “proceed-to-send” signal).
- The exchange B will now test if the called phone is busy. If not, the exchange B will send the ringing current to the called subscriber and the ringing tone to the caller (via exchange A).
- If the called subscriber picks up the phone, the conversatio n can start.
- The connection will be released if the call is finished.



The control plane is an overlay network that is typically using links of 64 kbit/s (or multiples of 64 kbit/s). This network consists of **Signaling Points** or SPs interconnected by **Signaling Links** or SL's (over which the signaling messages can be sent). There are three types of SP's:

Service Switching Point (SSP) where signaling messages can originate or terminate, typically in the local telephone switches or local exchanges

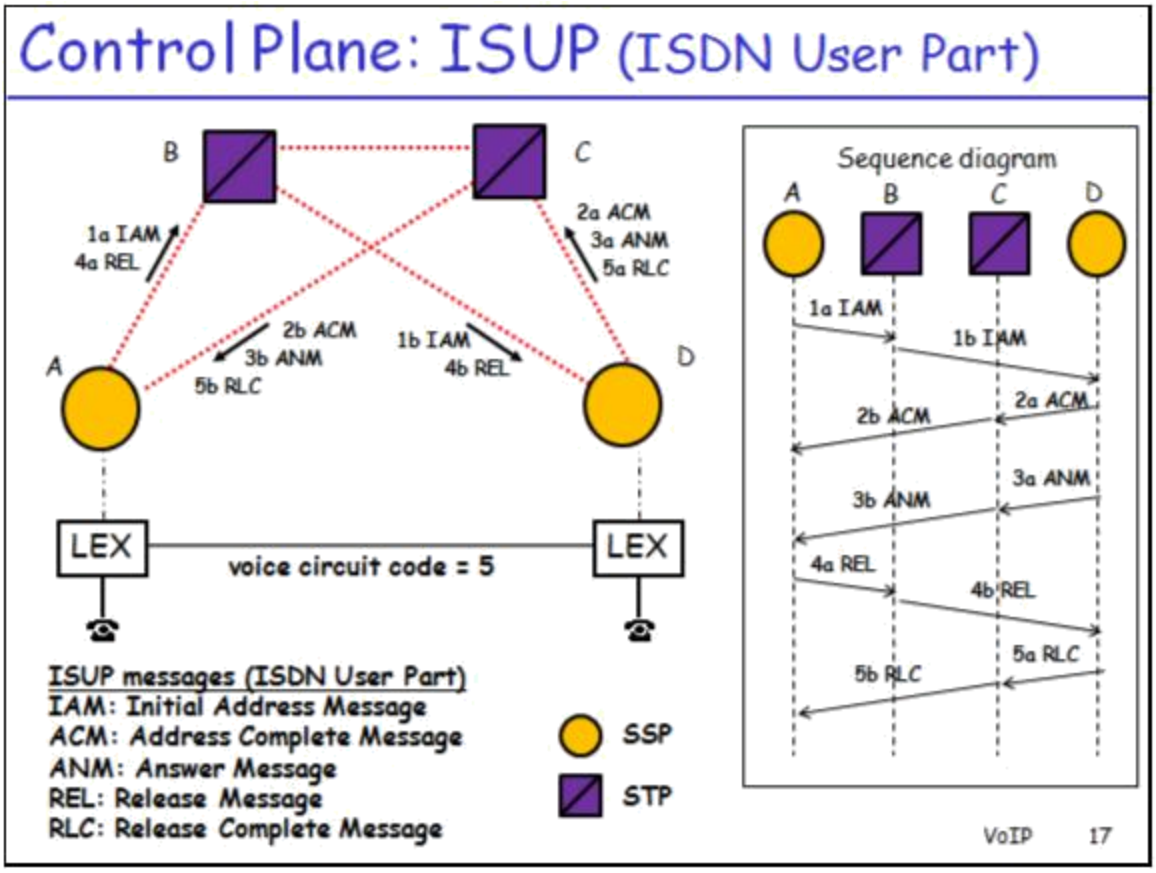
Signaling Transfer Points (STP) where signaling messages can be routed (note that STP and SSP may be combined at a single exchange)

Signaling Control Points (SCP) where intelligence is provided how to handle the calls, stored in databases (e.g. for local number portability, LNP).

This signaling network is the **SS7 network** (Signaling Systems No 7, standardized by the International Telecommunication Union or ITU).

The signaling network is message based (packet based). The signaling link layout does not have to correspond to the physical links (e.g. on the figure there are less signaling links than physical links). Normally a switch will have a service switch point (SSP), there will be fewer signaling transfer points (STP).

A good overview is provided at: www.pt.com/tutorials/ss7/index.html



When a call is placed to an out-of-switch number (=non local number), the originating SSP transmits an ISUP initia l address message (IAM) to reserve an idle trunk circuit from the originating switch to the destination switch (1a). The IAM includes the originating point code, destination point code, circuit identification code (circuit “5” in the example), dialed digits and, optionally, the calling party number and name. In the example, the IAM is routed via the home STP of the originating switch to the destination switch (1b). Note that the same signaling link(s) are used for the duration of the call unless a link fa ilure condition forces a switch to use an alternate signaling link.

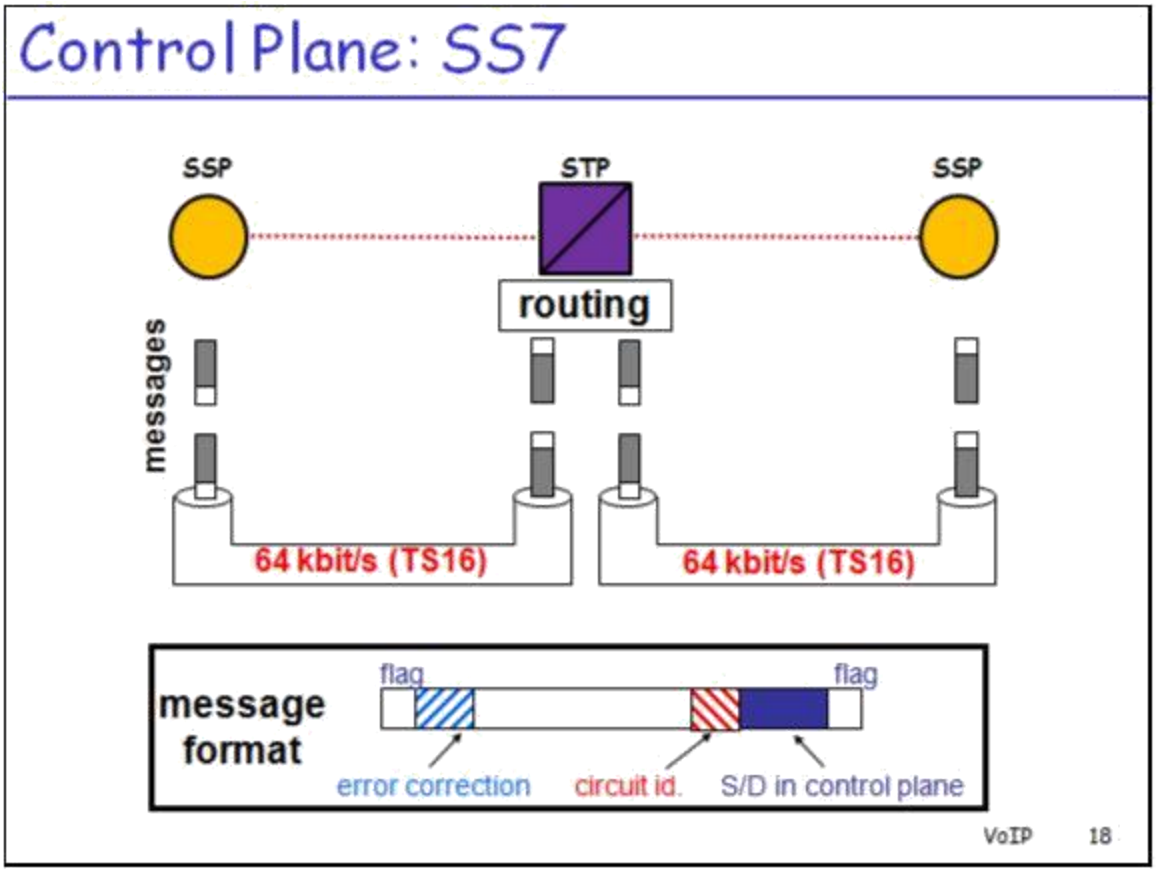
The destination switch examines the dialed number, determines that it serves the called party, and that the line is available for ringing. The destination switch rings the called party line and transmits an ISUP address complete message (ACM) to the originating switch (2a) (via its home STP) to indicate that the remote end of the trunk circuit has been reserved. The STP routes the ACM to the originating switch (2b), then the terminating switch provides power ringing to the called party and audible ringing tone to the calling party.

In the example shown above, the originating and destination sw itches are directly connected with trunks. If the originating and destination switches are not directly connected with trunks, the originating switch transmits an IAM to reserve a trunk circuit to an intermediate switch. The intermediate switch sends an ACM to acknowledge the circuit reservation request and then transmits an IAM to reserve a trunk circuit to another switch. This processes continues until all trunks required to complete the voice circuit from the originating switch to the destination switch are reserved.

When the called party picks up the phone, the destination switch terminates the ringing tone and transmits an ISUP answer message (ANM) to the originating switch via its home STP (3a). The STP routes the ANM to the originating switch (3b) which verifies that the calling party's line is connected to the reserved trunk and, if so, initiates billing.

If the calling party hangs-up first, the originating switch sends an ISUP release message (REL) to release the trunk circuit between the switches (4a). The STP routes the REL to the destination sw itch (4b). If the called party hangs up first, or if the line is busy, the destination switch sends an REL to the originating switch indicating the release cause (e.g., normal release or busy).

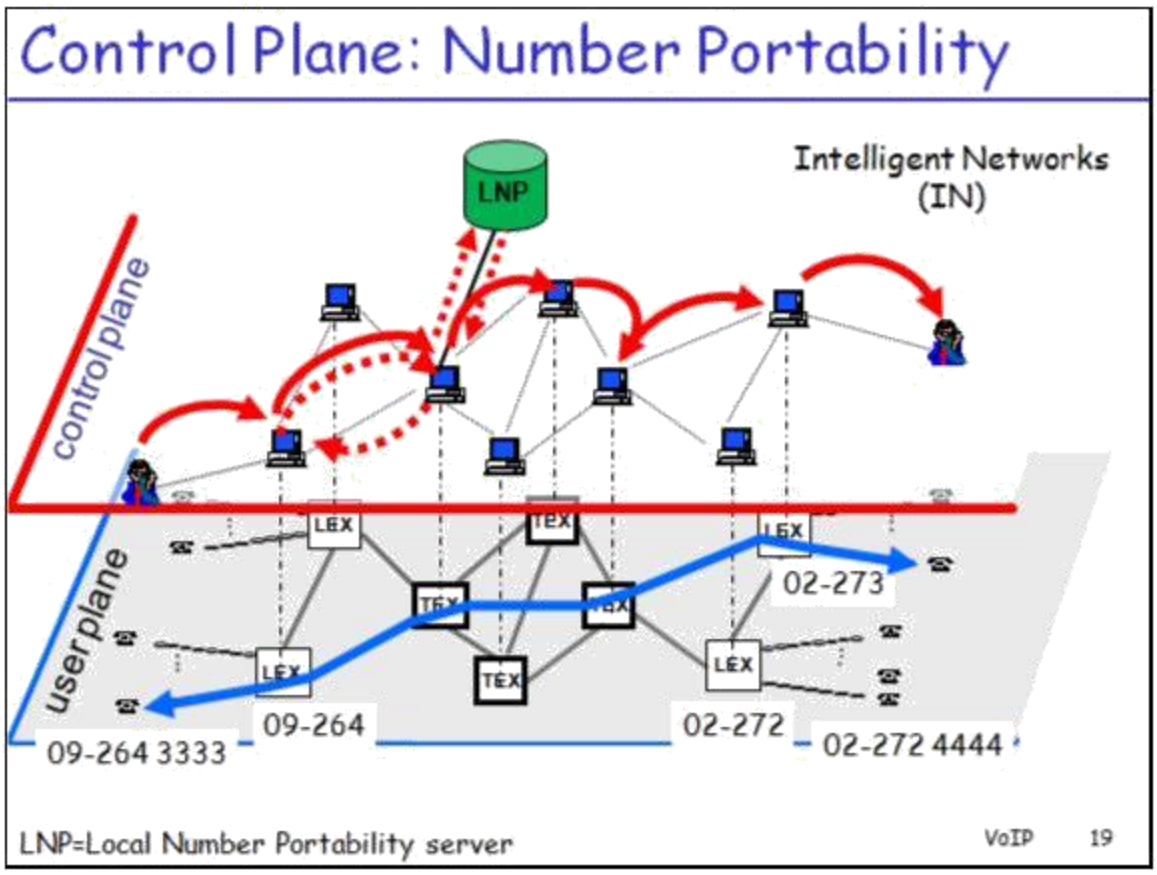
Upon receiving the REL, the destination switch disconnects the trunk from the called party's line, sets the trunk state to idle , and transmits an ISUP release complete message (RLC) to the originating switch (5a) to acknowledge the release of the remote end of the trunk circuit. When the originating switch receives (or generates) the RLC (5b), it terminates the billing cycle and sets the trunk state to idle in preparation for the next call.



An example of the transfer of information in the control plane is shown in the figure. The information to be transferred (e.g. telephone number) is put in messages which are transported over 64 kbit/s channels. The STP's have routing tables which will forward the message to the correct SL (signaling link). *Note that the messages may follow routes completely different from the final call route!*

Typical parts in the message are:

- error correction information
- S/D: originating point code and destination point code ("source" and "destination"). This is information used to route the message in the control plane
- For ISUP messages:
    - Circuit identification code (CIC): identifies the (voice) trunk circuit reserved by the originating switch to carry the call.



Number portability is the possibility to keep your telephone number when physically moving or when changing operator. In the example we show the operation of LNP (Local Number Portability) when moving to a new location. Note that only the principle is explained, the exact details are much more complex and out of scope for this course.

In the example, telephone 09-264 3333 is calling 02-272 4444.

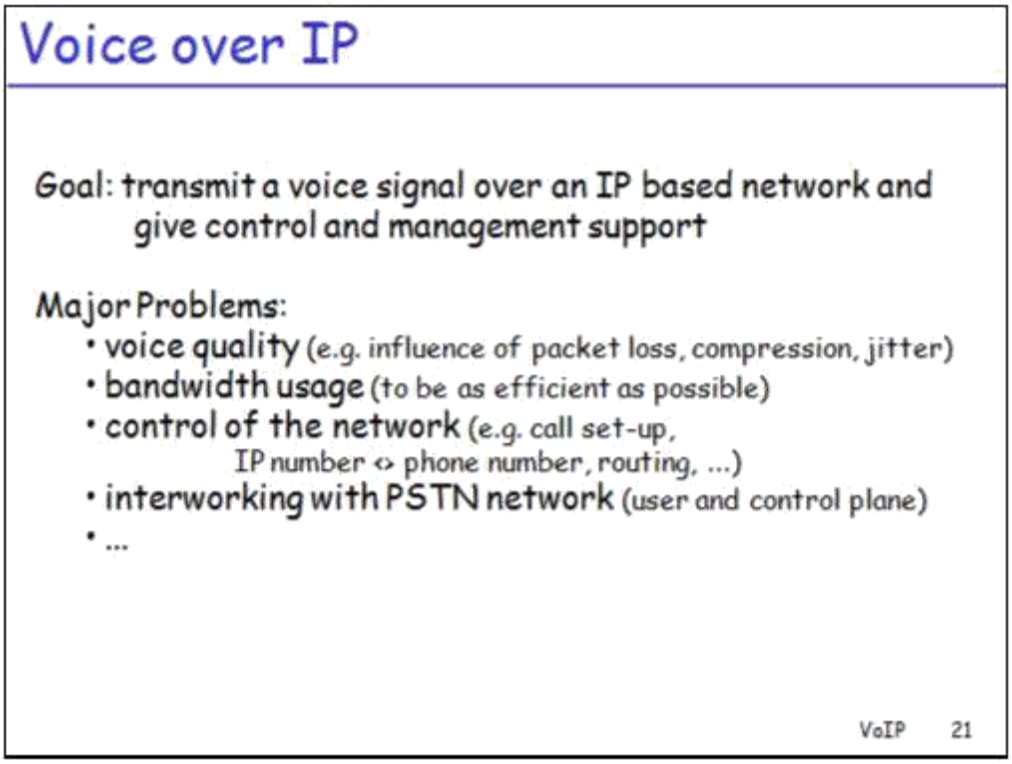
First consider the case where no number portability is used. In this case a simple routing will happen based on the called telephone number. The message in SS7 will use as source 09-264 (the number of the originating LEX) and as destination 02-272 (the number of the destination LEX). The intermediate STP's will have routing tables that will forward the message to the correct destination. Inside the message, the called phone number will be transported (02-272 4444).

Consider now the case where LNP is used. In this case we need a LNP server (typically inside a SCP or Signaling Control Point) and also the routing tables in the exchanges will be adapted. When the calling party (09-264 3333) will transfer to its LEX the destination phone number (using DTMF signaling), the LEX will observe in its local routing data base that the number is a ported number (actually it will be indicated that the whole range 02-272 xxxx is ported, although maybe only one number was ported). As a result an SS7 message will be sent to an intelligent server (LNP server) where it is known that the number 02-272 4444 is ported to the exchange 02-273. Upon receiving this information, the 09-264 LEX will then set-up the call using ISUP, with the corresponding control plane addresses: as source 09-264 (the number of the originating LEX) and as destination 02-27**3** (the number of the new destination LEX). The intermediate STP's will have routing tables that will forward the message to the correct destination (02-273!). Inside the message, the originally called phone number will be transported (02-27**2** 4444). The LEX 02-273 will connect to the phone 02-272 444 (ringing, etc.)

LRN: Location Routing Number = actual destination

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
4. Multimedia over IP



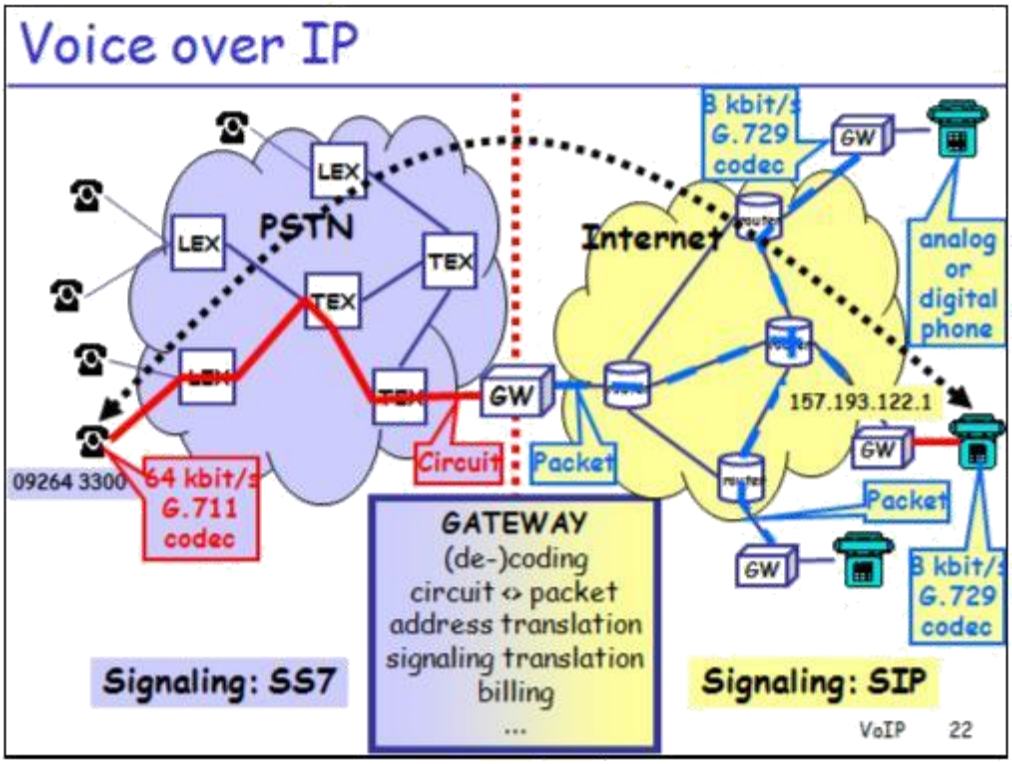
When considering the use of internet for the support of telephony, one often refers to Voice over IP (VoIP).

The goal of VoIP is to transmit a voice signal over internet with an acceptable quality. As it is clear from the previous paragraph on PSTN, there are many more functions (control and management) that should be supported in order to have a fully operational telephone service on an internet.

Some of the major problems are:

- Obtain good voice quality (taking into account the specific problems encountered on internet, such as: delay, jitter, packet loss, connectionless network). This will require the development of a QoS (Quality of Service) enabled internet.
- Use the bandwidth as efficient as possible in order to allow as many us ers as possible on the network and in order to support also the traditional internet services (such as e-mail, FTP, web access, …). This will require the use of a dvanced voice coders and decoders.
- Provide a number of specific control functions in the network (call set-up, resolution between phone number and IP address, routing, etc.). This will require a number of servers supporting specific control services.
- Provide interworking with the traditional PSTN network (both at user and control plane level). This will require gateways in order to obtain interworking at control and user plane.

The study of Voice over IP will be an ideal vehicle to introduce a number of basic concepts and technologies used in internet to support multimedia services.



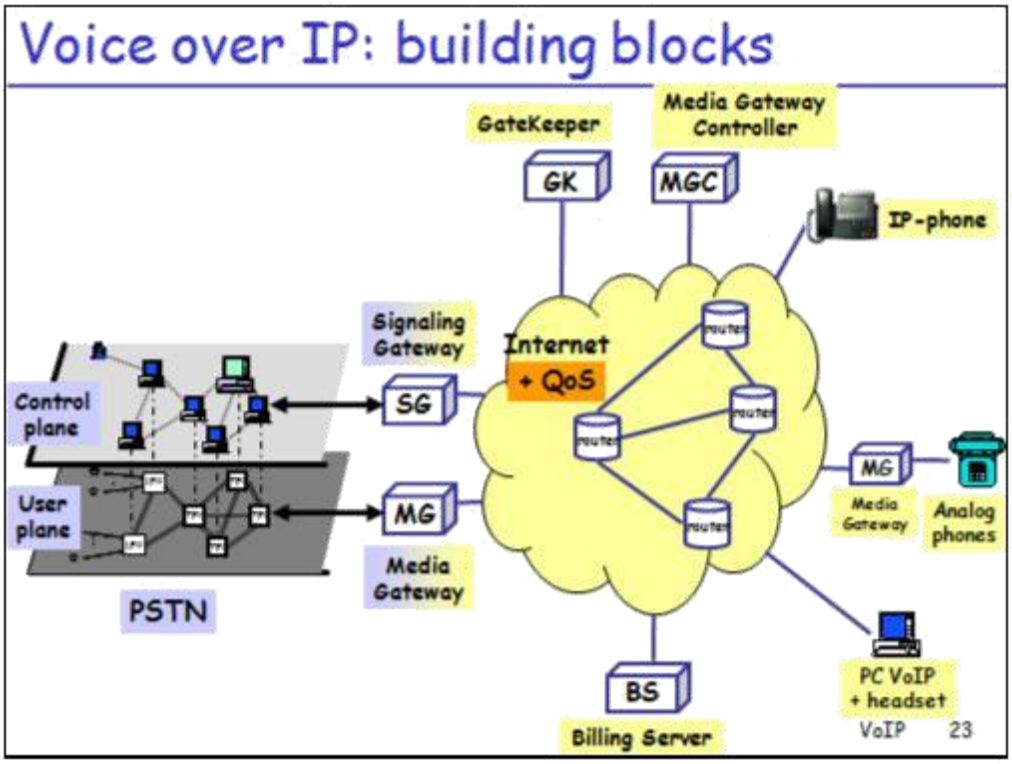
The goal of Voice over IP is to transport a voice telephone signal over a packet switched network. This will require a number of functions shown on the figure:

• Phone connected to internet: a coder (e.g. G.729) will generate a strea m of voice samples that will be transported in IP packets. At the receiving end the opposite conversion will be done using a decoder. Because a coder and decoder are always used together, one talks about a codec (CODEC = COder/DECoder).

• Phone connected to PSTN : the voice samples (from G.711 codec) will be transported in a circuit (64 kbit/s).

• A gateway will be provided in order to support phone calls between users in a PSTN and Internet. This gateway should support transcoding (e.g. between G.711 in PSTN and G.729 in Internet), conversion between time-slots and packets, address translation (telephone number <> IP address), signaling translation (PSTN SS7 <> IP SIP), exchange of billing information, etc.

Note: G.711, G.729, … are codecs that are standardized (the number refers to t he standard from ITU)

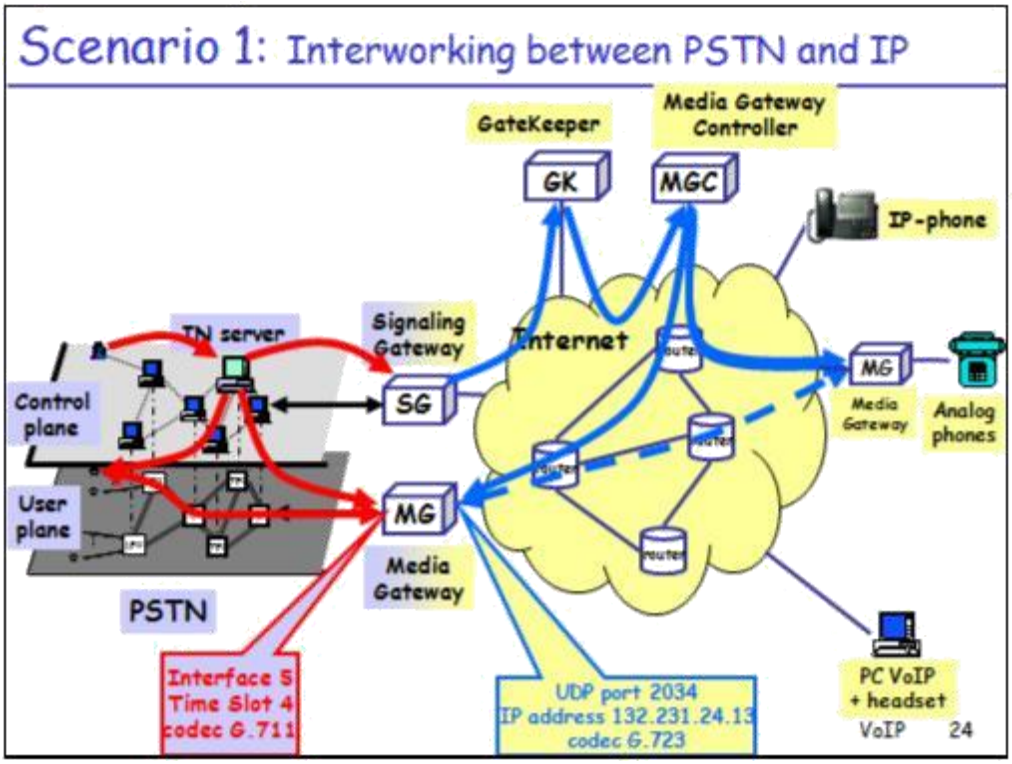


Some basic building blocks of a VoIP network are:

- Terminals: Several options are possible: PC with VoIP software and headset or loudspeaker/microphone, PC with VoIP software and interface card to normal telephone, media gateway connected to a normal phone, IP phone (which can be directly connected to internet with e.g. an Ethernet interface).
- Media Gateway (MG): gateway between PSTN and internet for the user plane information (= the voice signal). Is also used at the terminal side.
- Signaling Gateway (SG or SGW): responsible for the interworking of the signaling systems in PSTN and internet.
- Media Gateway Controller (MGC): responsible for the set- up of the connection inside internet. This gateway will know the addresses of the different phone subscribers, it will coordinate the call set-up , etc.
- Gatekeeper (GK): additional functionality, for example user aut hentication and access control.
- Billing Server (BS): server to support billing of the customers.
- QoS internet: extension to the current internet protocols in order to provide the necessary quality of service for voice (e.g. low delay, low jitter, low packet loss).

Note: This architecture corresponds to the MEGACO standard (Media Gateway Control protocol). This is both an IETF (RFC2885) and an ITU (H.248) standard (both date from the year 2000). Later in this chapter we will use SIP terminology.

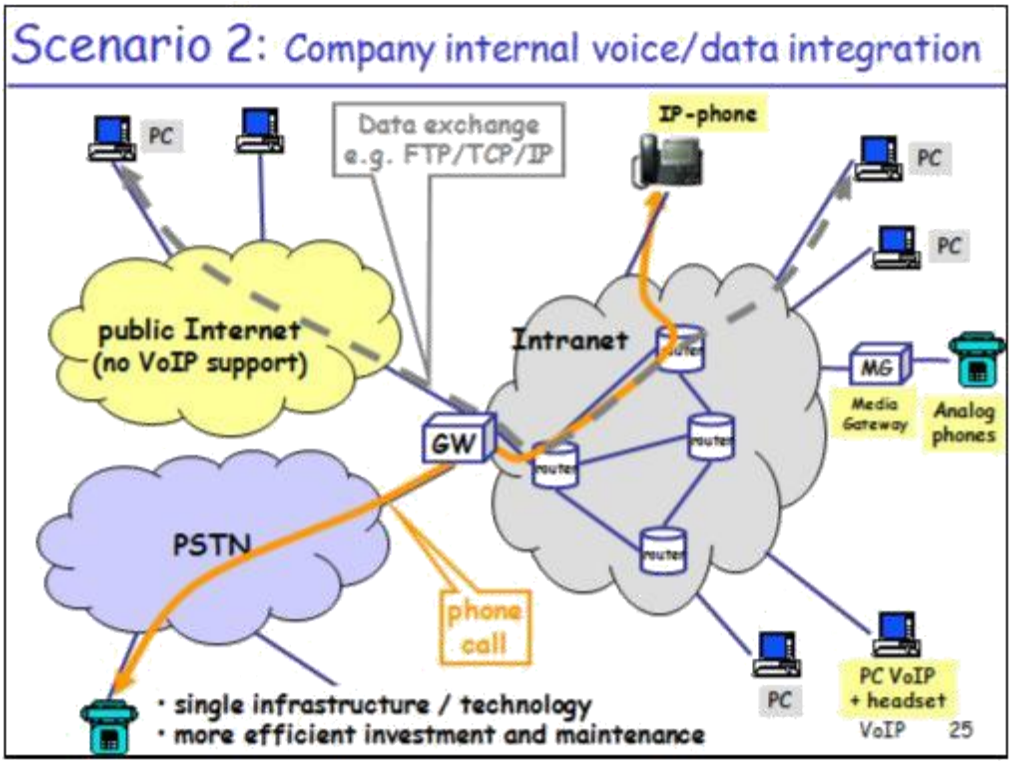
Note: SIP vs MEGACO: http://www.sipcenter.com/sip.nsf/html/SIP+and+MGCP+Megaco



The figure illustrates a first example: a simplified call set- up scenario where users in a PSTN network and an IP network are setting up a phone call.

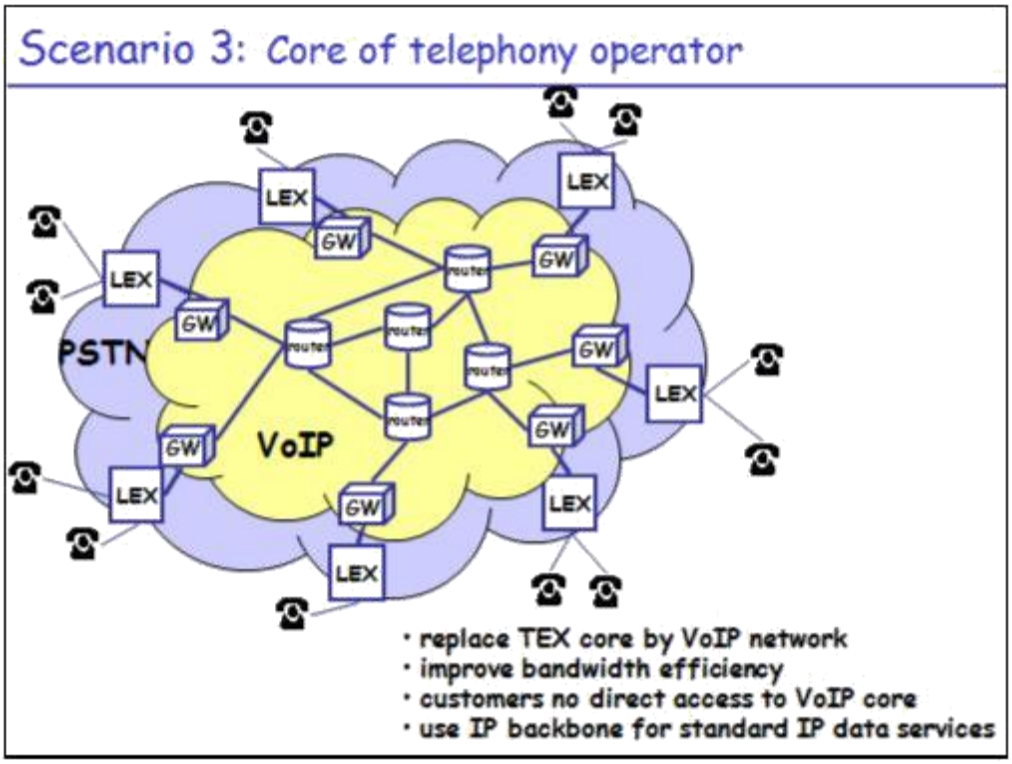
A user in the PSTN (PSTN-user) wants to call a user in internet (IP-user). The PSTN-user will send the phone number of the IP-user to his IN-server (IN = Intelligent Network). This IN-server knows that the person he wants to reach is an internet user and he will forward the request to the SG (Signaling Gateway). This SG will eventually check if the PSTN user is allowed on the IP network and will forward the request to the MGC (Media Gateway Controller). This MGC will contact the IP-user and his phone will ring. When he picks up the phone, the end-to-end connection can be set up (signals are exchanged in the other direction, not shown on the figure).

The identification of the two interfaces at the MG is indicated: at the PSTN side this is specified with a physical interface number, a timeslot number (in the TDM frame) and a codec type (G.711); at the IP side one observes a UDP port number, an IP address and a codec number (G.723). The MG will map the timeslot information (from PSTN side) in the packets (at IP side) and vice-versa and this after transcoding between the G.711 and G.723 codecs.



A second example of the use of VoIP is in an intranet environment. A company is using its intranet (=internal internet) to support both data and telephony traffic. This has important advantages in terms of infrastructure, maintenance, investment, etc. (especially in green-field situations where new infrastructure has to be deployed).

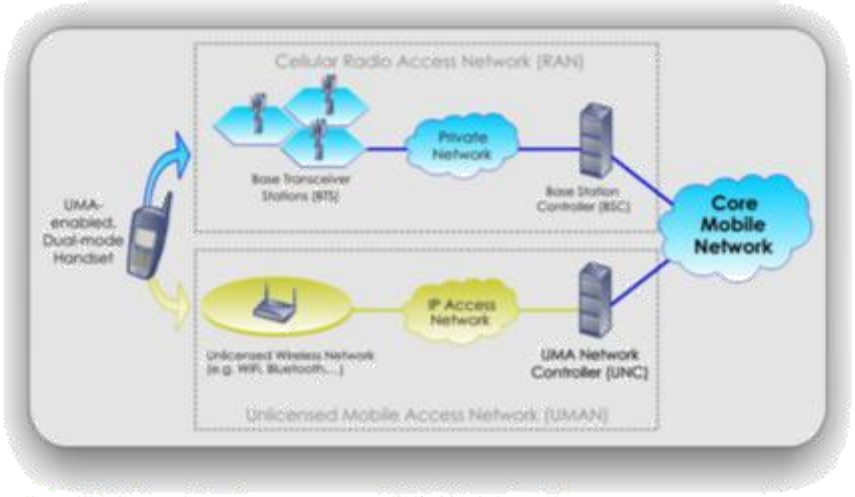
In order to connect to the outside world, a gateway is used which will direct all data traffic to the public Internet and the voice traffic to the PSTN (no VoIP in the public Internet).



A third example is using VoIP in the core of the public telephone network in order to (partially) replace the transit exchanges. Today we observe many incumbent telephony operators also offering IP services to their customers. In order to support these services, they also deploy a backbone IP network. They are considering to use this backbone IP network also for the support of telephony traffic using VoIP. In this case the VoIP is only used between normal telephone exchanges and no customers have direct access to it. In this way they are able to improve bandwidth usage and to further integrate their network, probably ending up with a single core network technology based on IP.

Note that there are other possible scenarios for the use of VoIP. One example are ITSP's or Internet Telephony Service Providers. These are telepho ny service providers who roll out a complete internet infrastructure, only for the support of voice traffic. They give direct access to their network for customers (e.g. business customers).

Recent Trends



- UMA: Unlicensed Mobile Access
- E.g. BT Fusion: Use of dual mode handsets (e.g. Nokia 6136) with GSM and IEEE802.11 interface

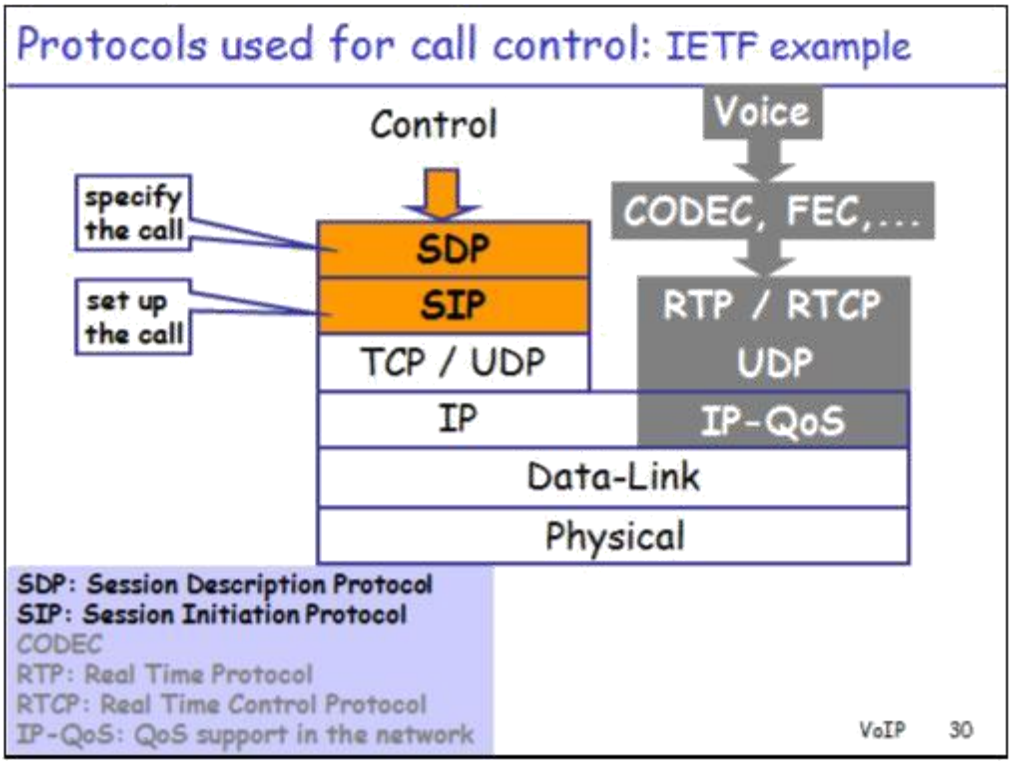
Recently we observe the use of VoIP to give access to mobile services in a home environment using a WLAN (IEEE802.11) access point.

Outside the home, normal GSM access is used, inside the home one switches to a local WLAN access point that is connected to a DSL access line. The connection to the UMA Network Controller is based on an IPSec tunnel.

Interesting links: http://www.umatechnology.org/index.htm, http://www.btfusionorder.bt.com/default.aspx

Content

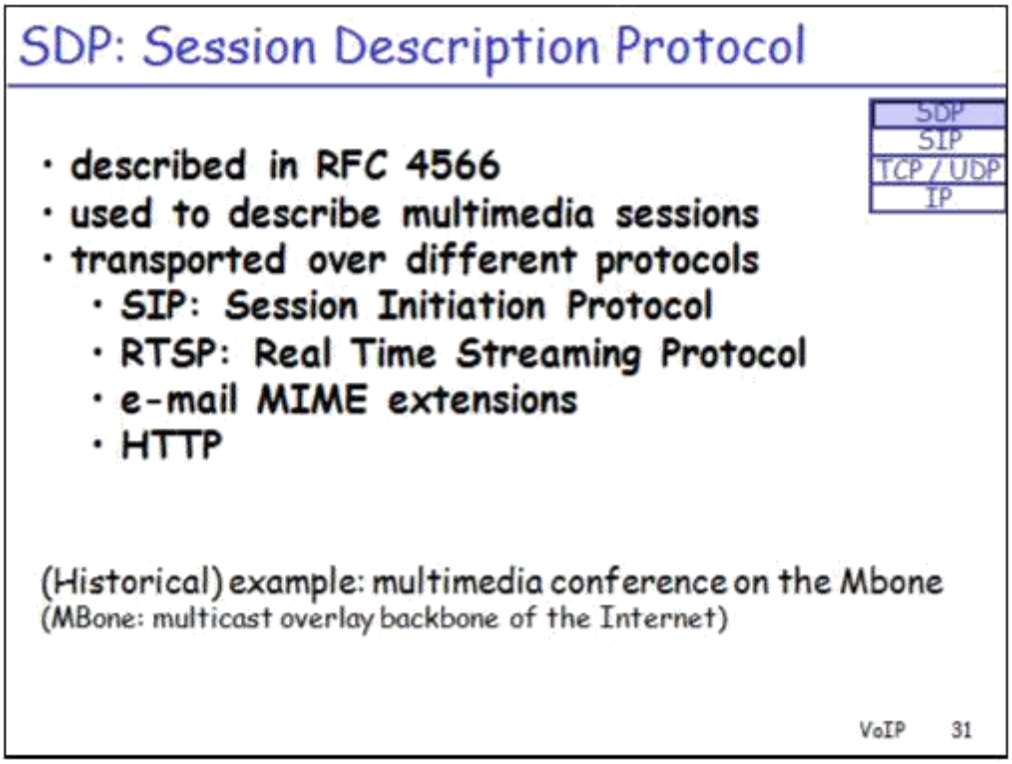
1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
     - SDP (Session Description Protocol)
     - SIP (Session Initiation Protocol)
   - User plane (terminal and network)
4. Multimedia over IP



This part will focus on an example architecture for the control of a VoIP network. The figure illustrates the protocols used for the control of the VoIP network and the protocols used for the voice transport itself (control plane and user plane)

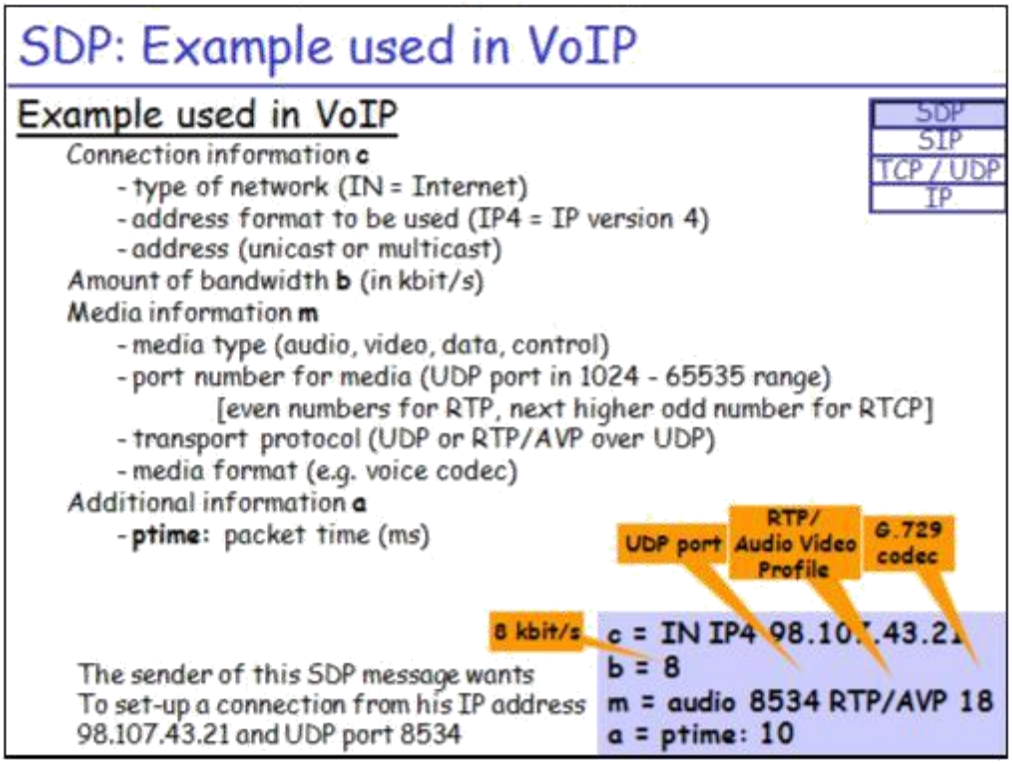
At the top of the control protocol stack, SDP is used (Session Description Protocol). This will specify the session (for VoIP we talk about a call) parameters (it is not really a protocol but a specification template). Below SDP the Session Initiation Protocol (SIP) will be used to set up the call (using the SDP to specify the call parameters). SIP is running on the classical TCP/UDP, IP, data link, physical layer protocol stack.

The voice protocol stack will use RTP/RTCP to support the transport of voice samples over internet. UDP is the preferred transport layer. In order to obtain the necessary quality for the voice signal, measures have to be taken in the network itself. Special QoS enabling technologies/protocols will be used (not shown here).



First some background: Mbone is a part of the Internet that supports IP multicast (see later), and thus permits efficient many-to- many communication. It has been used extensively for multimedia conferencing. Such conferences usually have the property that tight coordination of conference membership is not necessary; to receive a conference, a user at an Mbone site only has to know the conference's multicast group address and the UDP ports for the conference data streams. Session directories assist the advertisement of conference sessions and communicate the relevant conference setup information to prospective participants. SDP is designed to convey such information to recipients. SDP is purely a format for session description – it does not incorporate a transport protocol, and is intended to use different transport protocols as appropriate, including the Session Initiation Protocol or SIP, the Real Time Streaming Protocol (RTSP), electronic mail using the MIME extensions, and the Hypertext Transport Protocol (HTTP). SDP is intended to be general purpose so that it can be used for a wider range of network environments and applications than just multicast session directories.

The Session Description Protocol (SDP) is used to describe various parameters of a session. A session is very general (e.g. multimedia conference), but in this chapter it is used for a point-to-point voice over IP session (a VoIP call).

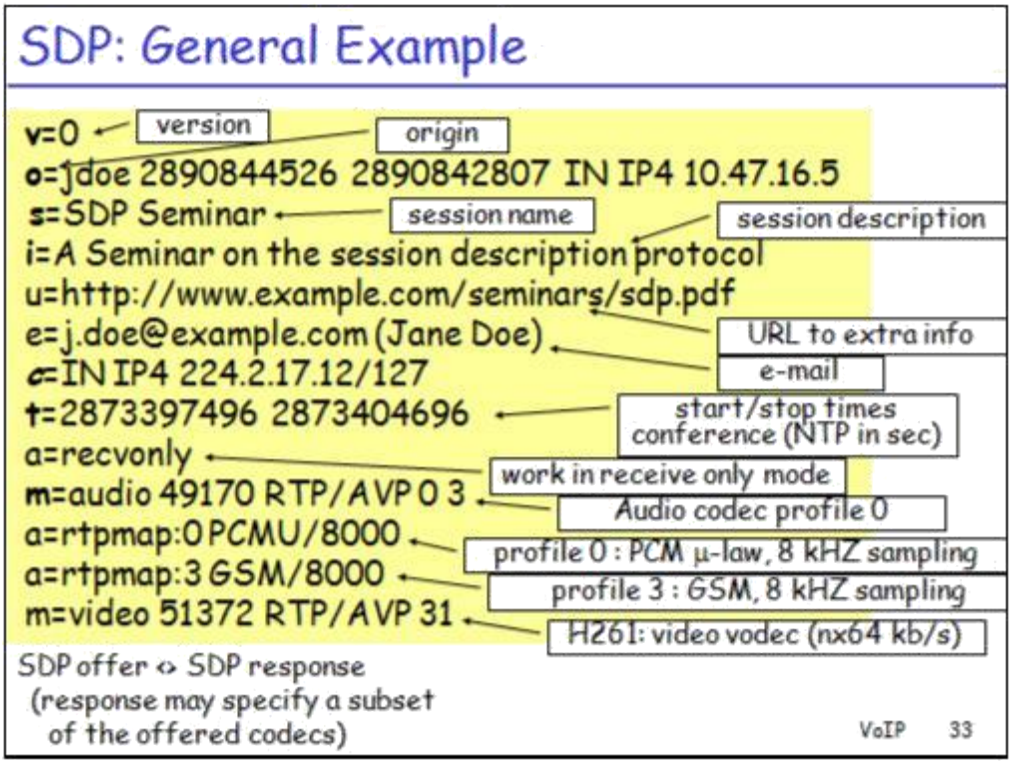


Different parameter categories used for VoIP are indicated by a single character (c, b, m, a):

- The connection information (indicated by **c**) is indicating the type of network (IP), the address format (IPv4 or IPv6), the destination address (unicast or multicast).
- The required amount of bandwidth (indicated by **b**), expressed in kbit/s.
- The media information (indicated by **m**):
  - whether it is audio, video, application (e.g. whiteboard, i.e. application data presented to the user), data (e.g. executable data not presented to the user) or control (e.g. an additional conference control channel for a session) information.
  - the port number (e.g. UDP port number in the range 1024-65535 with even numbers for RTP sessions and the next higher odd number for RTCP).
  - the transport protocol to be used for the media stream (UDP or RTP/AVP over UDP)
  - the media format (e.g. voice codec, video codec, …)
- Additional information (indicated by **a**): **ptime** (packet time in ms)

In this chapter on VoIP, the SDP description of a session will be transported in a SIP message.

Note that SDP may be used both for indicating the capabilities of one endpoint (offer) and the matching of the other endpoint (response). The caller may indicate for example 3 voice codecs that may be used, whereas the called party will respond with a single choice in a response SDP description.



This is a more general example of SDP (typically used for a multimedia conference). The details of the different parameters can be found in RFC 4566: SDP: Session Description Protocol (July 2006).

The example is describing the parameters used in a seminar on SDP that will be given by Jane Doe. More information can be found at http://www.example.com/seminars/sdp.pdf and for further information one may contact Jane Doe at j.doe@example.com. The seminar will use multicast address 224.2.1.1 (and the scope is limited to a TTL of 127, meaning that the multicast packets will be discarded after 127 hops). It is also indicated that the participants to the seminar should work in the receive only mode (it is not an interactive seminar). There are two audio codecs possible (PCM μ-law and GSM), specified dynamically in the SDP description.

There is one video stream possible, specified by the standard profile 31 (H261). (These standard audio/video profiles are specified in RFC 3551.)

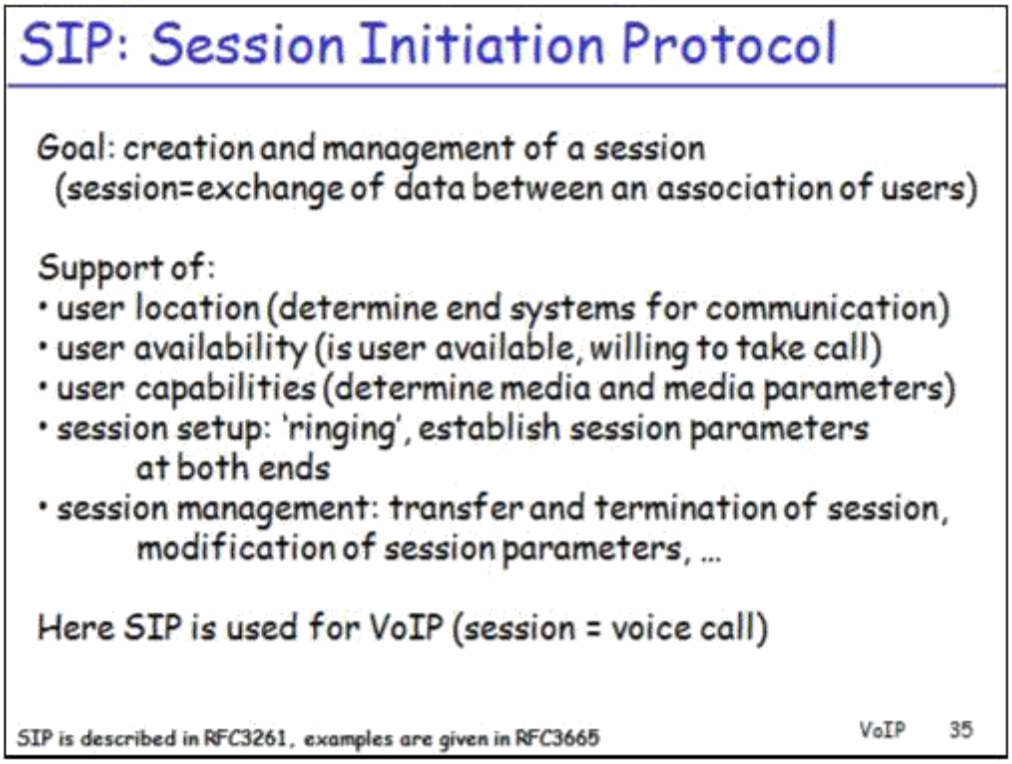
A number of the parameters indicated in this example are not appropriate for VoIP.

[NTP: Network Time Protocol: a protocol used to synchronize computers on the internet, RFC 1305]

[o = origin: 2nd and 3rd field are session- id and session-version resp.; id is a numeric string such that the tuple of <username>, <sess- id>, <nettype>, <addrtype>, and <unicast-address> forms a globally unique identifier for the session; version is increased when a modification is made to the session data.]

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
     - SDP (Session Description Protocol)
     - SIP (Session Initiation Protocol)
       - Intro
       - SIP compontents & operation
       - Security
       - PSTN interworking
   - User plane (terminal and network)
4. Multimedia over IP



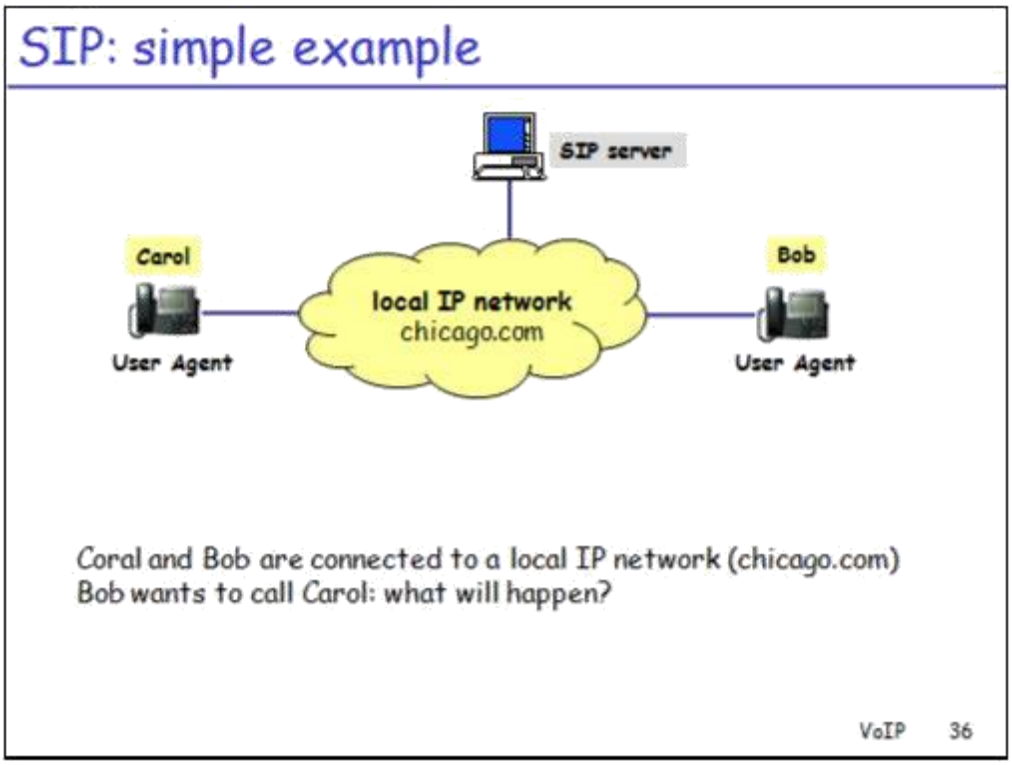
There are many applications of the Internet that require the creation and management of a session, where a session is considered an exchange of data between an association of participants. The implementation of these applications is complicated by the practices of participants: users may move between endpoints, they may be addressable by multiple names, and they may communicate in several different media - sometimes simultaneously. N umerous protocols have been authored that carry various forms of real-time multimedia session data such as voice, video, or text messages. The Session Initiation Protocol (SIP) works in concert with these protocols by enabling Internet endpoints (called user agents) to discover one another and to agree on a characterization of a session they would like to share. For locating prospective session participants, and for other functions, SIP enables the creation of an infrastructure of network hosts (called proxy servers) to which user agents can send registrations, invitations to sessions, and other requests. SIP is an agile, general-purpose tool for creating, modifying, and terminating sessions that works independently of underlying transport protocols and without dependency on the type of session that is being established.

SIP is an application- layer control protocol that can establish, modify, and terminate multimedia sessions (conferences) such as Internet telephony calls. SIP can also invite participants to already existing sessions, such as multicast conferences. Media can be added to (and removed from) an existing session. SIP transparently supports name mapping and redirection services, which supports personal mobility - users can maintain a single externally visible identifier regardless of their network location.

SIP supports five facets of establishing and terminating multimedia communications

• 1. User location: determination of the end system to be used for communication

• 2. User availability: determination of the willingness of the called party to engage in communications



• 3. User capabilities: determination of the media and media parameters to be used

• 4. Session setup: "ringing", establishment of session parameters at both called and calling party

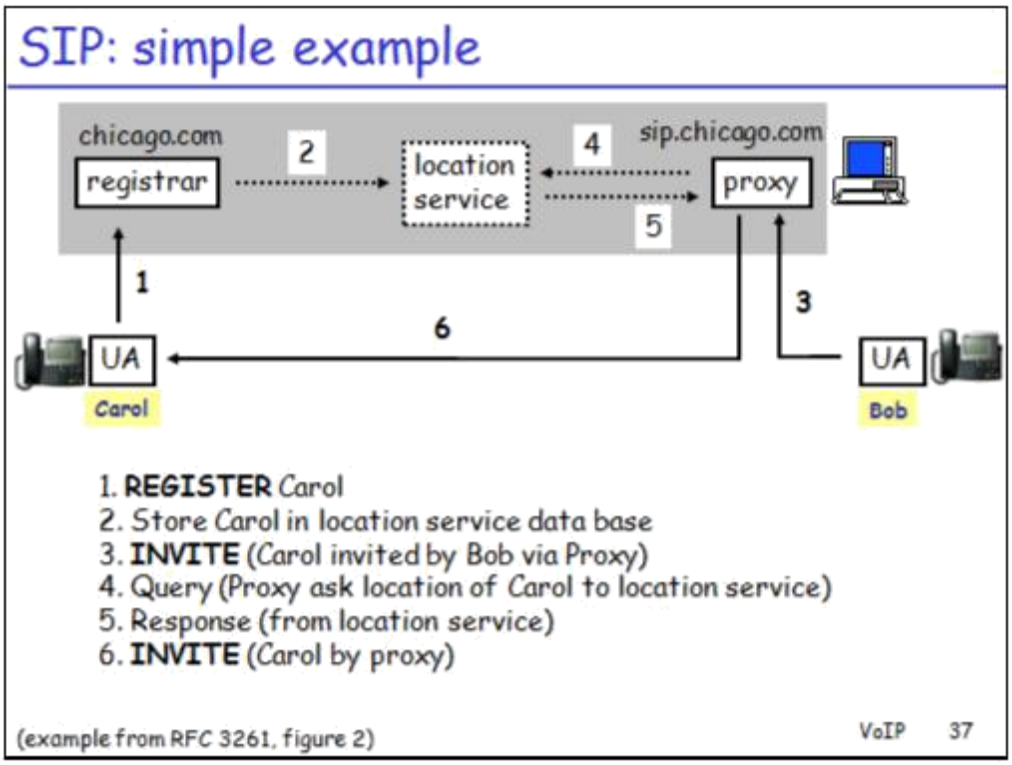
• 5. Session management: including transfer and termination of sessions, modifying session parameters, and invoking services

SIP is not a vertically integrated communications system. SIP is rather a component that can be used with other IETF protocols to build a complete multimedia architecture. Typically, these architectures will include protocols such as the Real-time Transport Protocol (RTP) (RFC 3550) for transporting real- time data and providing QoS feedback, the Real- Time streaming protocol (RTSP) (RFC 2326) for controlling delivery of streaming media, the Media Gateway Control Protocol (MEGACO) (RFC 3015) for controlling gateways to the Public Switched Telephone Network (PSTN), and the Session Description Protocol (SDP) (RFC 4566) for describing multimedia sessions. Therefore, SIP should be used in conjunction with other protocols in order to provide complete services to the users. However, the basic functionality and operation of SIP does not depend on any of these protocols.

SIP will support the set-up phase of a call, the termination of a call and the changes during a call. Therefore it is using a number of request and response messages. Examples are INVITE, OPTIONS, BYE, ACK, CANCEL, REGISTER.

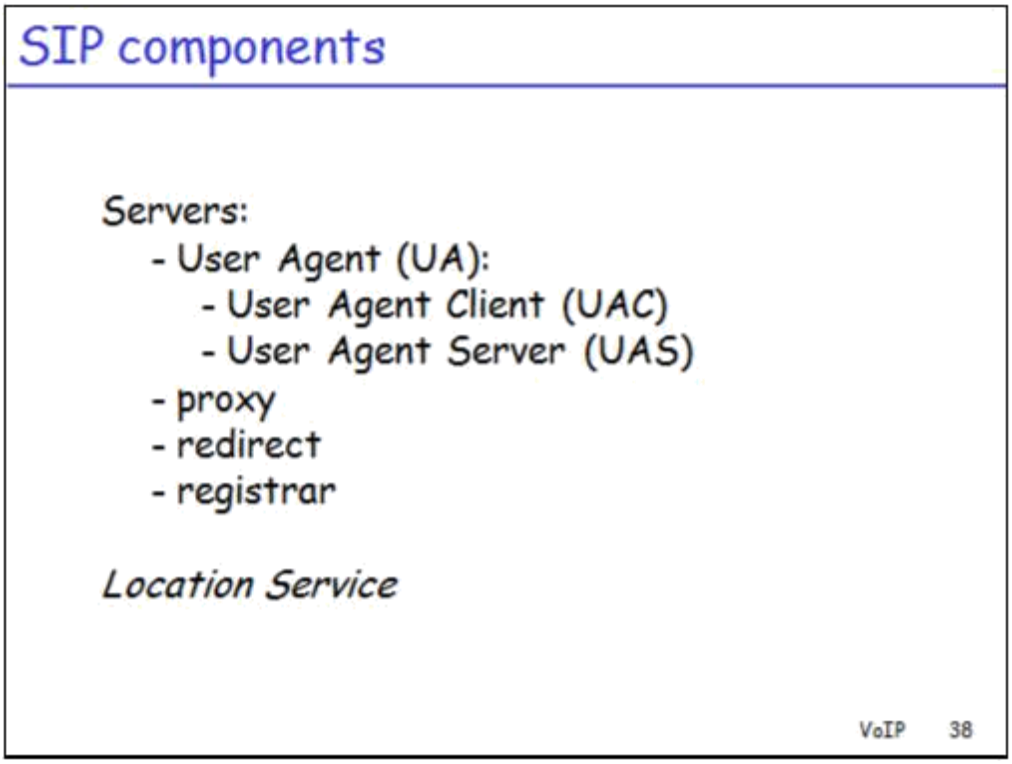
We will first consider a very simple example: two users (running a user agent on their terminal) are connected to an IP network and want to set-up a call (Bob will set-up a call to Carol). In the network there is a SIP server to support the call set-up (that SIP server may run several services, as explained next).

Note: Very interesting RFC's: SIP protocol in RFC3261, examples in RFC3665



A simplified message flow is illustrated in the figure. First Carol will REGISTER herself with the registrar server (1). This registrar will store the information in a location service database (2). When Bob wants to call Carol, he will send an INVITE to his proxy server (indicating that he wants to set- up a call with Carol) (3). Bob knows the location of this proxy server (e.g. he knows the IP address or he knows the name "sip.chicago.com" that can be resolved b y DNS). The proxy server will consult the location service database (4)(5) to know the location of Carol and will finally invite Carol. This will allow Bob and Carol to set-up the VoIP phone call.

In general the registrar, proxy and location service database are implemented on a single server (but this is not required).



Server: A server is a network element that receives requests in order to service them and sends back responses to those requests. Examples of servers are proxies, user agent servers, redirect servers, and registrars.

UserAgent (UA): A logical entity that can act as both a user agent client and user agent server **User Agent Client** (UAC): logical entity that creates and sends a new request. The role of UAC lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a UAC for the duration of that transaction. If it receives a request later, it assumes the role of a user agent server for the processing of that transaction.

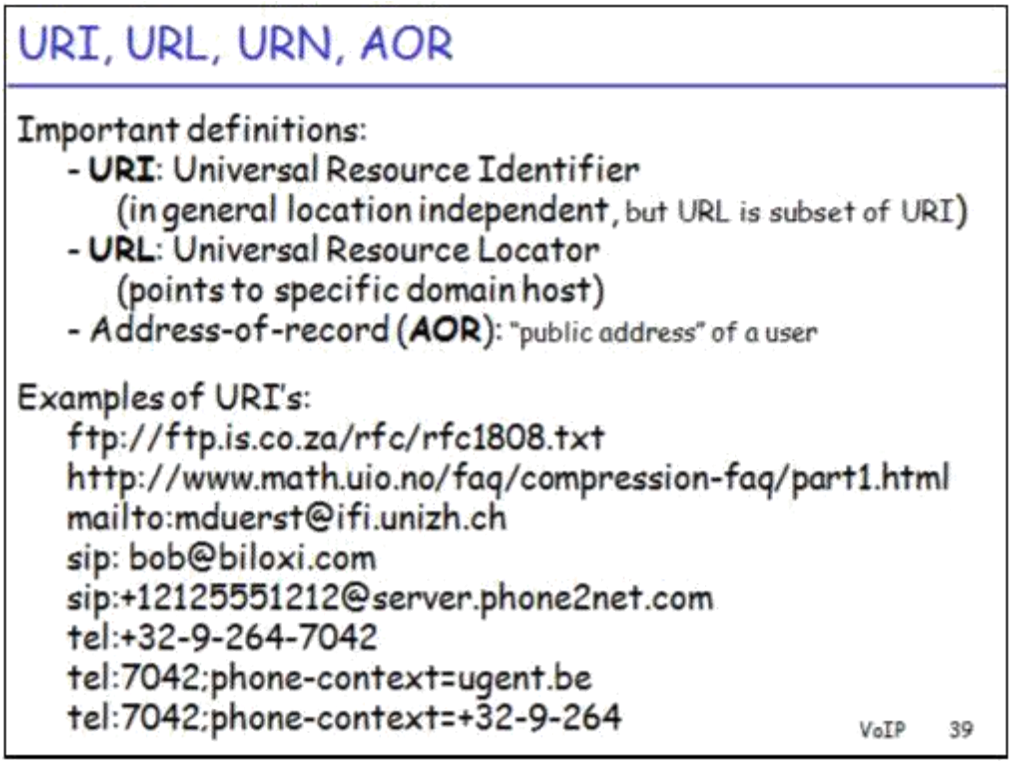
User Agent Se rver (UAS): A user agent server is a logical entity that generates a response to a SIP request. The response accepts, rejects, or redirects the request.

Proxy server: An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it.

Redirect server: A redirect server is a user agent server that generates 3xx responses [*Redirection: further action required to complete the request*] to requests it receives, directing the client to contact an alternate set of URI

Registrar: A registrar is a server that accepts REGISTER requests and places the information it receives into the location service for the domain it handles. It is similar to a Home Location Register (HLR) in GSM.

A **location service** is used by a SIP redirect or proxy server to obtain information about called party's possible location(s). It contains a list of bindings of address-of-record keys to zero or more contact addresses. The bindings can be created and removed in many ways; SIP defines a REGISTER method that updates the bindings. Note that contact addresses may have to be resolved using DNS. Note also that the protocol used between location service and proxy/redirect/registrar is not specified by SIP.



Uniform Resource Identifiers (URI) provide a simple and extensible means for identifying a resource. A URI can be further classified as a locator, a name, or both. The term "Uniform Resource Locator" (URL) refers to the subset of URI that identify resources via a representation of their primary access mechanism (e.g., their network "location"), rather than identifying the resource by name or by some other attribute(s) of that resource. The term "Uniform Resource Name" (URN) refers to the subset of URI that are required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable. More information in RFC 3986.

Address-of-Record: An address-of-record (AOR) is a SIP URI that points to a domain with a location service that can map the URI to another URI where the user might be available. An AO R is frequently thought of as the "public address" of the user. Example: sip: bob@biloxi.com

URI examples:

ftp://ftp.is .co.za/rfc/rfc1808.txt: URI for File Transfer Protocol services

http://www.math.uio.no/faq/compre s s ion-faq/part1.html: URI for Hypertext Transfer Protocol services

mailto:mdue rs t@ifi.unizh.ch: URI for electronic mail addresses

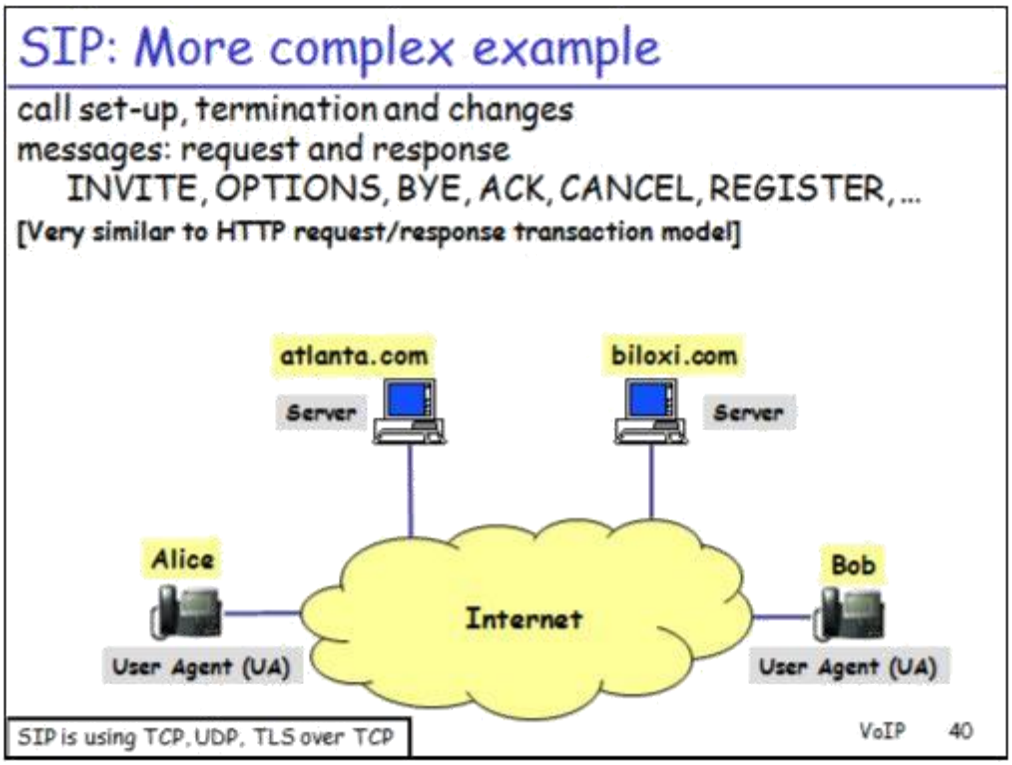
s ip: bob@biloxi.com: URI pointing to bob in the SIP context

s ip:+12125551212@server.phone2net.com;user=phone: URI pointing to phone in the SIP context

te l:+91-11-26597890: URI pointing to a phone number in India. The hyphens are inc luded to make the number more human readable; they separate country, area code local network and subscriber number

te l:7042;phone-context=iitd.ac.in : The URI describes a local phone number valid within the context "iitd.ac.in"

te l:7042;phone-context=+91-11-2659: The URI describes a local phone number that is valid within a particular phone prefix



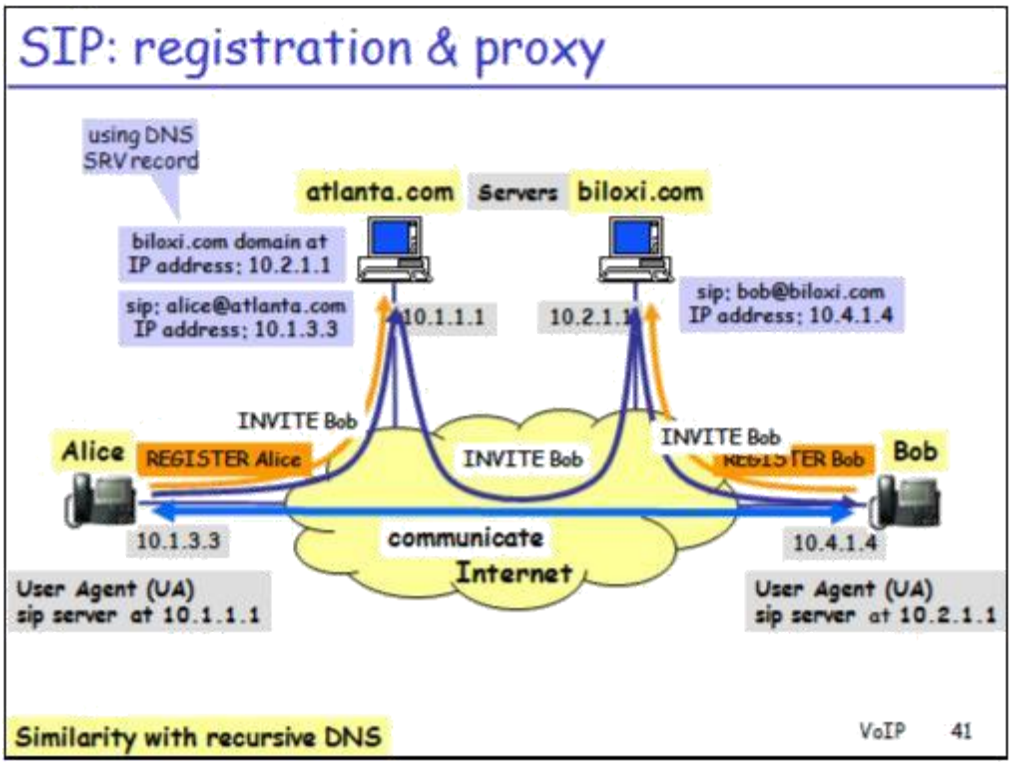
SIP will support the set-up phase of a call, the termination of a call and the changes during a call. Therefore it is using a number of request and response messages. Examples are INVITE, OPTIONS, BYE, ACK, CANCEL, REGISTER. SIP is very similar to the HTTP request/response model and uses the same format for the messages.

The figure illustrates a more complex example of the SIP building blocks. Alice and Bob are in a different domain (atlanta.com and biloxi.com) with their own servers (proxy/redirect, registrar, location service). Note that in practice even more servers will be involved.

The next slides will illustrate the operation of the servers in proxy or redirect mode.

Note: SIP may use TCP or UDP to transport messages. Along an end-to-end path, a mixture of UDP and TCP may be used (SIP has built- in reliability mechanisms). The protocol is indicated in the Via header. Example: TCP between Alice and atlanta.com and between atlanta.com and biloxi.com; UDP between biloxi.com and Bob. Also other transport protocols are possible, e.g. TLS over TCP. It is recommended that a server listens to the default SIP ports (5060 for TCP and UDP, 5061 for TLS over TCP).

Note: TLS (Transport Layer Security): The primary goal of the TLS Protocol is to provide privacy and data integrity between two communicating applications (very similar to SSL).



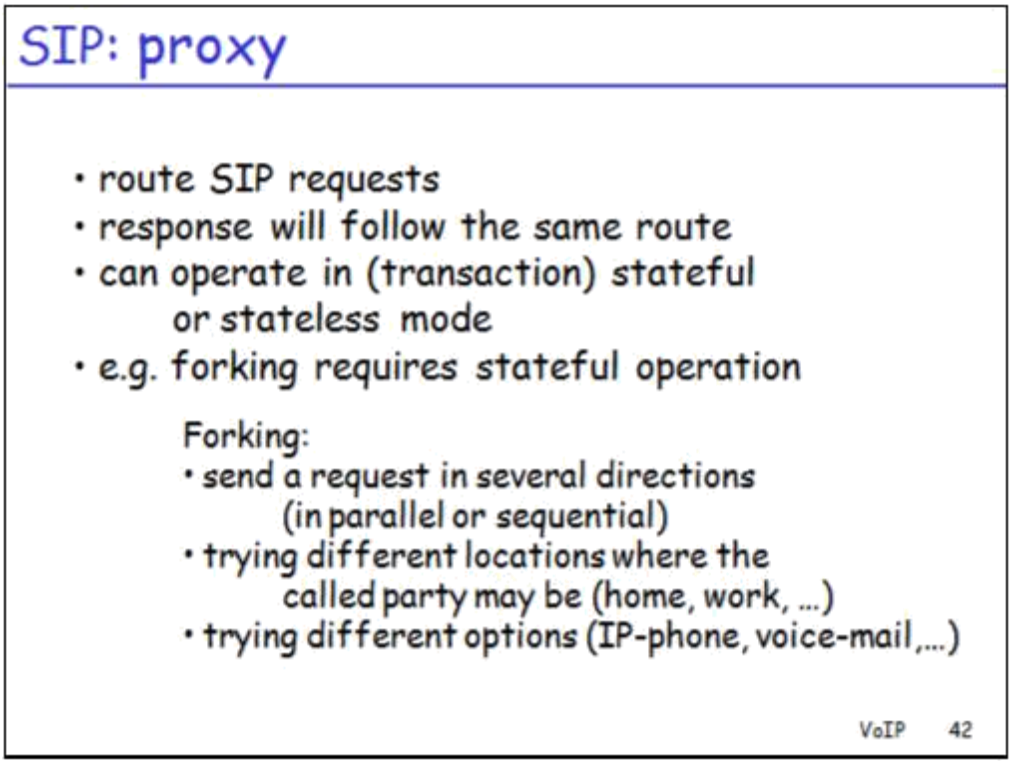
This first example show the registration and proxy operation of the SIP server. We have a user Alice in the atlanta.com "domain" and a second user Bob in the biloxi .com "domain". They have IP numbers 10.1.3.3 and 10.4.1.4 respectively (private IP numbers used for simplicity). The servers have IP numbers 10.1.1.1 (atlanta.com) and 10.2.1.1 (biloxi.com).

Before Alice can do anything, she has to configure here PC with the IP address of the SIP server where she wants to subscribe (10.1.1.1). This is similar to the IP number of the DNS server you want to use from your PC. The same holds for Bob (using the SIP server at 10.2.1.1). Note that a name of the SIP server is also possible (this name will be resolved by DNS).

The first action Alice and Bob will take is to register themselves to their respective SIP servers (using the **`REGISTER`** message). They will have a specific SIP URI (as indicated by **`sip:alice@atlanta.com`** and **`sip:bob@biloxi.com`**).

Now Alice can set up a call with Bob. In order to do this, she will send an **`INVITE`** message with all necessary information to set up the call. This will be sent to the SIP server of Alice (at 10.1.1.1). This SIP server will then find out where the SIP server for the biloxi.com domain (using a normal DNS) is. It will forward the **`INVITE`** message to the SIP server of biloxi.com (at 10.2.1.1). This SIP server will contact Bob because he knows that the SIP address of bob (**`sip:bob@biloxi.com`**) corresponds to the IP address 10.4.1.4. After sending some response messages, the communication will start. Note that it will no longer make use of the SIP servers (the User Agents of Alice and Bob will exchange information directly).

Note that the operation of the proxies is similar to recursive DNS where each DNS server is involved in a chain of requests. The records for SIP servers in a DNS are indicated by SRV (similar to MX for mail servers), see also RFC 2782, 2915 and 3263.

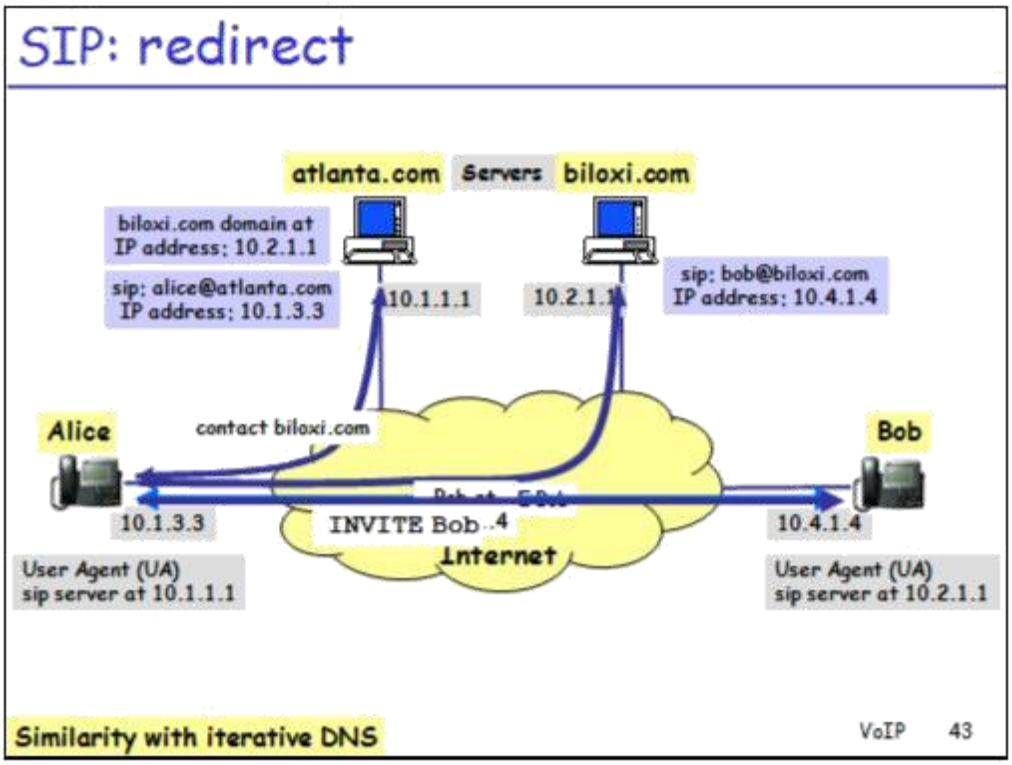


SIP proxies are elements that route SIP requests to user agent servers and SIP responses to user agent clients. A request may traverse several proxies on its way to its final destination. Each will make routing decisions, modifying the request before forwarding it to the next element. Responses will route through the same set of proxies traversed by the request in the reverse order.

Being a proxy is a logical role for a SIP element. When a request arrives, an element that can play the role of a proxy first decides if it needs to respond to the request on its own. For instance, the request may be malformed or the element may need credentials from the client before acting as a proxy. The element may respond with any appropriate error code. When responding directly to a request, the element is playing the role of a UAS.

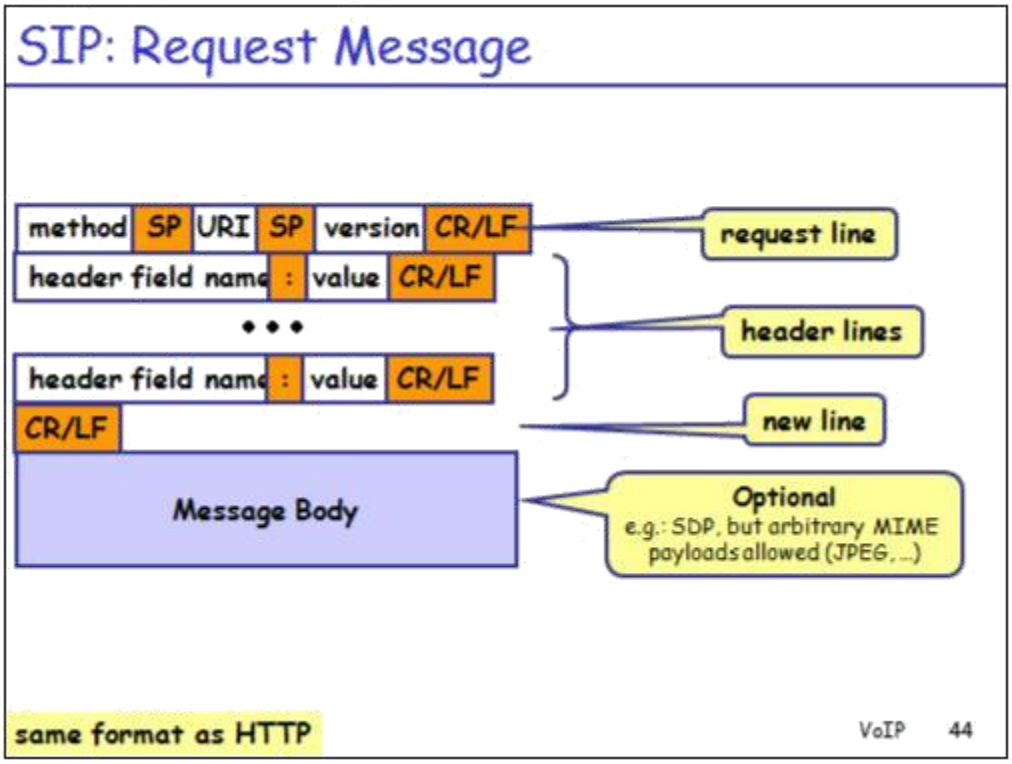
A proxy can operate in either a stateful or stateless mode for each new request. When stateless, a proxy acts as a simple forwarding element. It forwards each request downstream to a single element determined by making a targeting and routing decision based on the request. It simply forwards every response it receives upstream. A stateless proxy discards information about a message once the message has been forwarded. A stateful proxy remembers information (specifically, transaction state) about each incoming request and any requests it sends as a result of processing the incoming request. It uses this information to affect the processing of future messages associated with that request. A stateful proxy may choose to "fork“ a request, routing it to multiple destinations. Any request that is forwarded to more than one location must be handled statefully.

Important note: a SIP proxy will never be stateful with respect to a complete session (or call).



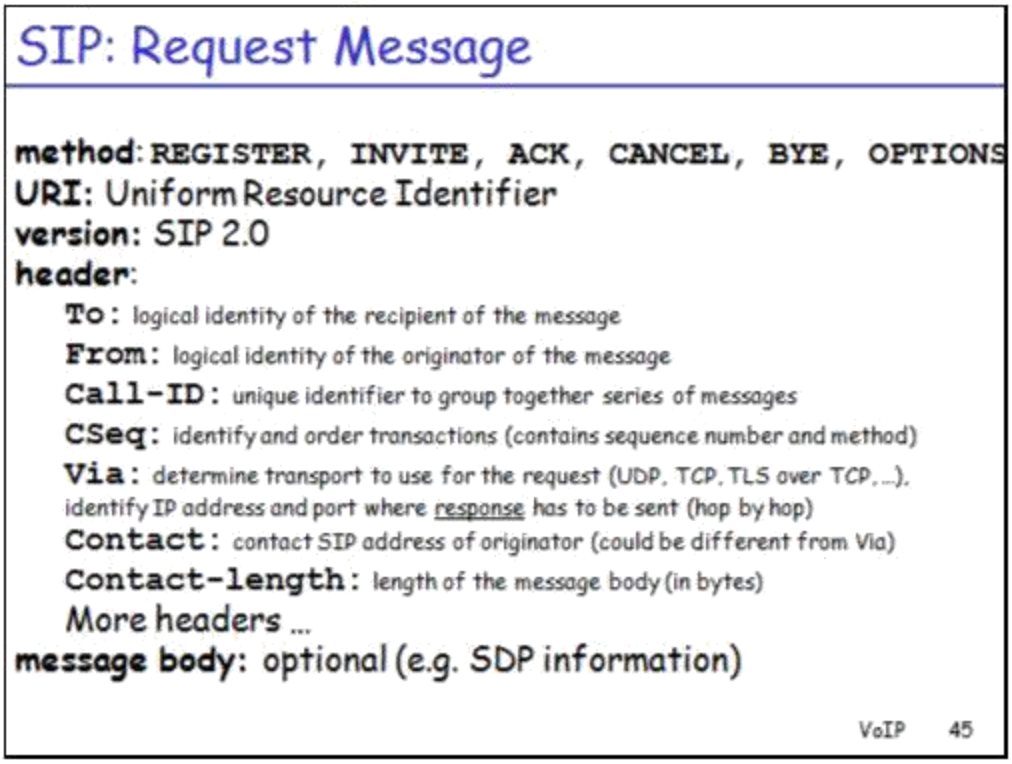
In some architectures it may be desirable to reduce the processing load on proxy servers that are responsible for routing requests, and improve signaling path robustness, by relying on redirection. Redirection allows servers to push routing information for a request back in a response to the client, thereby taking themselves out of the loop of further messaging for this transaction while still aiding in locating the target of the request. When the originator of the request receives the redirection, it will send a new request based on the URI(s) it has received. By propagating URIs from the core of the network to its edges, redirection allows for considerable network scalability. This is similar to the iterative operation of a DNS server.

The example shows the operation of the SIP server in redirect mode. The requests are always directly answered to the user agent client (the server does not take the initiative to contact the next server).



The layout of a request message is standardised and is shown in the figure. This is the same format as used in HTTP. The message body may be SDP but arbitrary MIME headers may be included: JPEG picture (e.g. a photo of the caller), ISUP (ISDN User Part), charging info, …

The request line contains the method (REGISTER, INVITE, ACK, CANCEL, BYE, OPTIONS, etc.), the URI (Uniform Resource Identifier) and the version of the SIP protocol.



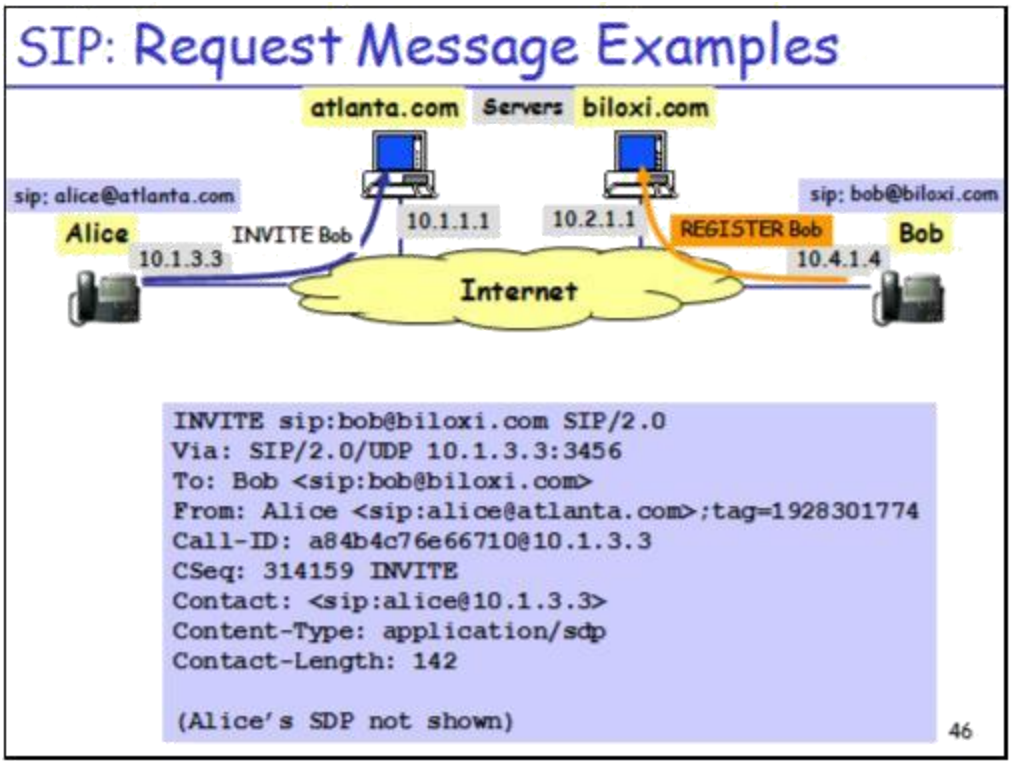
The request messages will first specify the method. Some examples are:

- REGISTER: used when a user agent is registering to a SIP server
- INVITE: used when a user agent client is setting up a call in order to invite the other party
- ACK: used as part of a three way handshake protocol (**`INVITE/200/ACK`**, see later example)
- CANCEL: used when a certain transaction has to be cancelled (by the client).
- BYE: used when the call is finished
- OPTIONS: used to negotiate the options of the user agents
- ...

A URI (Uniform Resource Identifier) will specify the destination for the message. Examples: for an INVITE this will be the destination User Agent, for a REGISTER this will be the responsible SIP server. Note that a URI may also contain a phone number.

A number of header lines are possible. Some important examples are listed above, but there are much more headers possible.

A body is optional (it may contain e.g. SDP information).

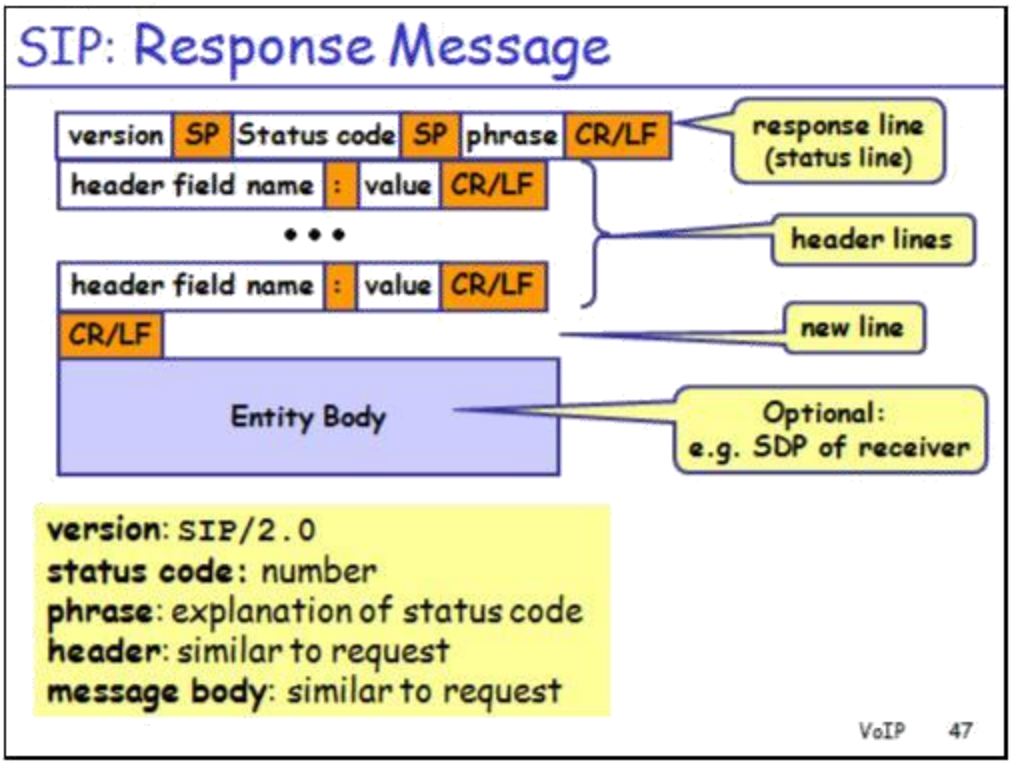


Two examples of SIP request messages are shown.

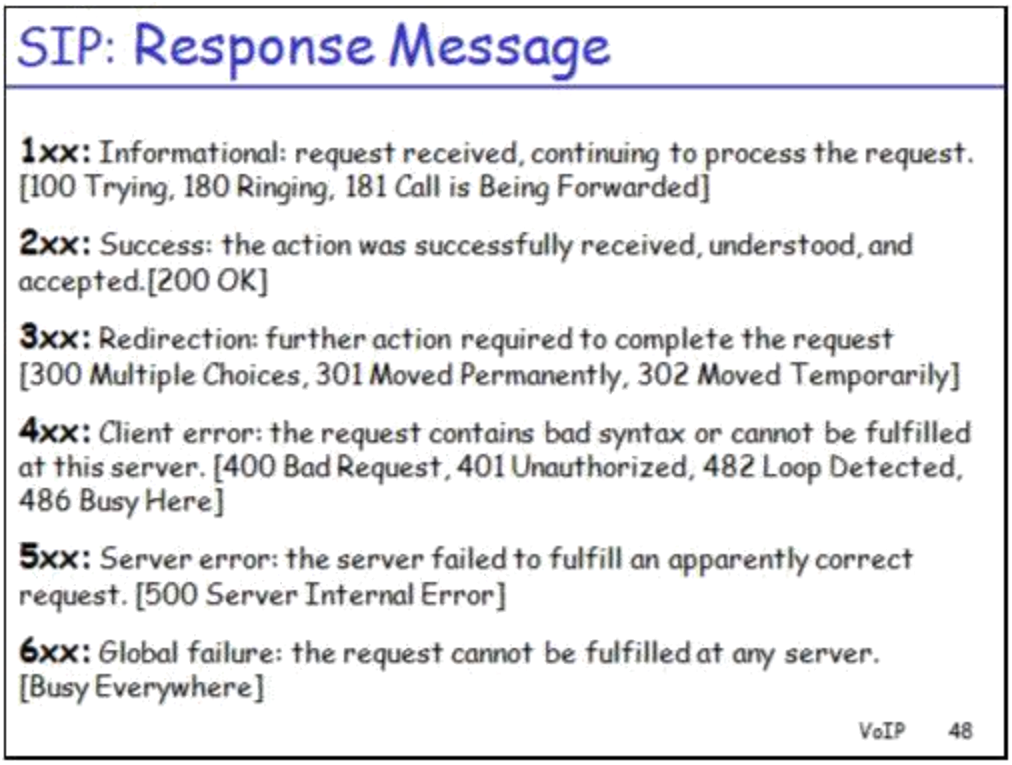
Below we observe a **REGISTER** message from Bob to the **sip:register.biloxi.com** server. The **To:** and **From:** fields are both the user agent client (**sip:bob@biloxi.com**). There is a call identification and a sequence number. The contact is **sip:bob@10.4.1.4** and the validity is 7200 minutes. No further information is included.

On the slide (above) an **INVITE** message is illustrated. In this example we observe a **Via:** line which indicates where the response to the request message has to be sent (IP address 10.1.3.3 at UDP port 3456). The content type is SDP and the length is 142 bytes.

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP 10.4.1.4:5060
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248 Call-
ID: 843817637684230@phone21.boxesbybob.com
CSeq: 1826 REGISTER
Contact: <sip:bob@10.4.1.4>
Expires: 7200 Contact-
Length: 0
```

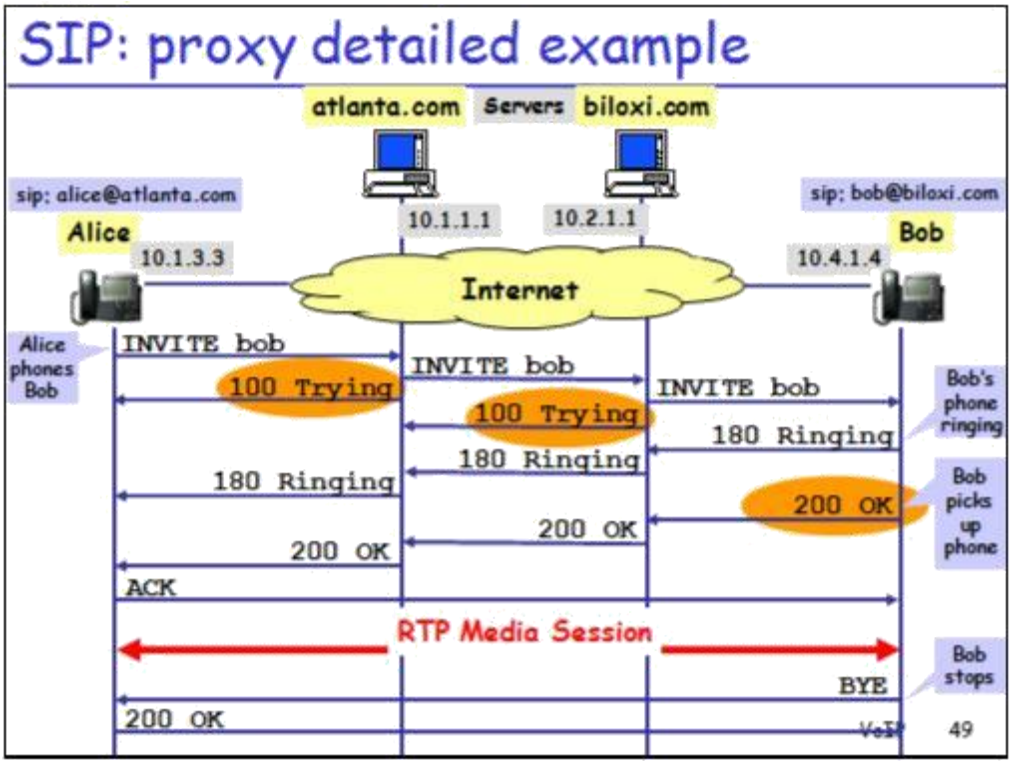


A response message looks similar to a request message but different fields are used. The response line starts with the SIP version and then it gives some status information (code + phrase explaining the code). Next again a number of header lines are possible. The last part of the response message contains the requested information.

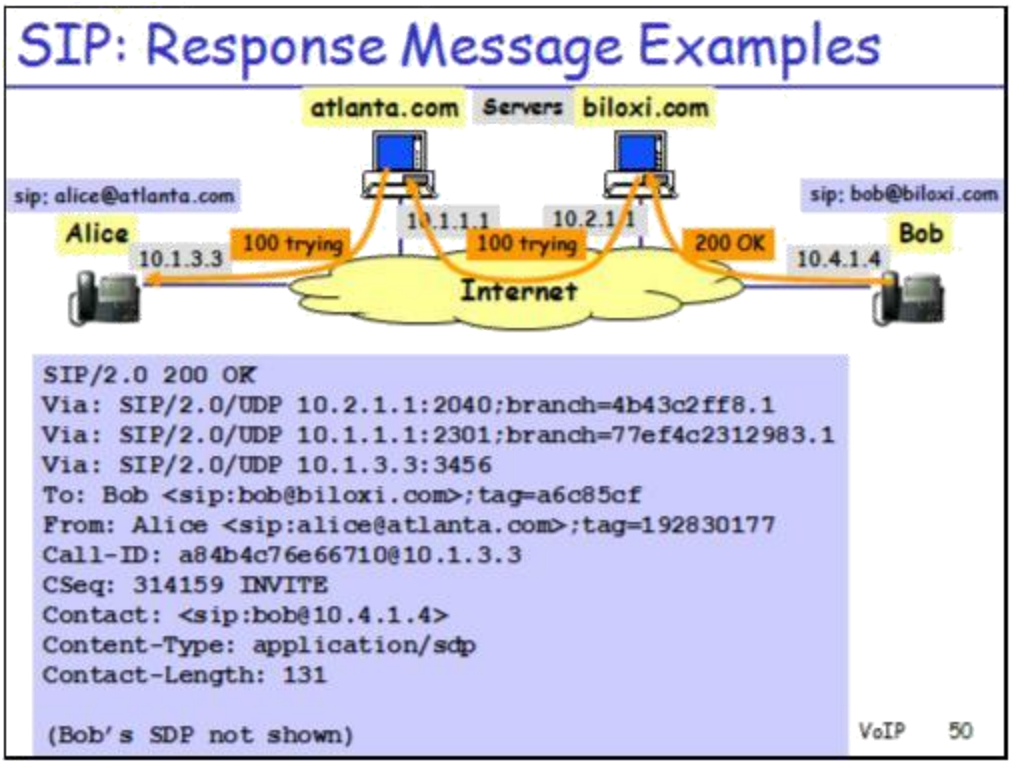


Status code classes:

- `1xx:` Informational: request received, continuing to process the request. [100 Trying, 180 Ringing, 181 Call is Being Forwarded]
- `2xx:` Success: the action was successfully received, understood, and accepted.[200 OK]
- `3xx:` Redirection: further action required to complete the request [300 Multiple Choices, 301 Moved Permanently, 302 Moved Temporarily]
- `4xx:` Client error: the request contains bad syntax or cannot be fulfilled at this server. [400 Bad Request, 401 Unauthorized, 482 Loop Detected, 486 Busy Here]
- `5xx:` Server error: the server failed to fulfil an apparently correct request. [500 Server Internal Error]
- `6xx:` Global failure: the request cannot be fulfilled at any server. [Busy Everywhere]



The figure illustrates an example of a call set-up when using a SIP proxy server. Alice will phone Bob and she is inviting him by sending an **`INVITE`** request message to her SIP server at atlanta.com. The atlanta.com server will forward the request to the server of Bob and at the same time it will send a response back to Alice (**`100 Trying`**). A similar action is taken by server biloxi.com. The **`INVITE`** from biloxi.com to Bob will result in Bob's phone starting to ring and at the same time a **`180 Ringing`** response message will be sent back to Alice (via the two SIP servers). When Bob picks up the phone a **`200 OK`** message will be sent back to Alice (via the two SIP servers) and Alice will send an **`ACK`** to Bob directly (from now on the SIP servers are no longer required). At that moment the conversation may start (RTP Media Session). The session will be stopped by Bob (**`BYE`**). Some examples of the response messages are shown in the next slide.



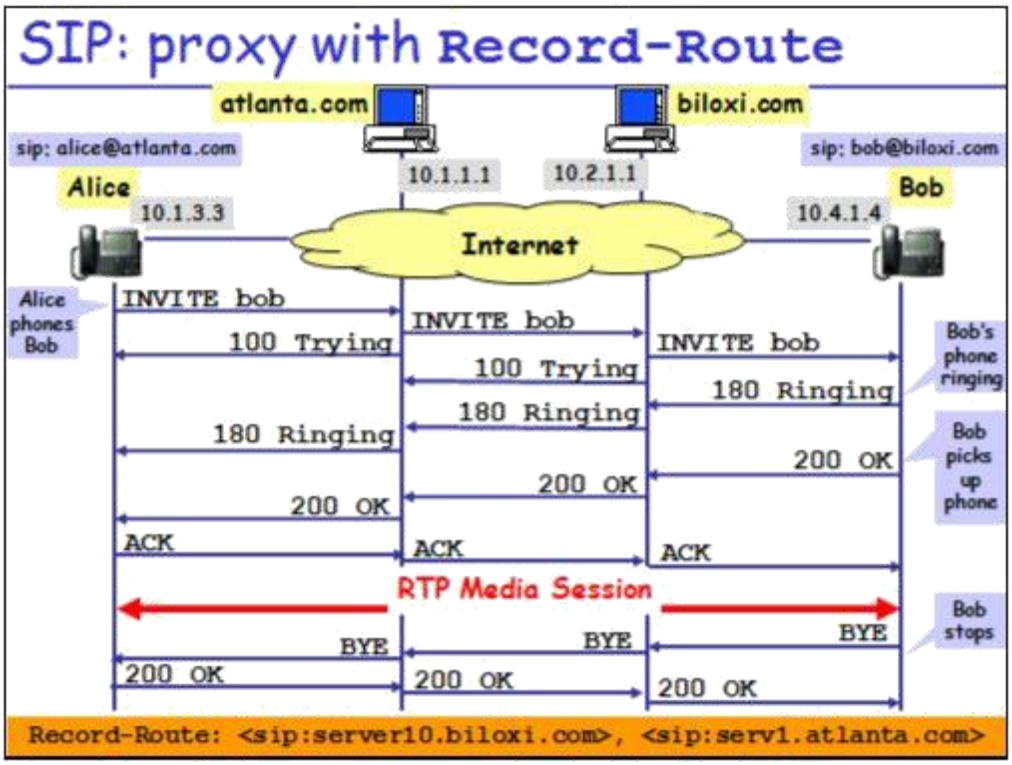
A few examples of response messages are illustrated. Below we observe the response (**`100 Trying`**) from the SIP server atlanta.com to Alice. The header field is nearly the same as in the **`INVITE`** message. Note that in the message from the biloxi.com SIP server to the atlanta.com SIP server, there is an extra **`Via:`** line indicating that the message has to go to the atlanta.com server. In the OK message (from Bob) we observe the 3 Via: lines indicating the way back to Alice. In this OK message, Bob's SDP is also added.

Trying from atlanta.com to Alice

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP 10.1.3.3:3456
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@10.1.3.3
CSeq: 314159 INVITE
Contact-Length: 0
```

Trying from biloxi.com to atlanta.com:

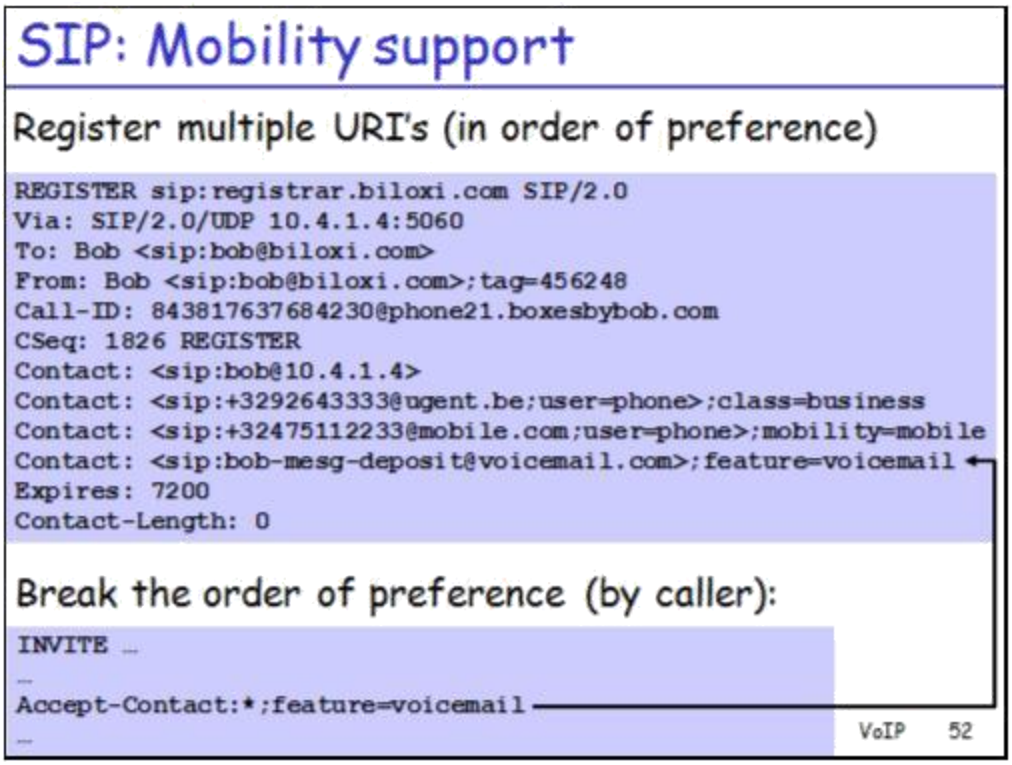
```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP 10.1.1.1:2301;branch=77ef4c2312983.1
Via: SIP/2.0/UDP 10.1.3.3:3456
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@10.1.3.3
CSeq: 314159 INVITE
Contact-Length: 0
```



In some cases, it may be useful for proxies in the SIP signaling path to see all the SIP messaging between the end points for the duration of the session (note: the RTP media session will not pass through the proxies). For example, if the atlanta.com and biloxi.com proxy servers wished to remain in the SIP messaging path beyond the initial **`INVITE`**, they would add to the **`INVITE`** a required routing header field known as **`Record-Route`** that contained a URI resolving to the hostname or IP address of the specific proxy. This information would be received by both Bob's SIP phone and (due to the **`Record-Route`** header field being passed back in the 200 (OK)) Alice's SIP phone and stored for the duration of the dialog. The atlanta.com and biloxi.com proxy server would then receive and proxy the ACK, BYE, and 200 (OK) to the BYE. Each proxy can independently decide to receive subsequent messages, and those messages will pass through all proxies that elect to receive it. This capability is frequently used for proxies that are providing mid-call features. Example of Record-Route:

```
Record-Route: <sip:server10.biloxi.com>, <sip:serv1.atlanta.com>
```

In this case the ACK, BYE and OK messages (at the end of the example) are routed through the proxies (biloxi and atlanta).



This REGISTER example illustrates the support of mobility. Bob registers a number of URI's where he may be reached. He will be first contacted at his perso nal SIP phone. If he is not responding, one will try his phone at work and next his mobile. If none of these respond, the call will be forward to his voicemail.

It is possible to specify in a request the way one wants to contact Bob (not sequential). The use of

`Accept-Contact:*;feature=voicemail`

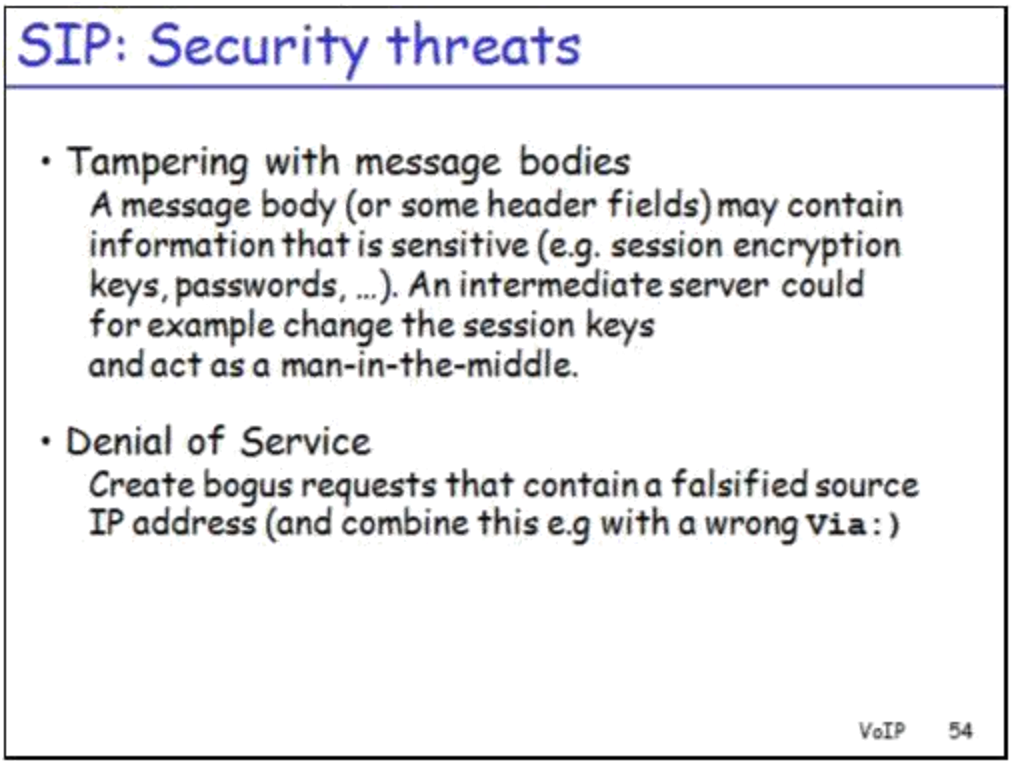
in an INVITE will direct the call directly to Bobs voice-mail.

Note that a user may also update his location service database every time he gets to a new location (each time using the REGISTER method).

Important note: SIP will also be used in the future for the signaling in the core network of the third generation mobile networks (3GPP: Third Generation Partnership Project, developing 3th generation networks). The system is called IMS (IP Multimedia core network Subsystem).

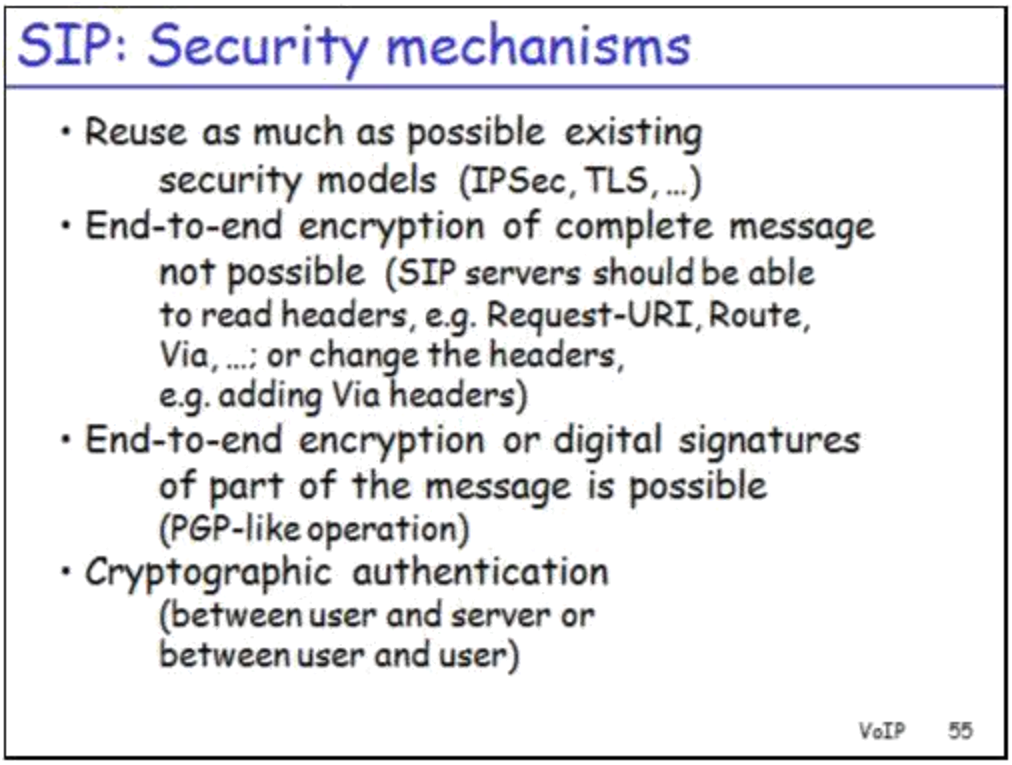
Registration Hijacking: The SIP registration mechanism allows a user agent to identify itself to a registrar as a device at which a user (designated by an address-of-record) is located. A registrar assesses the identity asserted in the From header field of a **`REGISTER`** message to determine whether this request can modify the contact addresses associated with the address-of-record in the To header field. While these two fields are frequently the same, there are many valid deployments in which a third-party may register contacts on a user's behalf. The From header field of a SI P request, however, can be modified arbitrarily by the owner of a UA, and this opens the door to malicious registrations. An attacker that successfully impersonates a party authorized to change contacts associated with an address-of-record could, for example, de-register all existing contacts for a URI and then register their own device as the appropriate contact address, thereby directing all requests for the affected user to the attacker 's device. This example shows the need for **authentication of the UA**.

Impe rsonating a server: Consider the case in which a registration sent to biloxi.com is intercepted by chicago.com, which replies to the intercepted registration with a forged 301 (Moved Permanently) response. This response might seem to come from biloxi.com yet designate chicago.com as the appropriate registrar. All future **`REGISTER`** requests from the originating UA would then go to chicago.com. This example show the need for **authentication of the server**.



Tampe ring with message bodies: Consider a UA that is using SIP message bodies to communicate session encryption keys for a media session. Although it trusts the proxy server of the domain it is contacting to deliver signaling properly, it may not want the administrators of that domain to be capable of decrypting any subsequent media session. Worse yet, if the proxy server were actively malicious, it could modify the session key, either acting as a man- in-the- middle, or perhaps changing the security characteristics requested by the originating UA. This family of threats applies not only to session keys, but to most conceivable forms of content carried end-to-end in SIP. These might include MIME bodies that should be rendered to the user, SDP, or encapsulated telephony signals, among others. Attackers might attempt to modify SDP bodies, for example, in order to point RTP media streams to a wiretapping device in order to eavesdrop on subsequent voice communications. For these reasons, the UA might want to secure SIP message bodies, and in some limited cases header fields, end-to-end. The security services required for bodies include confidentiality, integrity, and authentication. These end-to-end services should be independent of the means used to secure interactions with intermediaries such as proxy servers.

Denial of Service: Denial-of-service attacks focus on rendering a particular network element unavailable, usually by directing an excessive amount of network traffic at its interfaces. A distributed denial-of-service attack allows one network user to cause multiple network hosts to flood a target host with a large amount of network traffic. Attackers can create bogus requests that contain a falsified source IP address and a corresponding Via header field that identify a targeted host as the originator of the request and then send this request to a large number of SIP network elements, thereby using hapless SIP UAs or proxies to generate denial-of-service traffic aimed at the targeted host.

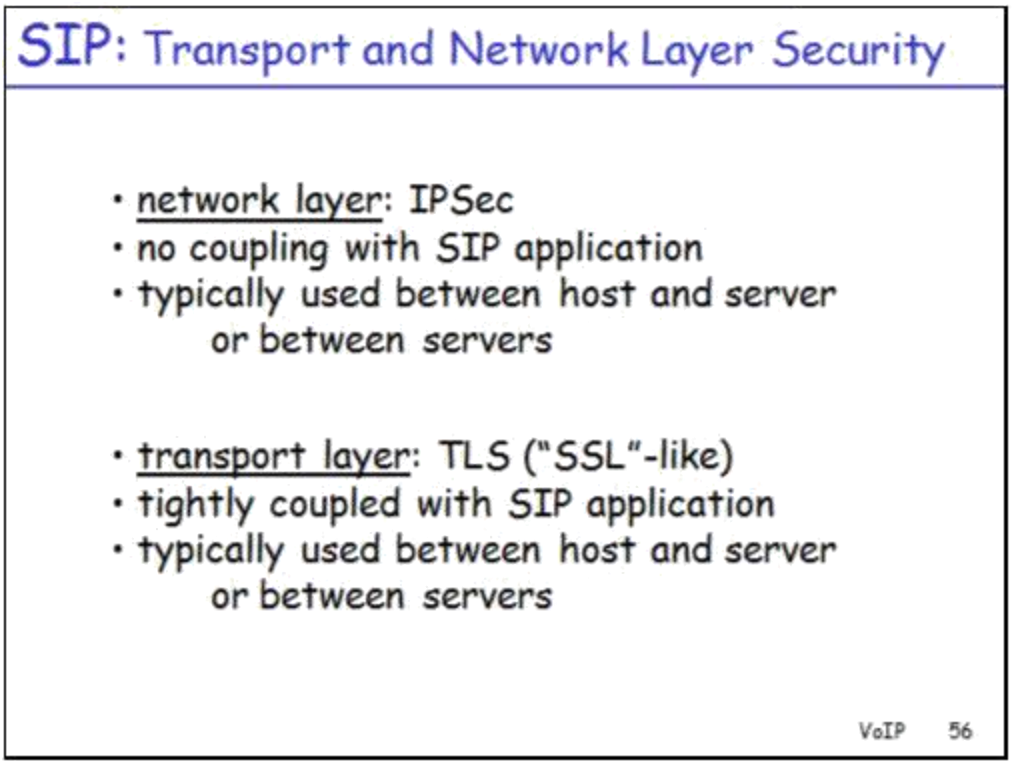


From the threats described above, we gather that the fundamental security services required for the SIP protocol are: preserving the confidentiality and integrity of messaging, preventing replay attacks or message spoofing, providing for the authentication and privacy of the participants in a session, and preventing denial-of-service attacks. Bodies within SIP messages separately require the security services of confidentiality, integrity, and authentication. Rather than defining new security mechanisms specific to SIP, SIP reuses wherever possible existing security models derived from the HTTP and SMTP space.

Full encryption of messages provides the best means to preserve the confidentiality of signaling - it can also guarantee that messages are not modified by any malicious intermediaries. However, SIP requests and responses cannot be naively encrypted end-to-end in their entirety because message fields such as the Request-URI, Route, and Via need to be visible to proxies in most network architectures so that SIP requests are routed correctly. Note that proxy servers need to modify some features of messages as well (such as adding Via header field values) in order for SIP to function. Proxy servers must therefore be trusted, to some degree, by SIP UAs. To this purpose, low- layer security mechanisms for SIP are recommended, which encrypt the entire SIP requests or responses on the wire on a (server level) hop-by- hop basis, and that allow endpoints to verify the identity of proxy servers to whom they send requests.

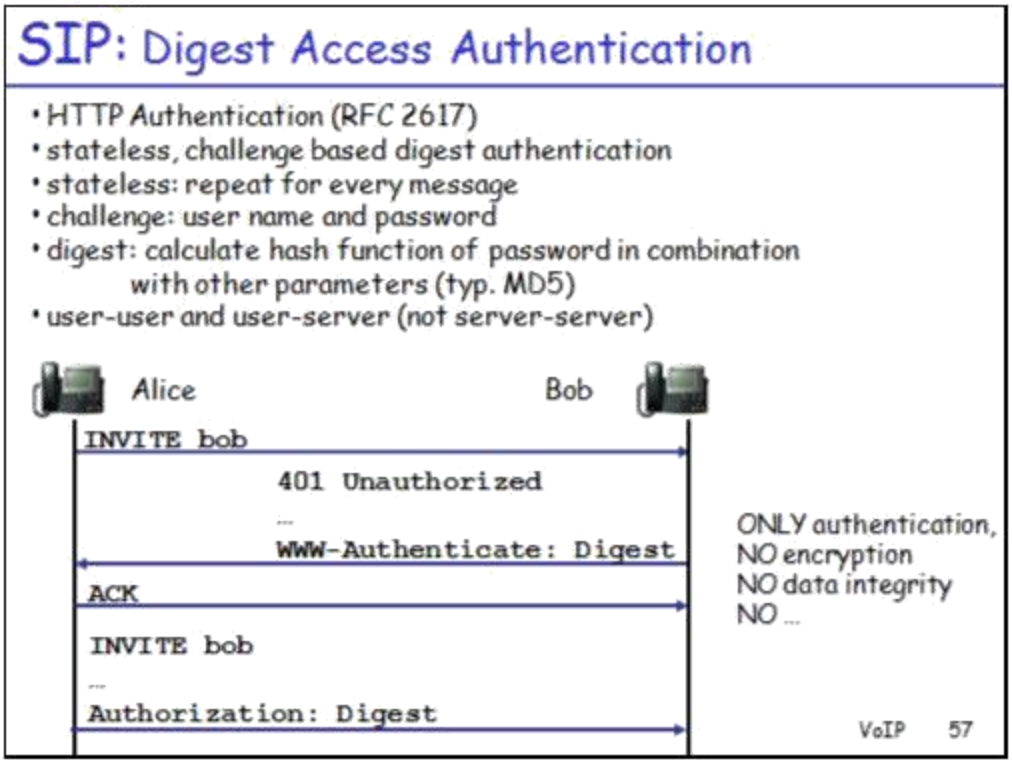
It is also possible to encrypt parts of the SIP message (e.g. the body) with encryption methods similar to PGP (although PGP is no longer supported in the latest RFC3261).

SIP entities also have a need to identify one another in a secure fashion. When a SIP endpoint asserts the identity of its user to a peer UA or to a proxy server, that identity should in some way be verifiable. A cryptographic authentication mechanism is provided in SIP to address this requirement.



IPSec is a set of network-layer protocol tools that collectively can be used as a secure replacement for traditional IP (Internet Protocol). IPSec is most commonly used in architectures in which a set of hosts or administrative domains have an existing trust relationship with one another. IPSec is usually implemented at the operating system level in a host, or on a security gateway that provides confidentiality and integrity for all traffic it receives from a particular interface (as in a VPN architecture). IPSec can also be used on a hop-by-hop basis.

TLS provides transport-layer security over connection-oriented protocols (TCP); "tls" (signifying TLS over TCP) can be specified as the desired transport protocol within a Via header field value or a SIP-URI. TLS is most suited to architectures in which hop-by-hop security is required between hosts with no pre-existing trust association. For example, Alice trusts her local proxy server, which after a certificate exchange decides to trust Bob's local proxy server, which Bob trusts, hence Bob and Alice can communicate securely. TLS must be tightly coupled with a SIP application. Note that transport mechanisms are specified on a hop-by-hop basis in SIP (e.g. between the UA and the proxy one may use TCP and between two proxies one may use UDP), thus a UA that sends requests over TLS to a proxy server has no assurance that TLS will be used end-to-end.



An important task is the authentication of the user (towards the server or towards the other user). Again SIP is using an already existing standard developed for HTTP authentication (see also RFC 2617).

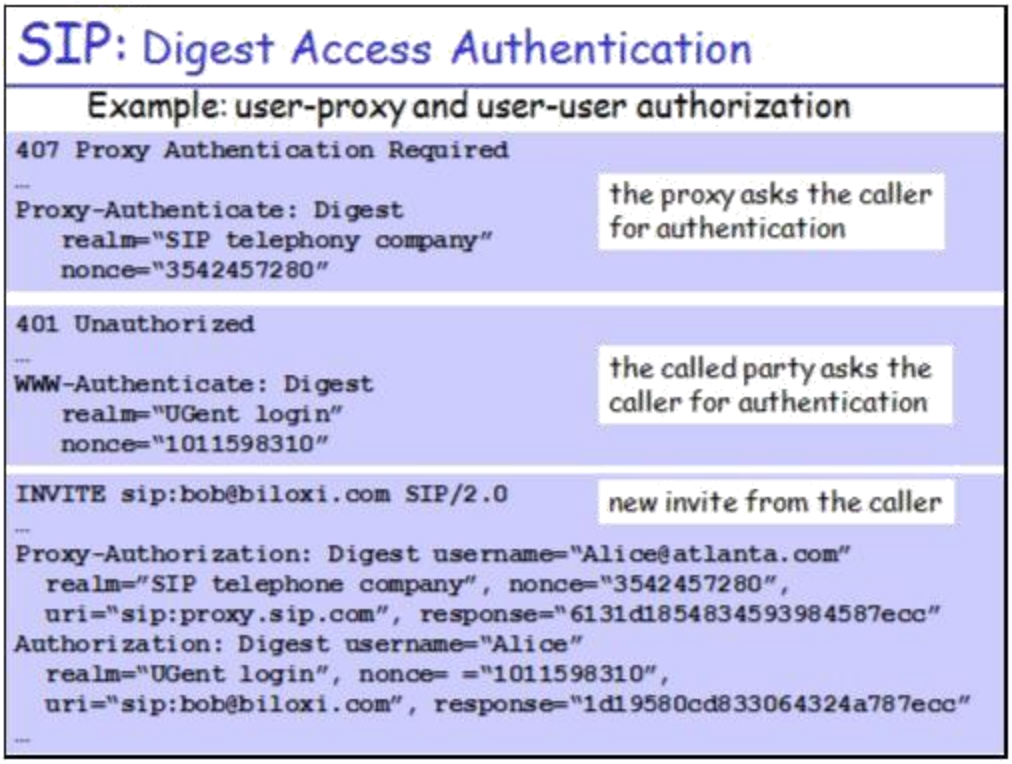
Consider the case where Bob wants to authenticate Alice when he is invited by Alice.

Alice will send an **`INVITE`** to Bob but she does not know that authentication is required. Bob will respond with a **`401 Unauthorized`** reply message asking Alice to authenticate herself. Alice will ACK this reply and will send a new **`INVITE`** message, now including the necessary authentication.

This process is stateless: authentication is required for every message.

The process is challenge/response based: the challenge is the request to provide a username and password, the response will provide back the username and password. HTTP was initially using "Basic Authentication" where the password was sent as clear tex t in a HTTP message. A much better approach is the HTTP Digest Access Authentication where the password is not sent in clear text but a digest (typ. MD5 hash function) is calculated of the password (in combination with other parameters). In this way the password is not readable when sent over the Internet.

In a similar way, a server may ask a user for authentication (using **`407 Proxy Authentication Required`**).

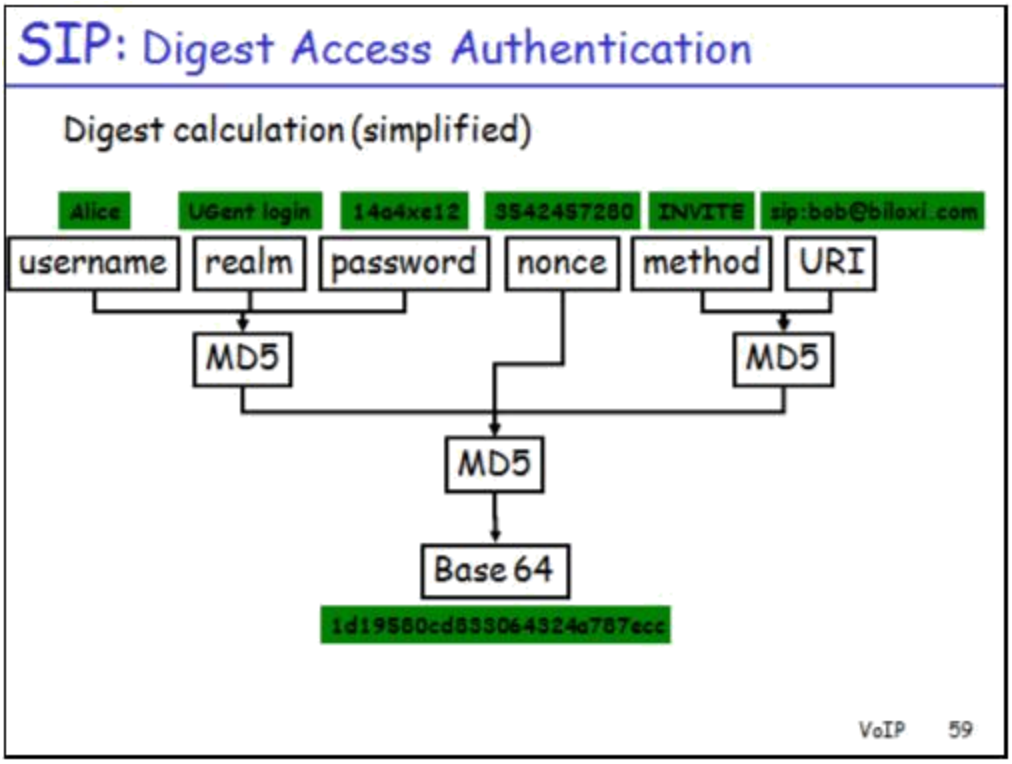


Parts of the messages exchanged in the user-proxy and user- user authentication process are illustrated.

When a caller is sending an **INVITE** message to the called party, this message will first pass his proxy. This proxy may ask for authentication by replying with a **407 Proxy Authentication Required** response message (see first message above). This message will a.o. contain a **Proxy-Authenticate** field (indicating that a digest is requested). This field also contains a realm, used to identify the scope of the authorization asked. In this example the "SIP telephony company" username/password of the caller is asked. The nonce is used in the digest calculation (to avoid replay attacks). The caller will send a new **INVITE** to the proxy (with the correctly filled **Proxy-Authorization** field) who will forward it to the called party (eventually over some other proxy servers).

When the **INVITE** arrives at the called party, this user will respond with a **401 Unauthorized** message, using the **WWW-authenticate** field, now with a realm "UGent login" (indicating that the username/password should be used to login at UGent) (see second message above).

The caller will now again send a new **INVITE** with both the correct **Proxy-Authorization** and **Authorization** fields (see third message). We observe the digest username (in clear text), the realm, the nonce, URI of the server or user that requested the authentication and the response (which is the actual digest, see next).



A simplified example of the calculation of a digest is shown above (indicated by **response="1d19580cd833064324a787ecc"** in the previous slide). A first hash (e.g. MD5 with a 128 bits hash) is calculated over username, realm and password, a second hash is calculated over method and URI. Finally a last hash is calculated over the two other hash results and the nonce. This will be Base 64 coded.

Base 64 coding:

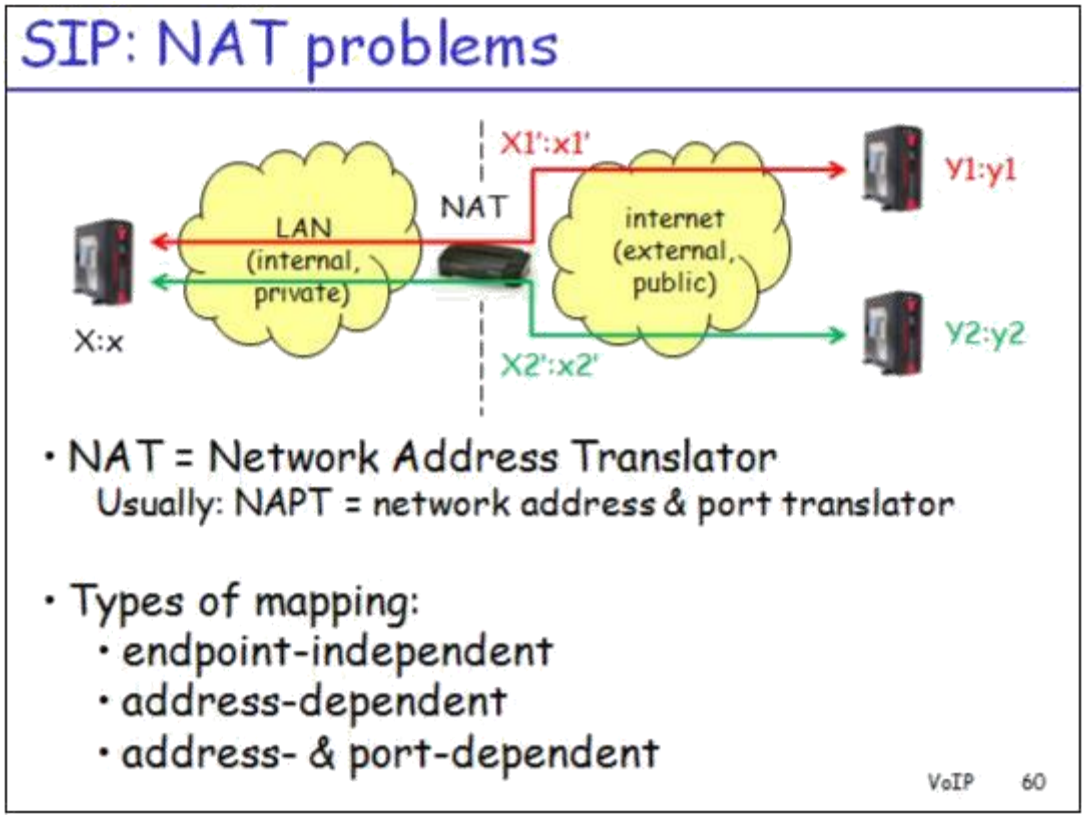
Each character is converted to ASCII (in hexadecimal notation):

A 41 , l 6C, i 69, c 63, e 65, ...)

This is put in a binary form (4 bits per hex symbol) and 6 bits are put together and encoded in Base-64 encoding, then the 6 bits are represented in a decimal form and afterwards in a hexadecimal form:

4	1	6	C	6	9	6	3	6	5	(ASCII in hex notation)
0100	0001	0110	1100	0110	1001	0110	0011	0110	0101	(bin groups of 4)
010000	010110	110001	101001	011000	110110	0101…				(bin groups of 6)
16	22	49	41	24	54	...				(dec)
10	16	31	29	18	36	...				(hex)
Q	W	x	p	Y	2	...				(BASE-64)

More info on conversions see: RFC 4648 (The Base16, Base32, and Base64 Data Encodings), www.lookuptables.com, www.ascii.cl/conversion.htm



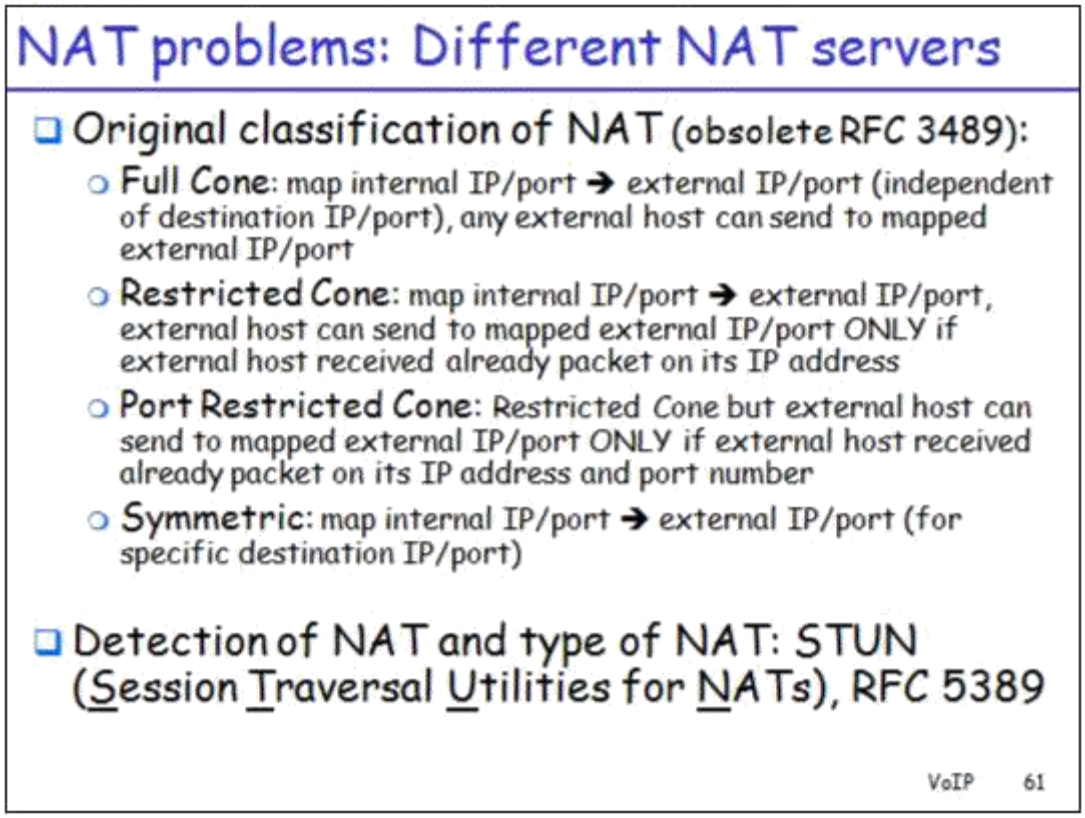
Traditional NAT [RFC3022] has two main varieties -- Basic NAT and Network Address/Port Translator (NAPT). NAPT is by far the most commonly deployed NAT device. NAPT allows multiple internal hosts to share a single public IP address simultaneously. When an internal host opens an outgoing TCP or UDP session through a NAPT, the NAPT assigns the session a public IP address and port number, so that subsequent response packets from the external endpoint can be received by the NAPT, translated, and forwarded to the internal host. The effect is that the NAPT establishes a NAT session to translate the (private IP address, private port number) tuple to a (public IP address, public port number) tuple, and vice versa, for the duration of the session.

The following address and port mapping behavior are defined [RFC4787]:

• *Endpoint-Independent Mapping:* The NAT reuses the port mapping for subsequent packets sent from the same internal IP address and port (X:x) to any external IP address and port. Specifically, X1':x1' equals X2':x2' for all values of Y2:y2

• *Address-Dependent Mapping:* The NAT reuses the port mapping for subsequent packets sent from the same internal IP address and port (X:x) to the same external IP address, regardless of the external port. Specifically, X1':x1' equals X2':x2' if and onlyif, Y2 equals Y1.

• *Address and Port-Dependent Mapping:* The NAT reuses the port mapping for subsequent packets sent from the same internal IP address and port (X:x) to the same external IP address and port while the mapping is still active. Specifically, X1':x1' equals X2':x2' if and only if, Y2:y2 equals Y1:y1.



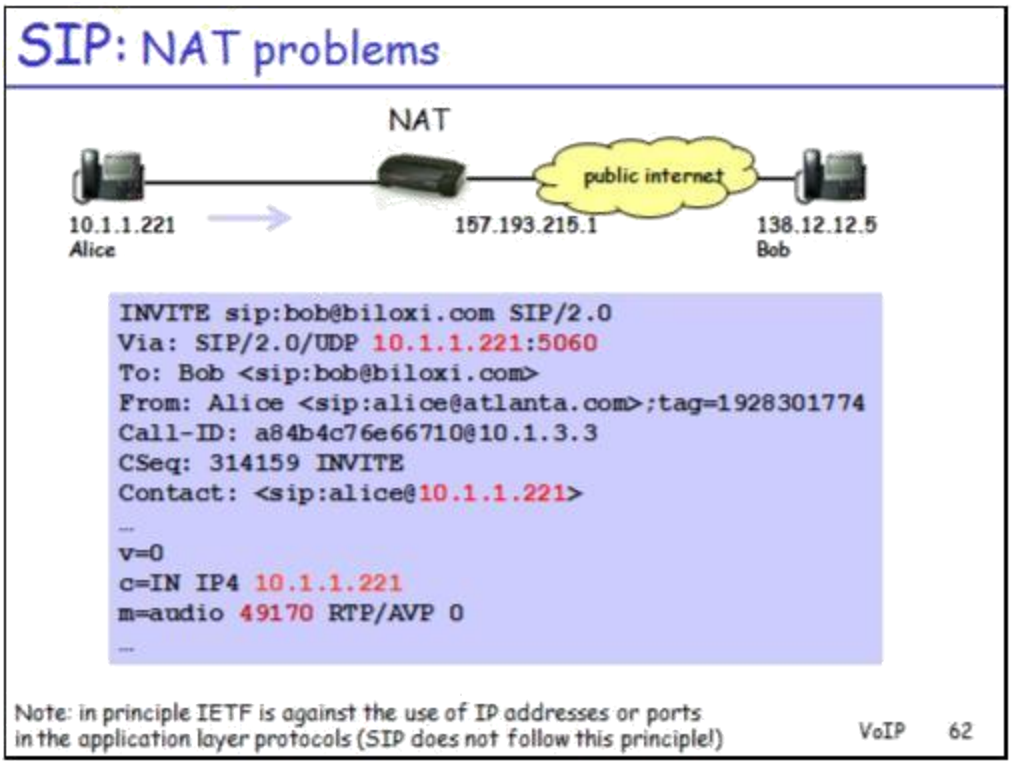
It has been observed that NAT treatment of UDP varies among implementations. The original classification in obsolete RFC3489 (March 2003):

- Full Cone: A full cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port (independent of the destination). Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address and host.

- Restricted Cone: A restricted cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Unlike a full cone NAT, an external host (with IP address X) can send a packet to the internal host only if the internal host had previously sent a packet to IP address X.

- Port Restricted Cone: A port restricted cone NAT is like a restricted cone NAT, but the restriction includes port numbers. Specifically, an external host can send a packet, with source IP address X and source port P, to the internal host only if the internal host had previously sent a packet to IP address X and port P.

-Symmetric: A symmetric NAT is one where all requests from the same internal IP address and port, to a specific destination IP address and port, are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a packet can send a UDP packet back to the internal host.



The use of N AT (Network Address Translation) is as such not compatible with SIP. The major problems are:

1) The response to a request can not be routed back to the originator because the private IP address in the **`Via`** header (10.1.1.221) is not routable in the public internet.

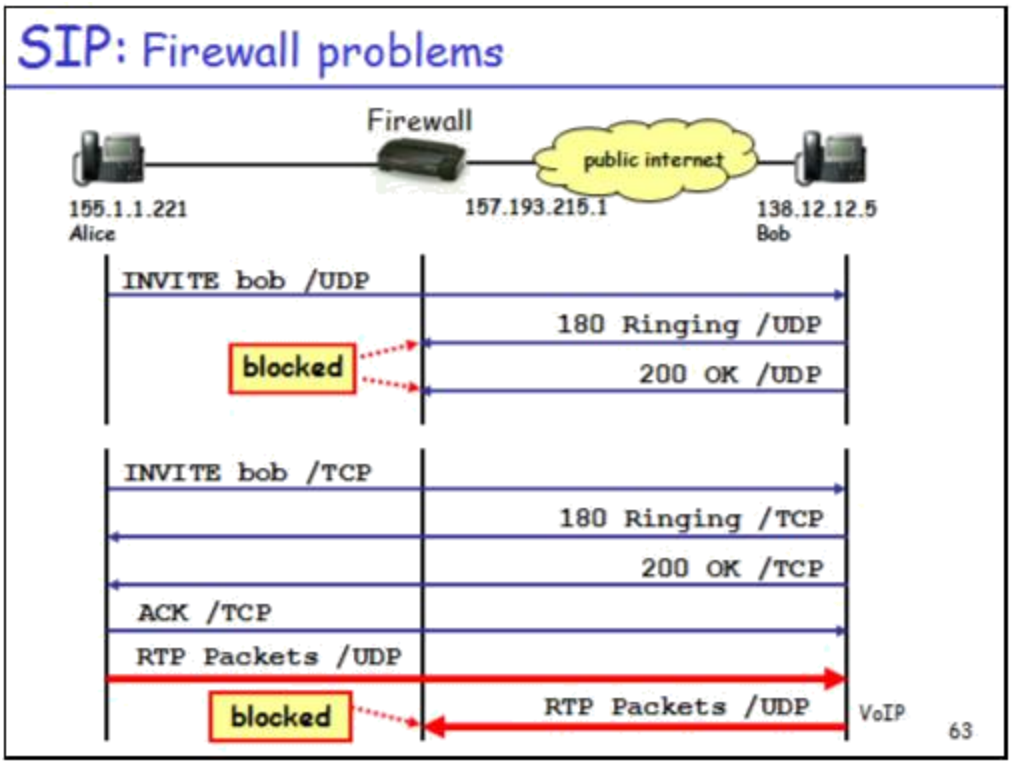
2) Future requests during a session would be misrouted (based on an incorrect **`Contact`** header).

3) RTP packets sent by Bob would be misrouted (based on an incorrect IP address for the media in the SDP description: **`c=IN IP4 10.1.1.221`**)

Note that the two port numbers (5060 and 49170) may be changed by the NAT and may cause signaling or media exchange to fail.

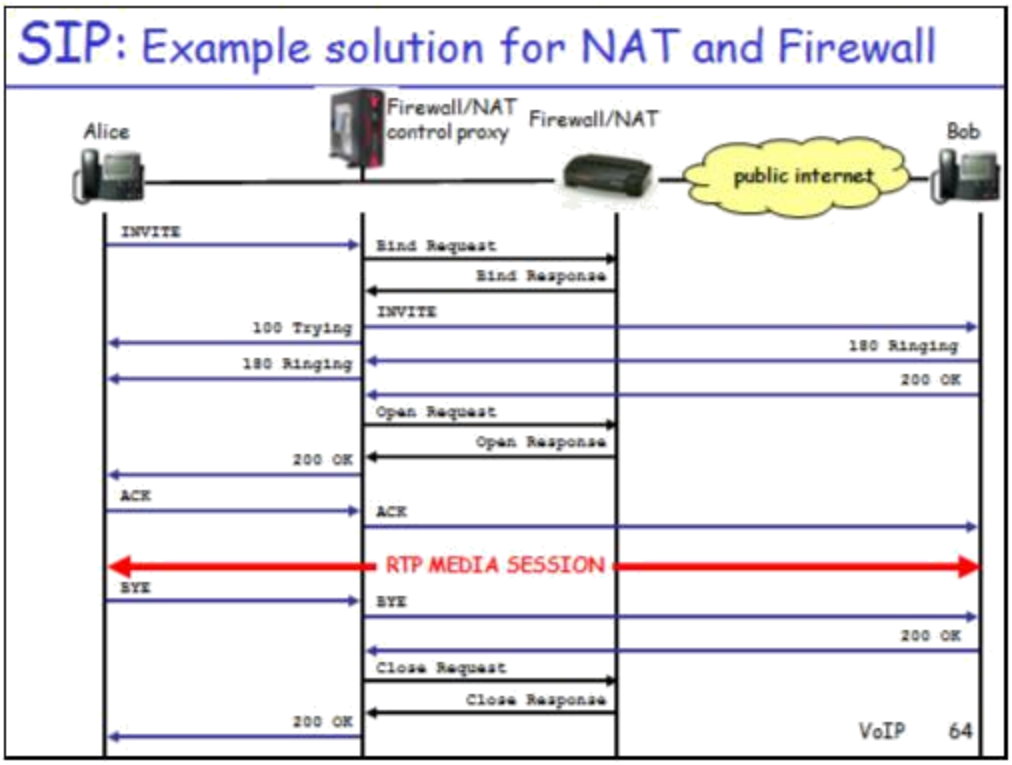
Only 1) can be solved in SIP using the Via parameters: received=<IP address it was received from>; rport=<the port incoming request was received on>

The rport extension is defined in RFC3581: see its section 6 for an example.



The use of a firewall is also a problem for SIP based communication. A firewall will typically block the UDP ports for incoming packets. If UDP is used to communicate between the user and the proxy, the packets from the proxy will be blocked. In addition, the RTP stream (typically running on UDP) will also be blocked.

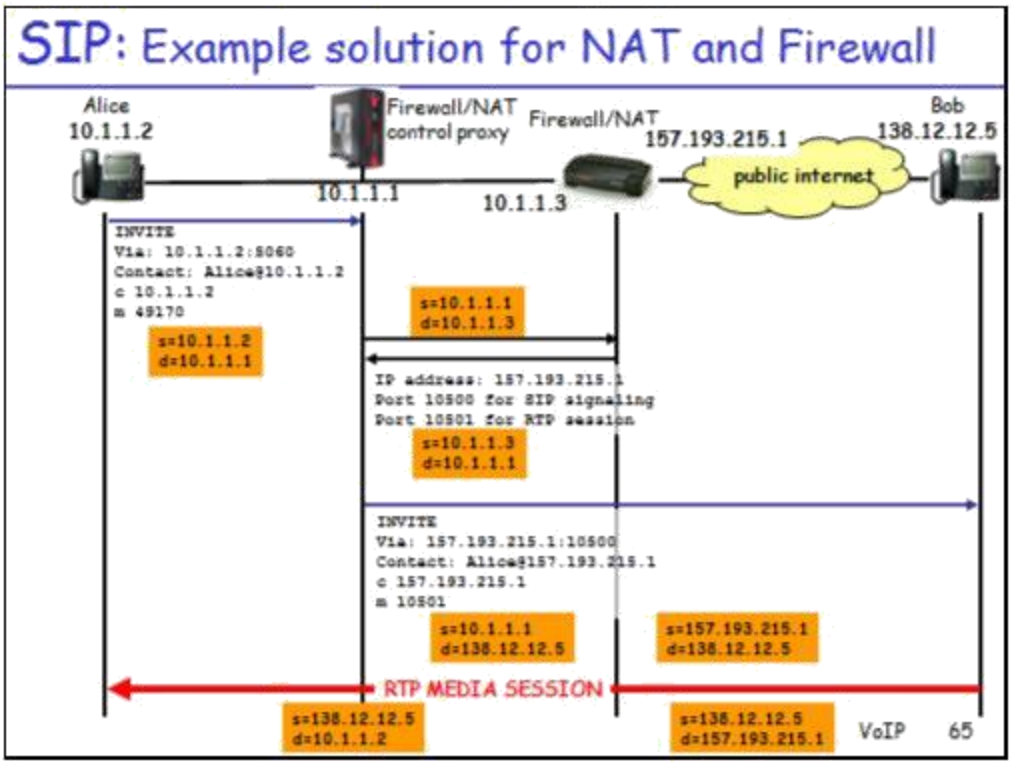
Two examples are shown above: the first example uses UDP for the signaling, the second example is using TCP for the signaling.



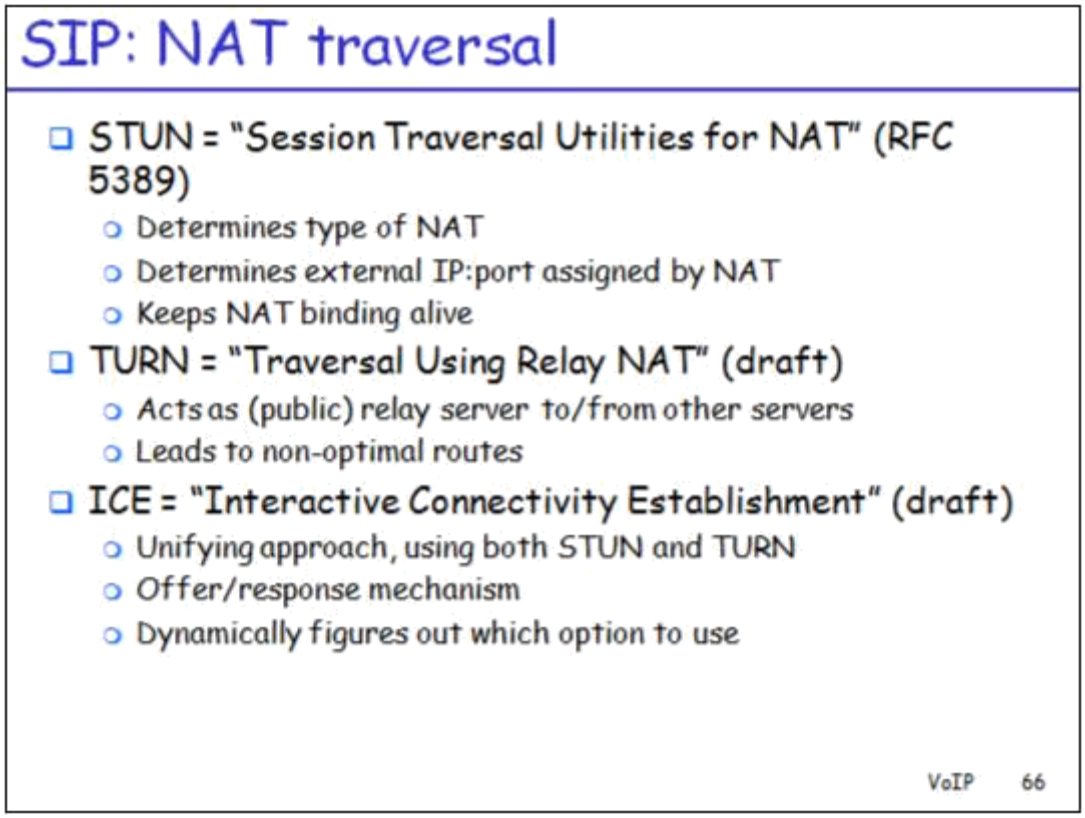
The figure shows a possible solution for the NAT and Firewall problem. A Firewall/N AT control proxy is added (note that this may be integrated with the Firewall/NAT on a single platform).

Alice will send an **INVITE** to Bob via the Firewall/NAT control proxy. This proxy will communicate with the Firewall/N AT (via Bind messages) to inform about the session and know the ports that will be used in the NAT. With this information an (adapted) **INVITE** is sent to Bob. When Bob accepts the call, the Firewall/N AT control proxy will send an Open request to the Firewall in order to open the correct ports for the RTP Media Session stream. After the session, the Firewall/N AT will be informed, removing the NAT binding and closing the appropriate Firewall ports.

Note that the protocol used between the Firewall/Nat control proxy and the Firewall/N AT gateway is not part of SIP (it may be a proprietary protocol).



The example is worked out in some more detail. Note that not all messages are shown and the messages are far from complete (only the interesting information is indicated). The IP source and destination addresses used in the IP packets are indicated in orange (for the RTP Media Session, only the flow from right to left is indicated). The NAT operation is clearly illustrated.



STUN: used to be "Simple Traversal of User Datagram Protoc ol (UDP) Through Network Address Translators (NATs)", RFC 3489

Most recent version of STUN: RFC 5389. It has been defined since the "classic STUN" of RFC 3489 had some flaws: *"The address and port learned through classic STUN are sometimes usable for communications with a peer, and sometimes not. Classic STUN provided no way to discover whether it would, in fact, work or not, and it provided no remedy in cases where it did not. Furthermore, classic STUN's algorithm for classification of NAT types was found to be faulty, as many NATs did not fit cleanly into the types defined there." (from RFC 5389)*

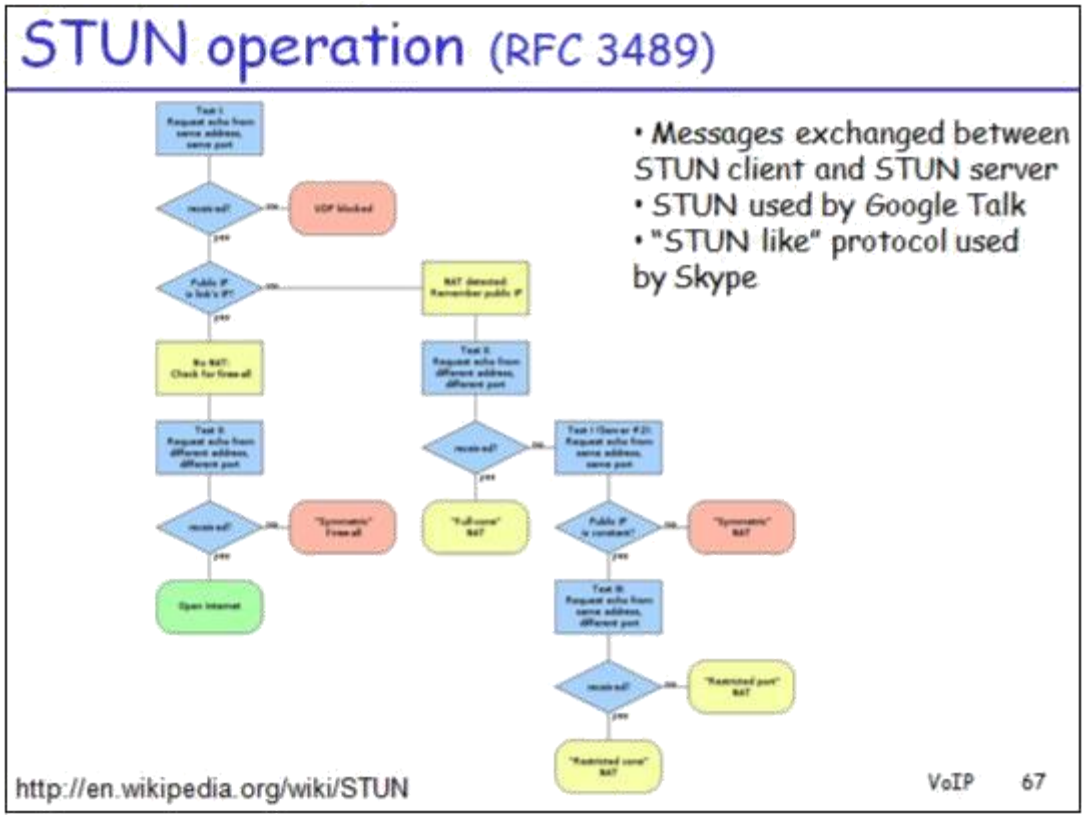
Some info on STUN concepts and the problems it addresses can be found in:

http://www.newport-networks.com/cust-docs/33-NAT- Traversal.pdf

TURN: Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN); draft-ietf-behave-turn-11, 29 Oct. 2008

ICE: "Interactive Connectivity Establishment (ICE): A P rotocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", dr aft-ietf- mmusic- ice-19, 29 Oct. 2007

Specifically for SIP: Managing Client Initiated Connections in the Session Initiation Protocol (SIP), draft-ietf-sip-outbound-16, Oct. 2008 (see http://tools.ietf.org/html/draft-ietf-sip-outbound-16). This addresses the problem of making a client behind a Firewall/NAT reachable for incoming calls, i.e. enable SIP messages from the external network (public internet).



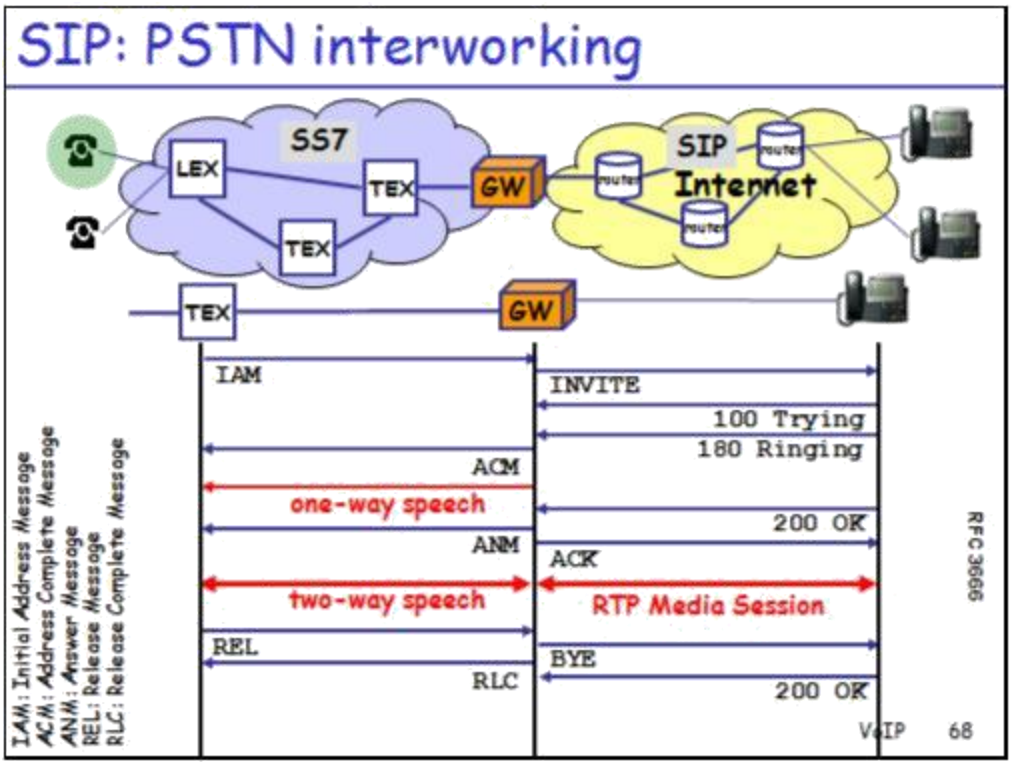
STUN is a simple client-server protocol. A client sends a request to a server, and the server returns a response. There are two types of requests - Binding Requests, sent over UDP, and Shared Secret Requests, sent over TLS over TCP. Shared Secret Requests ask the server to return a temporary username and password. This username and password are used in a subsequent Binding Request and Binding Response, for the purposes of authentication and message integrity.

Binding requests are used to determine the bindings allocated by NATs. The client sends a Binding Request to the server, over UDP. The server examines the source IP address and port of the request, and copies them into a response that is sent back to the client. There are some parameters in the request that allow the client to ask that the response be sent elsewhere, or that the server send the response from a different address and port. There are also attributes for providing message integrity and authentication.

The STUN Binding Request is used to discover the presence of a NAT, and to discover the public IP address and port mappings generated by the NAT. Binding Requests are sent to the STUN server using UDP. When a Binding Request arrives at the STUN server, it may have passed through one or more NATs between the STUN client and the STUN server. As a result, the source address of the request received by the server will be the mapped address created by the NAT closest to the server. The STUN server copies that source IP address and port into a STUN Binding Response, and sends it back to the source IP address and port of the STUN request. For all of the NAT types, this response will arrive at the STUN client.

When the STUN client receives the STUN Binding Response, it compares the IP address and port in the packet with the local IP address and port it bound to when the request was sent. If these do not match, the STUN client is behind one or more NATs (but he does not know what type of NAT). To determine that, the client uses additional STUN Binding Requests. The client would send a second STUN Binding Request, this time to a different IP address, but from the same source IP address and port. If the IP address and port in the response are different from those in the first response, the client knows it is behind a symmetric NAT. To determine if it's behind a full-cone NAT, the client can send a STUN Binding Request with flags that tell the STUN server to send a response from a different IP address and port than the request was received on. In other words, if the client sent a Binding Request to IP address/port A/B using a source IP address/port of X/Y, the STUN server would send the Binding Response to X/Y using source IP address/port C/D. If the client receives this response, it knows it is behind a full cone NAT.

STUN also allows the client to ask the server to send the Binding Response from the same IP address the request was received on, but with a different port. This can be used to detect whether the client is behind a port restricted cone NAT or just a restricted cone NAT.

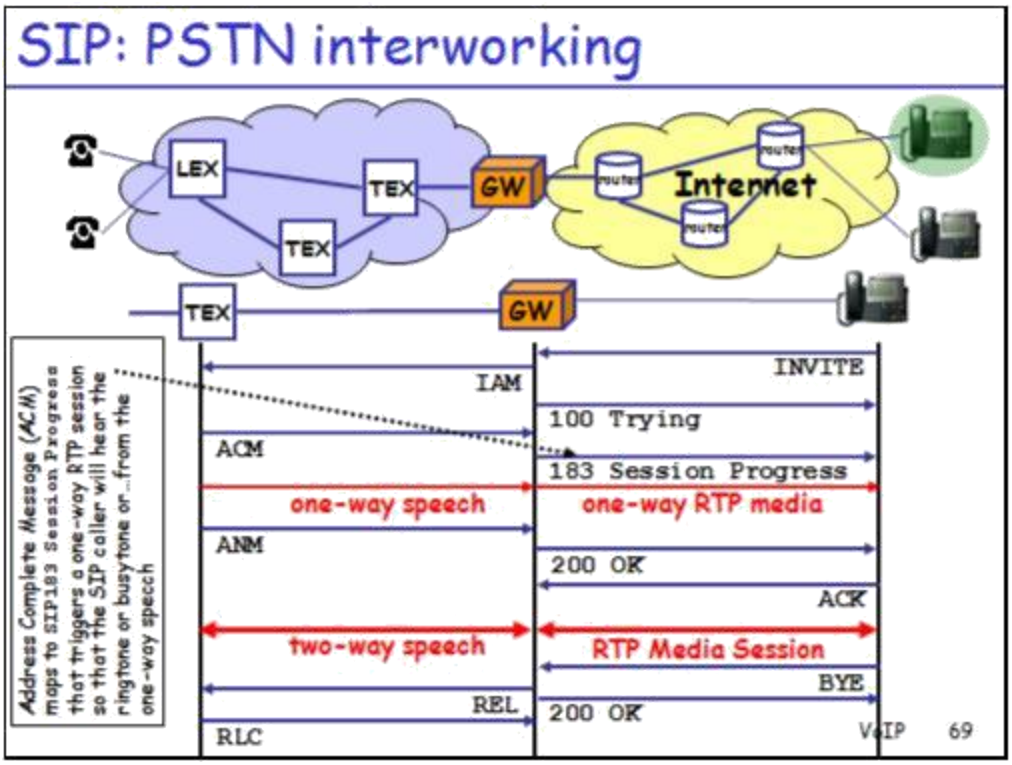


This example shows the interworking between a PSTN and a VoIP network. A call is initiated at the PSTN phone using ISUP messages (SS7). The GW (Gateway, combining the functions of a Signaling GW and Media GW) will do the mapping of the signaling messages and will also provide the transcoding of the voice signal (e.g. G.711 G.729). Note that the mapping of the signaling messages will not be a one-to-one mapping (in general, SIP has richer signaling capabilities) meaning that some messages will stop at the GW.

After the ACM (Address Complete Message) has been sent to the TEX (Transit Exchange), a one-way speech channel is set-up. This is typical for a phone network and it is used to support in-band signaling (=signaling in the bandwidth used for the speech signal), e.g. sending the ringing tone from the called party to the caller.

The voice connection will be established when the two-way speech connection and the RTP media session are set-up.

More information and detailed examples in RFC 3666.



This example shows a call initiated at the SIP phone.

An important difference with the previous example occurs when the ACM arrives in the GW. The ACM message does not trigger the 180 Ringing message because in the PSTN network, the ringing signal will be sent over the one-way speech channel (in-band signal). This ringing tone (on the one-way speech channel) should be transmitted on a (one-way) RTP media session (the SIP client will listen to the incoming RTP stream and hear the ringing signal). In order to alert the SIP client that he should listen to the RTP session, the **183 Session Progress** signal is sent (this will typically include an SDP description).

IAM: Initial Address Message

ACM: Adress Complete Message

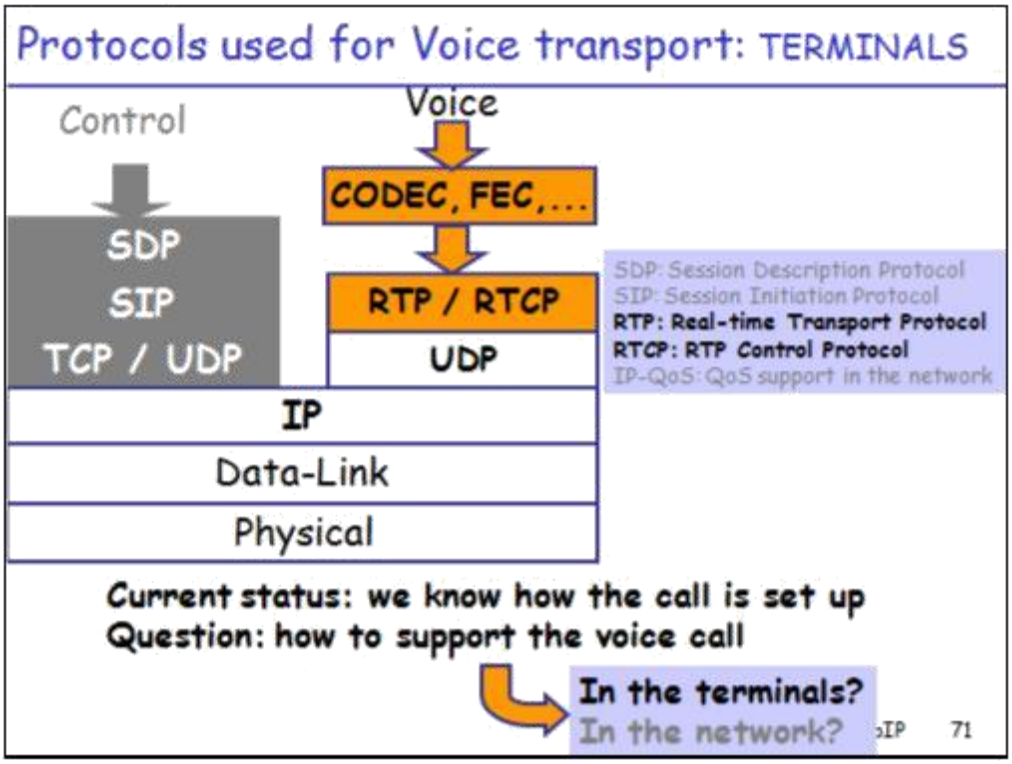
REL: Release

RLC: Release Complete

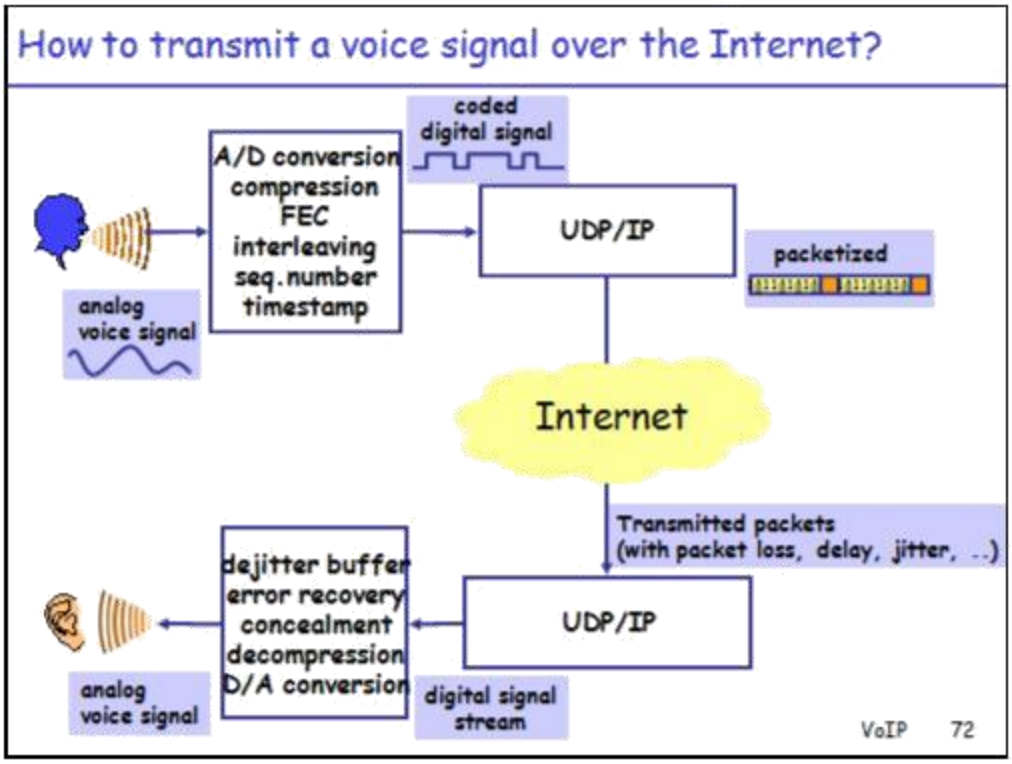
Note: detailed info on SIP-PSTN call flows can be found in RFC 3666.

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
4. Multimedia over IP



In the terminals we have several methods and protocols to support voice over IP. When transmitting voice over internet, one will compress the voice signal (in a codec) in order to reduce the necessary bandwidth, but also other treatments will be done in order to support the transfer of the voice signal with acceptable quality. Examples are: add redundant information for packet loss recovery (e.g. Forward Error Correction), insert timing information (e.g. using RTP), etc.



Let us first have a look at how the analog signal is transferred over an internet based network.

At the sender side the analog voice signal will be converted to a digital signal (sampling in time and amplitude in an A/D converter) and eventually a compression algorithm will be used to reduce the bitrate. In order to reduce the influence of packet loss in the network, error correcting measures may be taken at the sender side. Examples are Forward Error Correction (FEC), interleaving and the addition of sequence numbers. The addition of timing information (timestamp) may be useful in order to eliminate jitter (=delay variation) or to detect silent periods in the conversation. This will result in a digitally coded signal that will be transported over the UDP/IP protocol stack.

The receiver will use a dejitter buffer in order to reduce the influence of the jitter in the network (possibly also using the timestamp information). Error recovery will be based on the FEC information from the sender possibly combined with the interleaving. The influence of lost packets may be reduced by the use of packet concealment: a missing packet may be replaced by e.g. noise or an interpolation. Decompression and D/A conversion will generate the voice signal, which will be a degraded version of the original voice signal.

Overview: Terminal Actions

NETWORK problems:
- packet loss
- delay
- delay variation (jitter)
- bandwidth limitation

SENDER side solutions:
- A/D conversion (see PSTN)
- compression/decompression (Coding/Decoding: CODEC)
- FEC (Forward Error Correction)
- interleaving
- sequence number and time stamp (RTP)

RECEIVER side solutions:
- dejitter buffer
- echo cancellation (see PSTN)
- packet concealment

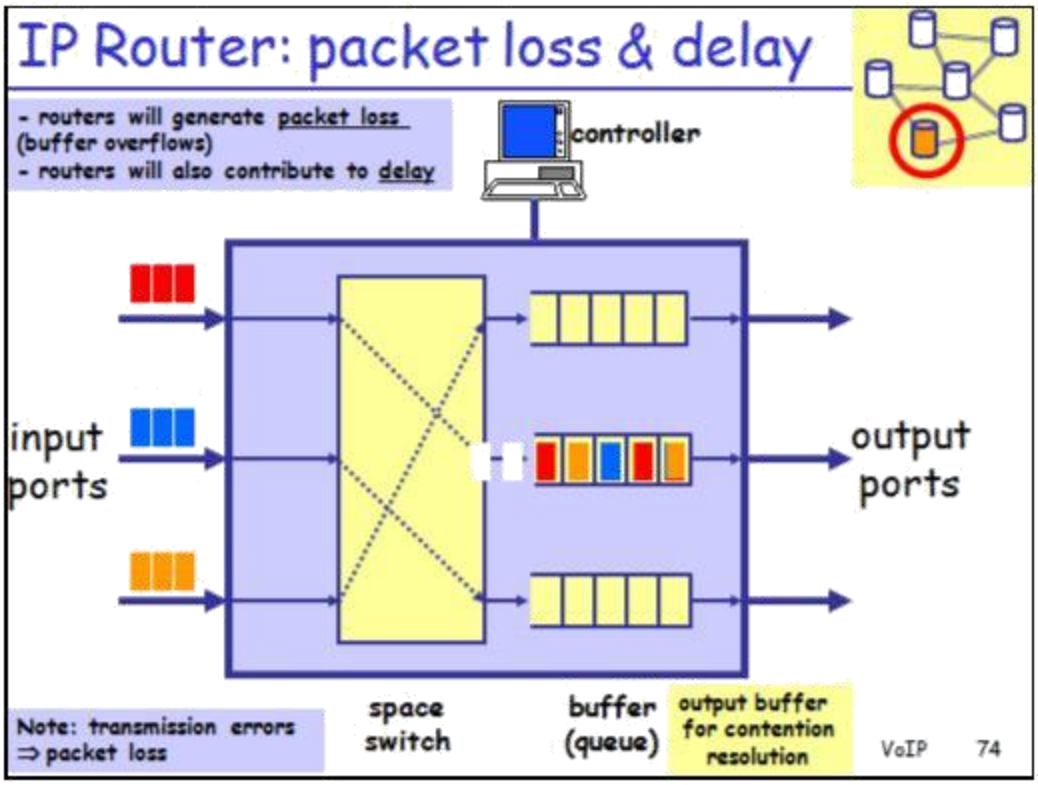
VoIP 73

A first part will study the problems encountered in an IP based network. The most important problems are packet loss, delay, delay variation (jitter) and bandwidth limitation. This is due to the best effort behavior of the internet protocol.

A next part will discuss some important measures taken at the sender side to overcome the internet limitations: compression/decompression (in a codec) in order to reduce the required bandwidth, forward error correction (FEC) in order to allow reconstruction of the signal when parts are missing, interleaving of the information in order to reduce the influence of burst errors (= group of consecutive packets that are lost), addition of a sequence number in order to detect missing or out-of-order packets, addition of time stamps in order to detect silence periods or to reduce the jitter.

A third part will discuss in more detail the receiver side measures: dejitter buffer, echo cancellation (already discussed in PSTN part) and packet concealment.

There is a last part (not shown in the overview) that will give some examples of the influence of the network impairments on the perceived voice quality (MOS = Mean Opinion Score).



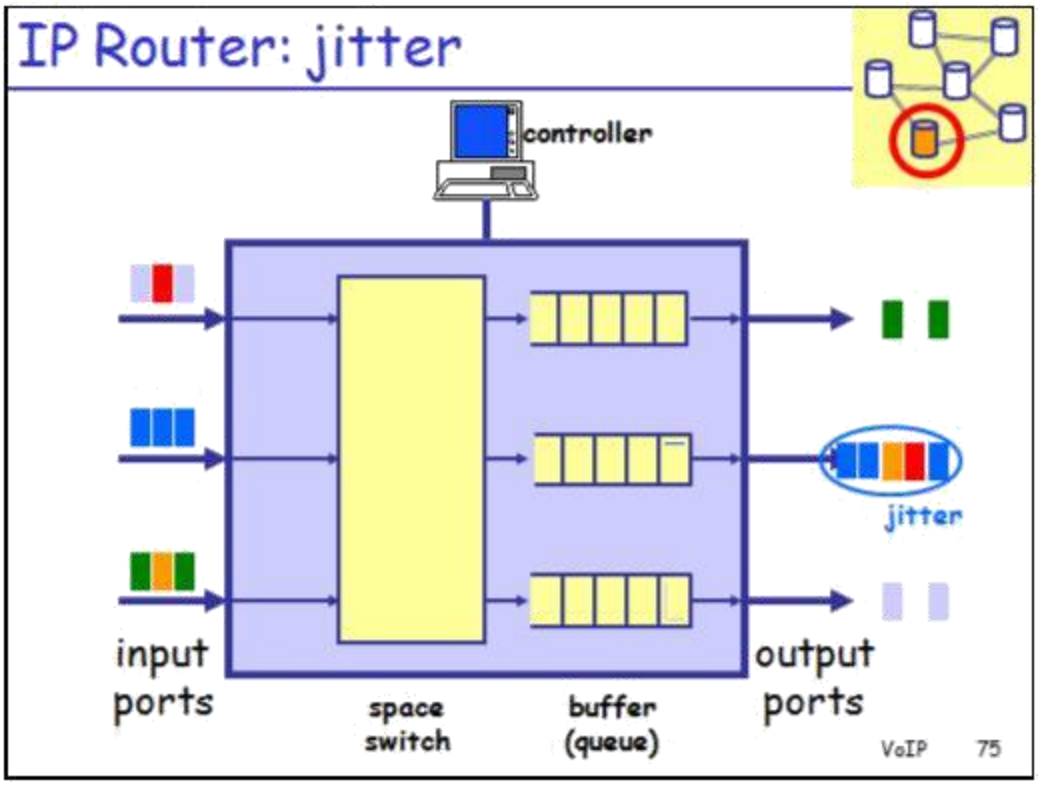
The major contributor to packet loss is the router in the network. The basic functionality of a router is:

- Space switching to transfer a packet from a certain input port (interface) to a certain output port (interface).
- Buffering (queuing) for contention resolution: multiple packets arriving at the same time may contend for the same output line. Buffering is also required to provide sufficient time for routing table look-up (longest prefix match).
- Control will be responsible for maintaining the routing tables (e.g. use of OSPF routing protocol), controlling the space switch, scheduling the buffers, etc.

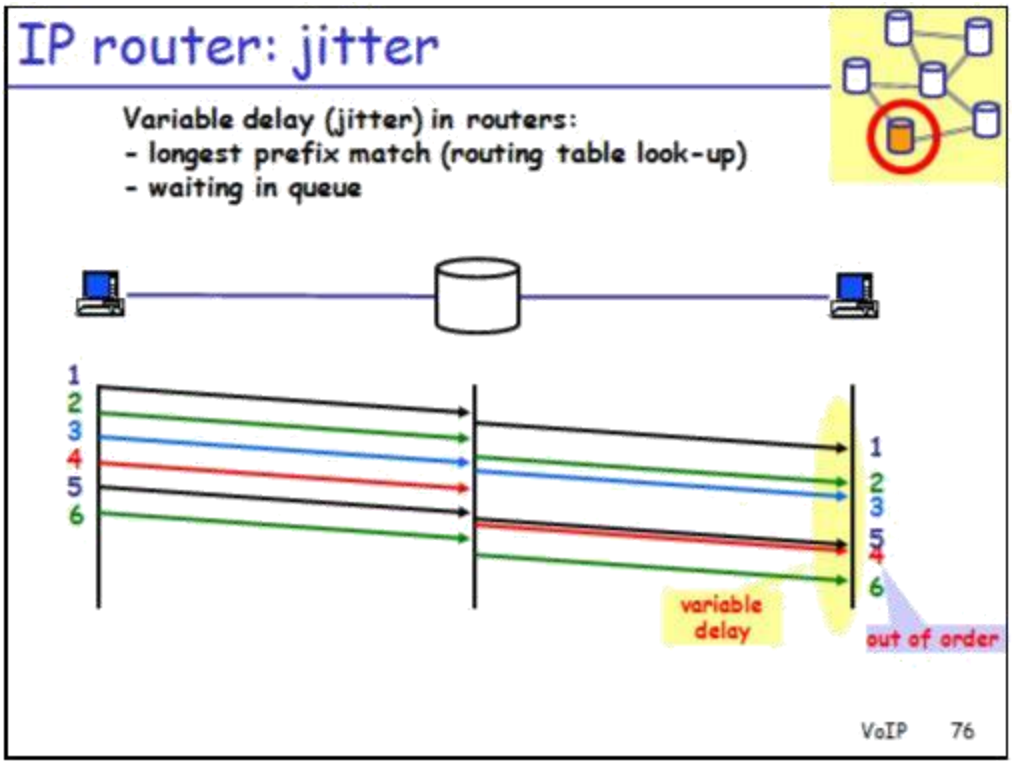
The figure illustrates several packets arriving on different input ports, contending for the same (middle) output port. This will result in a delay for certain packets.When the buffer gets full, overflow will occur resulting in packet loss.

Another reason for packet loss is due to transmission errors (not shown on the figure). Transmission errors will result in bit errors (or bursts of bits) but this will result in packet loss (wrong bits will be detected by the checksums used in TCP/UDP and IP headers).

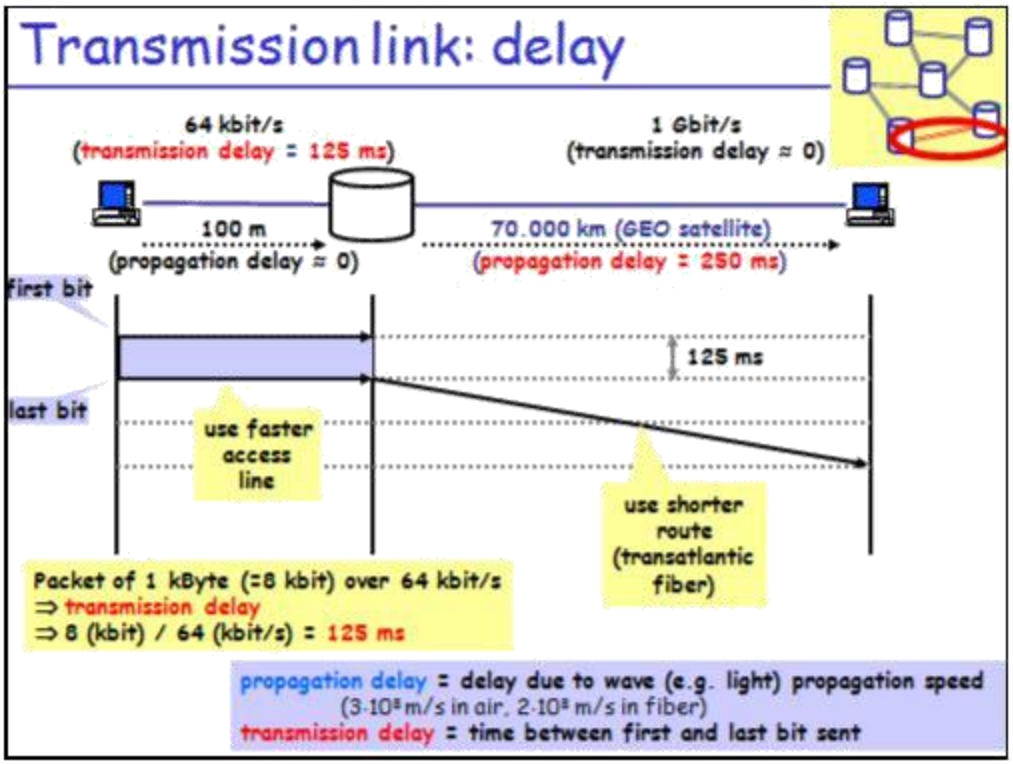
Note: this is a simplified scheme (e.g. when a packet arrives, some time will be required to know the outlet port by using the routing table and doing a longest prefix match)



Routers will also be responsible for the delay variations (jitter) observed when crossing a network. Because the buffers have to resolve the contention, it may happen that packets from the same traffic flow may have to wait more or less in the buffers (depending on the other traffic contending for the same output link).



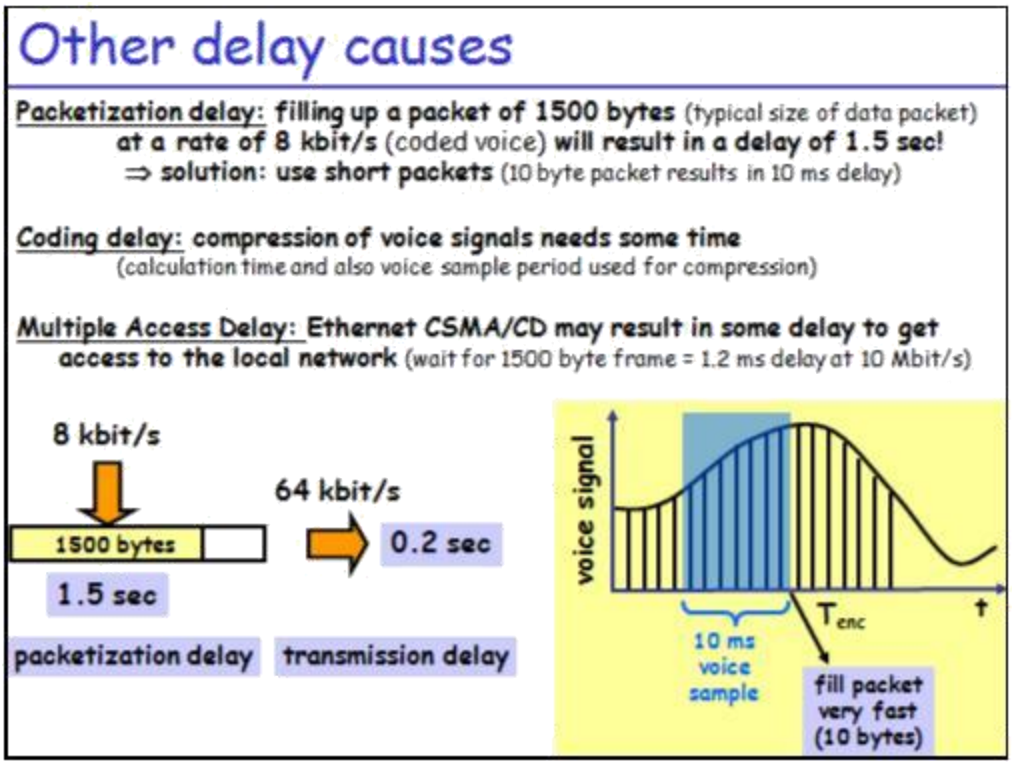
The delay variation in a router will result in end-to-end delay variations and even out of order arrival becomes possible. This is illustrated in the figure: 6 packets from the same traffic flow are transmitted at regular time intervals. Due to delay variations in a router, they will arrive with a time difference between the packets which will vary (this is the jitter).



A first problem is the delay in the network. This delay will impact the influence of echo (if the delay is less than a few 10 ms, it has no impact in the degradation quality) and will reduce the interactivity of the conversation (if the delay is a few 100 ms, interactivity is still acceptable but starts to degrade).

A first important delay component is the transmission delay. This is the time it takes to transmit a packet over a transmission line with a certain bandwidth. The example shows that it takes 125 ms to transmit a packet of 1 kbyte over an access line of 64 kbit/s (e.g. ISDN line): this is the time between the first bit and the last bit sent. If one would use an access line of 5 Mbit/s (e.g. possible with a cable modem), the transmission delay would reduce to 1.6 ms.

A second important delay component is the propagation delay, limited by the propagation speed of electromagnetic waves: the speed of light is about 3×10^8 ms in air and 2×10^8 in optical fiber. This component becomes important for transmission over long distances. The example shows a transatlantic connection via a Geo Stationary Satellite (GEO) which is at a distance of about 35000 kilometer from earth. This results in a propagation delay of about 250 ms (from earth to satellite and back to earth). The way to reduce this delay is to use shorter distances. In the case of the use of satellite links, one can replace them eventually by fiber links (e.g. transatlantic cable), resulting in a shorter propagation distance and therefore a lower delay value (e.g. 10000 km of fiber results in 50 ms delay).

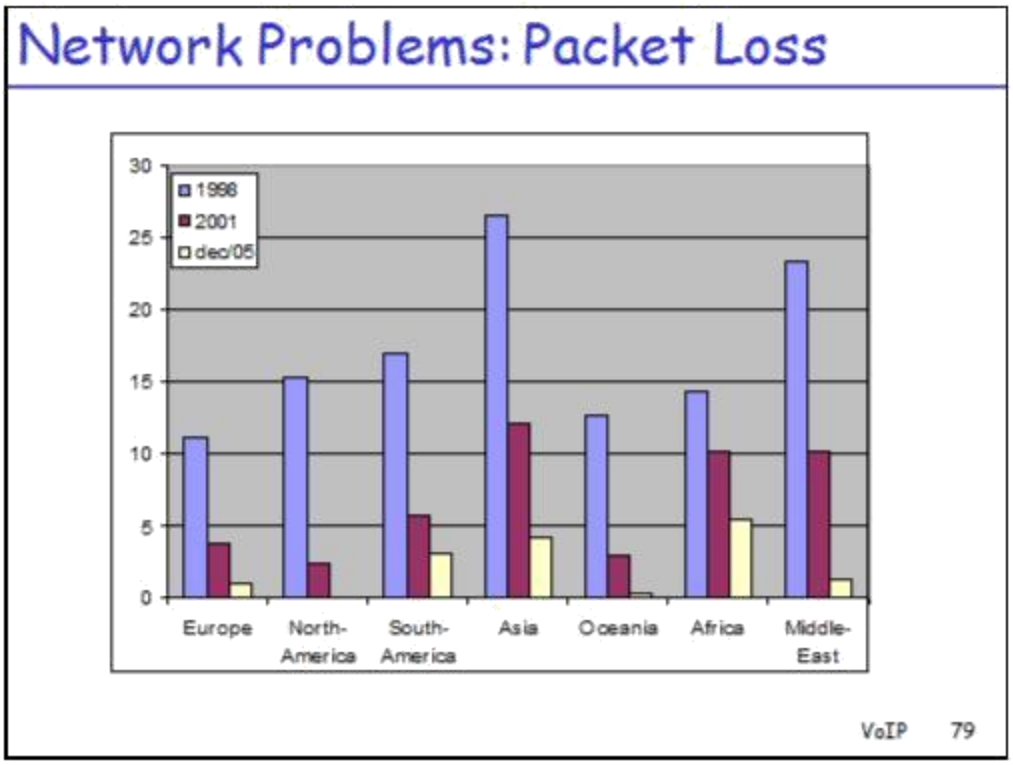


There are a number of other causes for the delay. A first important delay component is the packetization delay. This will depend on the packet size used and on the fill rate of the packet. Suppose one would use an IP packet of (about) 1500 bytes (typically used for data packets) that is filled up at a rate of 8 kbit/s (example of a codec rate). This would result in about 1.5 seconds before the packet would be filled. Note that the packetization delay is different from the transmission delay, as illustrated in the figure (left). Once the packet is filled with information it will be sent over the outgoing line interface (at the speed of that link). The example shows a 64 kbit/s line speed. In order to reduce the packetization delay, one may use much smaller packet sizes. The typical packet size used with the 8 kbit/s codec is 10 bytes resulting in a packetization delay of 10 ms.

A second source of delay is the coding delay. This will correspond to the voice sample duration (e.g. 10 ms) and some additional time required for the calculation (eventually there is an additional look-ahead delay, see part on codecs). Note that the coding delay will be combined/overlapped with the packetization delay: one may take a voice sample of 10 ms and do a fast coding. The filling of the packet may also be much faster than the average codec rate (e.g. 8 kbit/s) because at the end of the calculation all bytes (e.g. 10 bytes) are available to fill the packet immediately. This will result in a overall coding + packetization delay of (10 ms + coding calculation time).

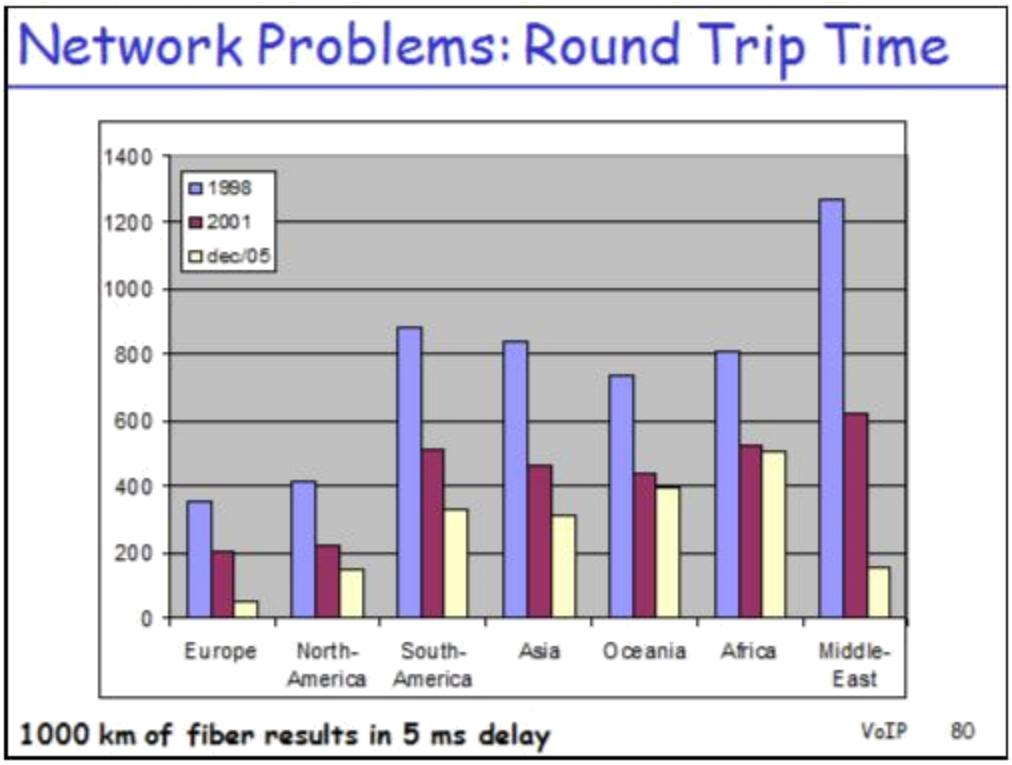
A last example of a source of delay is the access delay to a shared medium (e.g. Ethernet). Due to the shared use of the medium, it is possible that one has to wait for other sources sending on the shared medium (CSMA/CD). Example: suppose one has to wait for an Ethernet frame of 1500 bytes on a 10 Mbit/s segment. In this case the delay will be in the order of 1.2 ms. In case of heavy traffic on the Ethernet segment, this could introduce more delay.

Note: the overall delay from speaker to listener is sometimes called the M2E delay (Mouth to Ear).

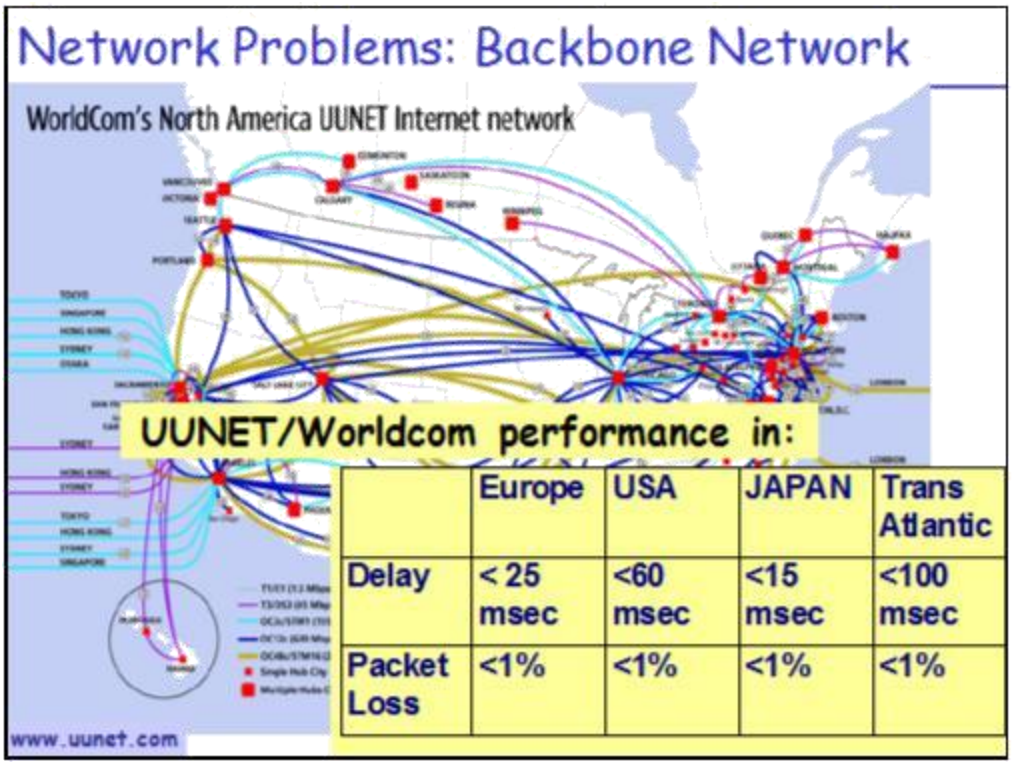


This figure shows some measurements done on the Internet from a location. The protocol used was ICMP with the ping tool. A large number of servers all over the world were “pinged” and the roundtrip delay (see other figure) and packet loss (=num ber of lost ping messages) were measured. We observe measurements done in 1998 and in 2001. A drastic improvement is observed. This is partially due to a better uplink but also because the backbone networks improved considerably. Also recent numbers were added showing the drastic improvement over recent years.

Note the still relatively high levels of packet loss (ranging from a 0.5 up to 5%)!



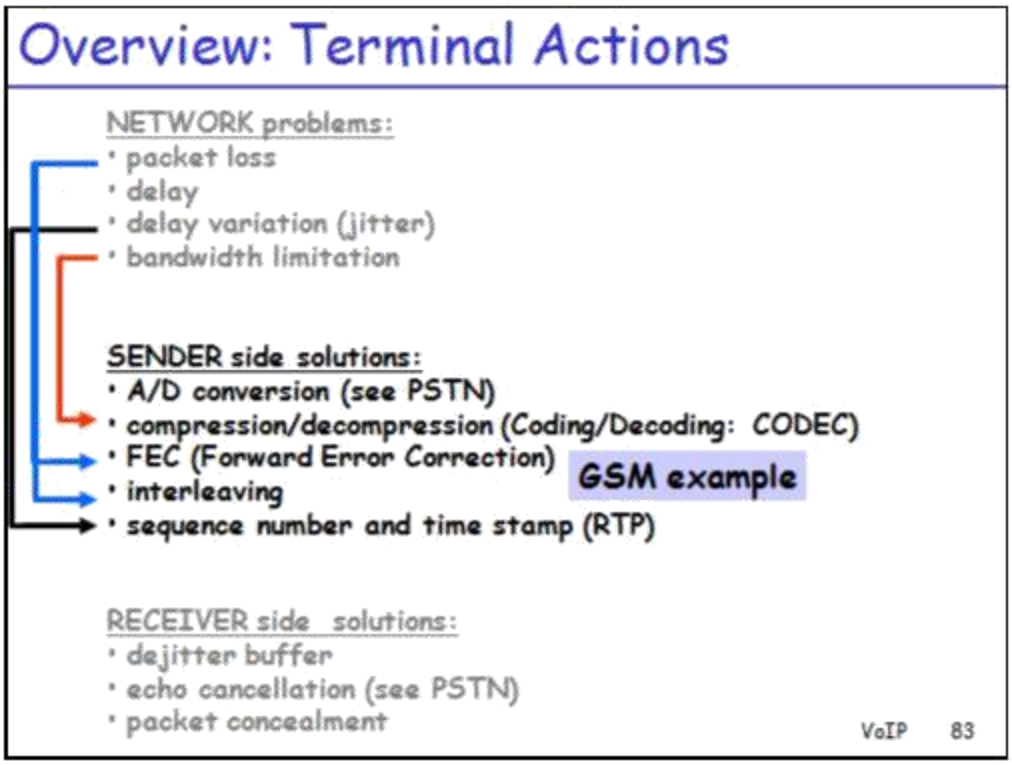
Here we observe the roundtrip delay (= 2 times the delay from source to destination!). Again high values are observed (range of 50 to 500 ms).



An example of the network behavior of a single (large) ISP (Internet Service Provider) is illustrated for UUNet (Worldcom) [2002]. It is observed that the delays can be kept reasonable (large part is due to propagation delay).

Content

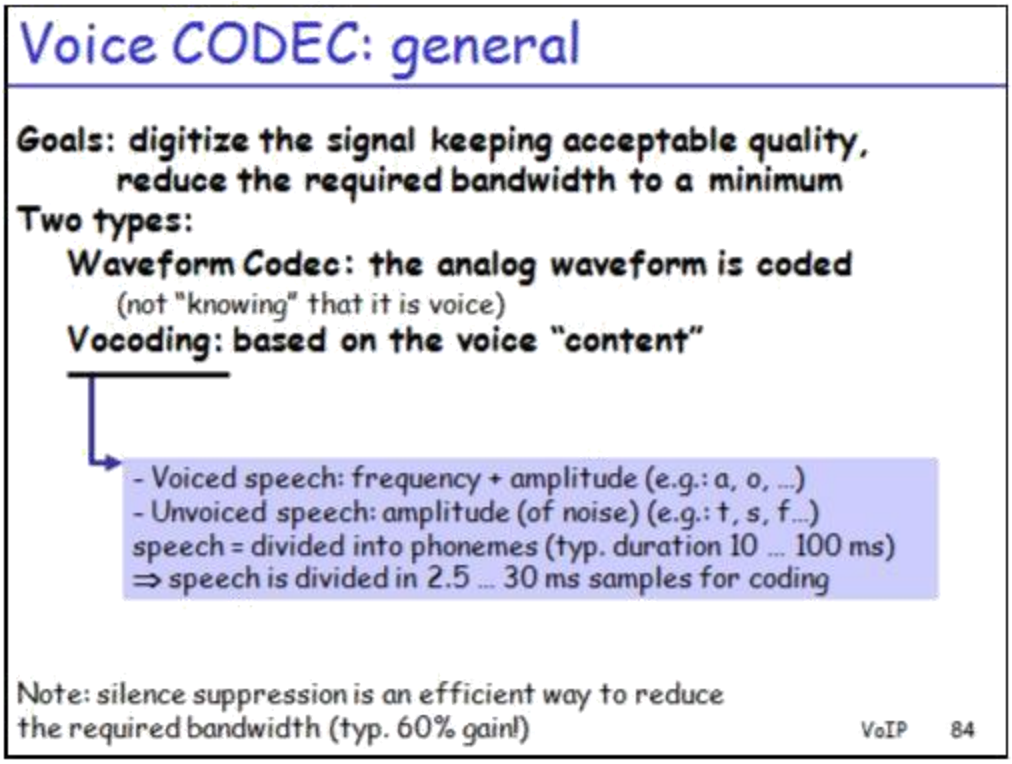
1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     - Sender side solutions
     - Receiver side solutions
     - Influence on voice quality
4. Multimedia over IP



In order to solve the problems encountered in the network, solutions are discussed that may be implemented in the terminals (sender and receiver).

At the sender side we observe:

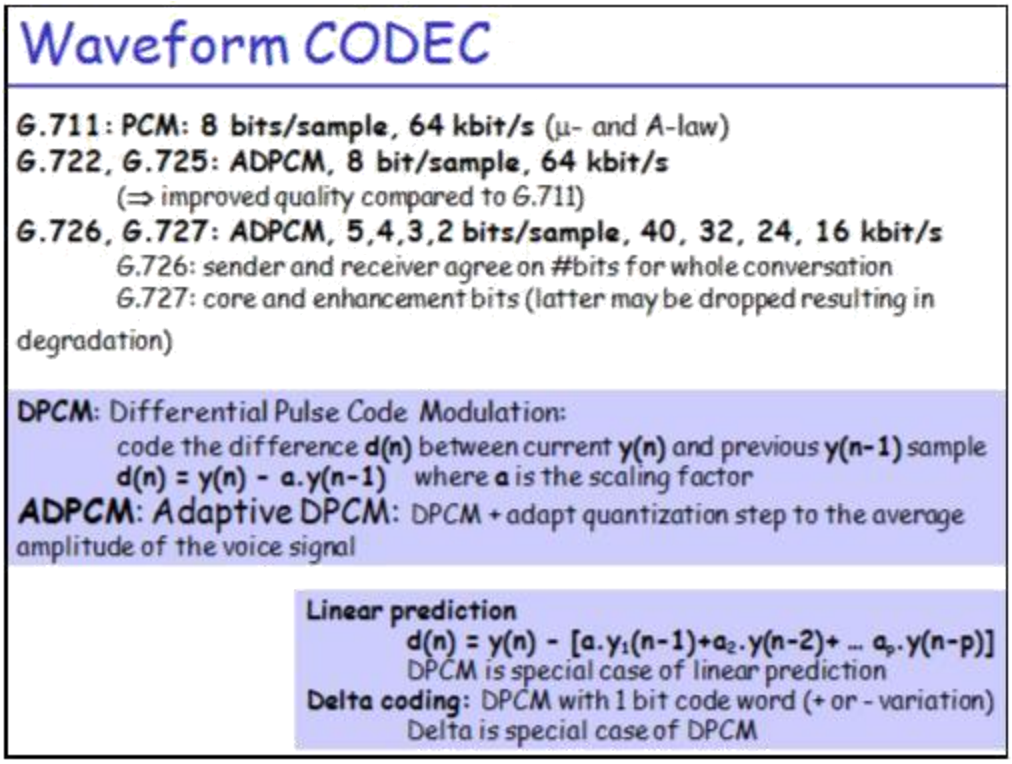
- compression to reduce the required bandwidth
- FEC and interleaving to act against packet loss
- use timestamp to counteract delay variation



A coder will be used to reduce the bandwidth of the generated digital speech signal, keeping an acceptable quality. In general one talks about a CODEC = COder + DECoder (normally a sender will also act as a receiver, requiring indeed also decoding functionality).

There are two important types of codecs: the waveform codec and the vocoder. A waveform codec will treat the voice signal as an analog signal, not knowing it is specifically a voice signal. It will use generic techniques for signal compression. A vocoder on the other hand will use some specific information related to the speech characteristics. There will be voiced and unvoiced speech. The speech will be divided in phonemes (typ. duration 10 to 100 ms). Speech samples used for coding will be between 2.5 and 30 ms.

Note: In general the voice activity (in one direction) is only 40% of the total connection time. This is due to the fact that one is talking roughly only 50% of the time and in addition, when one is talking there are always some pauses accounting for an additional 10% of silence. This could also be effectively used to reduce the overall bitrate. This is however not taken into account in the further discussion.



The most important waveform codec is the G.711 standard: Pulse Code Modulation (PCM). This codec is used today for the digital transmission of voice signals in the PSTN network. There are two versions, one for Europe (A- law) and one for the US (μ- law). The difference is due to a different logarithmic A/D conversion (see also PSTN part). This codec results in a bitstream of 64 kbit/s or 8 bit samples every 125 μσ.

Another type of waveform codec is the Adaptive Differential PCM codec. There are different ADPCM codecs: G.722 and G.725 are using 8 bits (resulting in a bitrate of 64 kbit/s) and the G.726 and G.727 are using 5,4,3 or 2 bits for coding (resulting in 40, 32, 24 or 16 kbit/s bitrate). The voice quality of the G.722 and G.725 is better than that of the G.711.

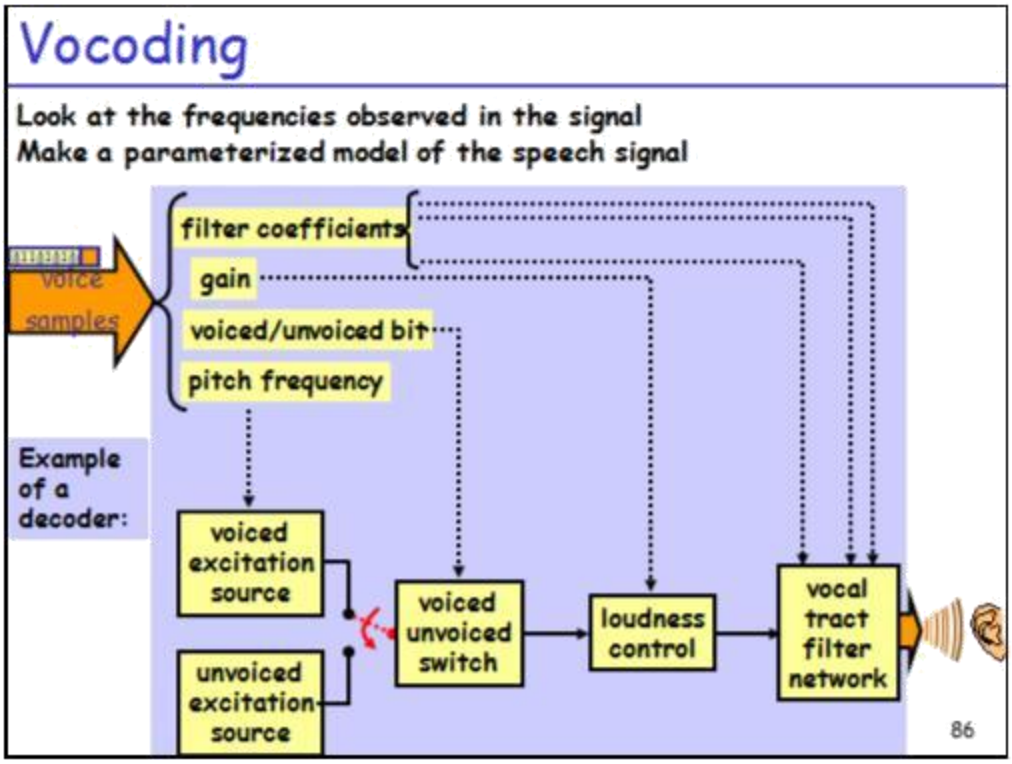
The fact that there are differential PCM codecs accounts for the fact that differences between the current voice sample and previous voice samples is coded. These codecs will use analysis algorithms that make use of the technique of dynamically adapting the quantizer step size in response to variations in input signal amplitude.

A more advanced DPCM codec is using linear prediction: a new sample value is predicted (based on p previous samples) and the difference between this prediction and the actual value y(n) is coded.

A very simple DPCM coding is delta coding where only one bit is used to indicate if the signal amplitude is increasing or decreasing.

Note: A large number of codecs have been standardized by ITU-T (International Telecommunication Union - Telecommunications) [see also www.itu.ch].

Note: All waveform codecs are producing digital voice samples every 125 μσ. The sample size depends on the codec used (8, 5,4,3 or 2 bits).



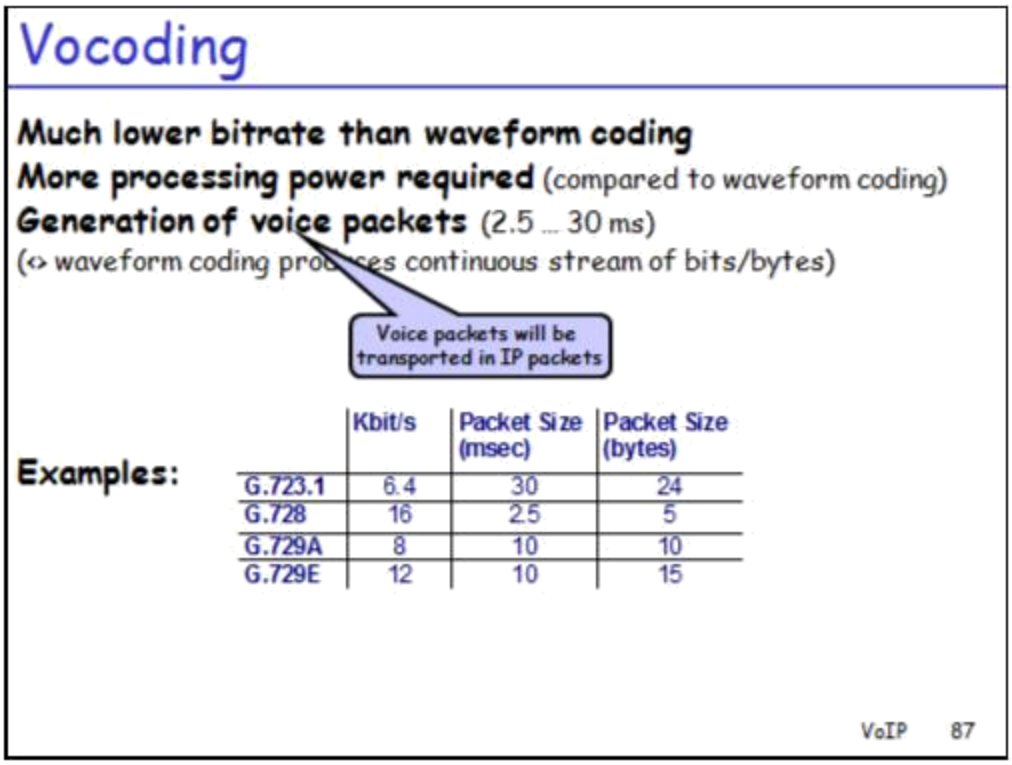
Unlike waveform coding, vocoding is based less on the analog waveform than on the human vocal tract. It divides speech into voiced and unvoiced speech.

• Voiced speech resonates at a frequency that is characteristic for the human vocal tract, and is coded as the frequency plus the amplitude.

• Unvoiced speech is typically a consonant (such as T, S, K) and is represented as an amplitude without any specific frequency (noise).

In this way speech is divided into phonemes which are the basic building blocks of speech (example: the word DOT contains 3 phonemes: an unvoiced D, a voiced O and an unvoiced T). Phonemes may last 10 to 100 ms, depending on the word spoken. Speech is therefore divided into 2.5 to 30 ms samples so that the voiced and unvoiced parts can be identified. Vocoding achieves a significantly lower bitrate than waveform coding and requires more processing power in the codec.

[Note: determination of unvoiced: lower power, more zero crossings]

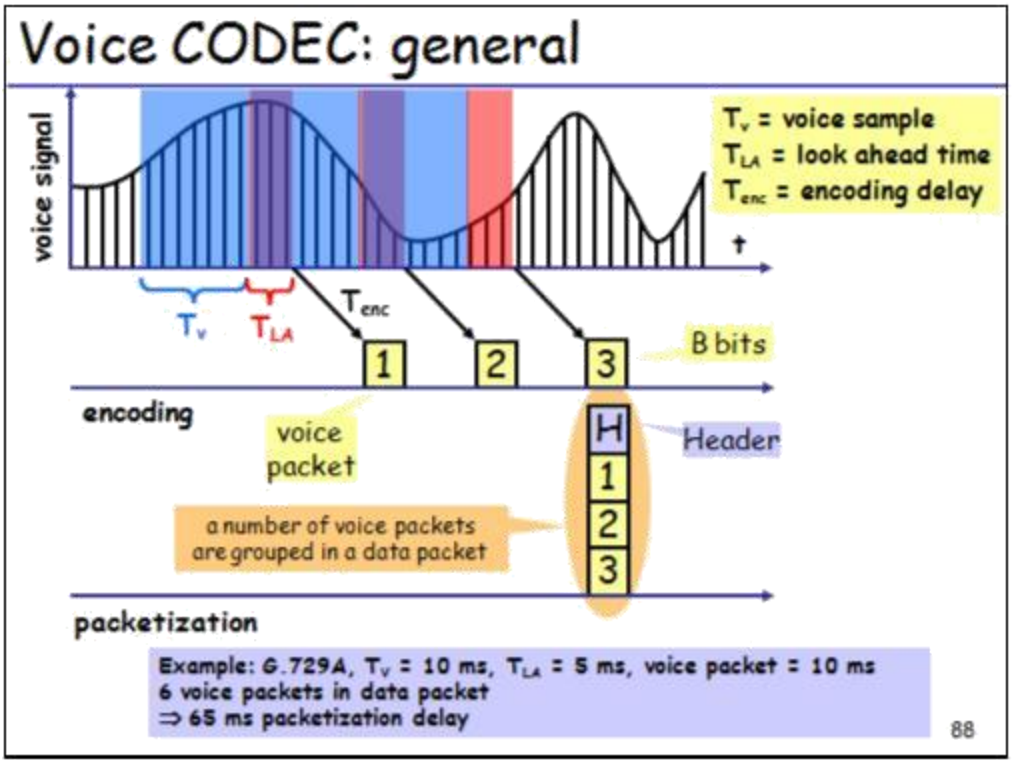


A few examples of vocoders are illustrated.

The G.723.1 codec is using a bitrate of 5.3 or 6.4 kbit/s with a voice packet size of 30 ms containing 20 or 24 bytes of information.

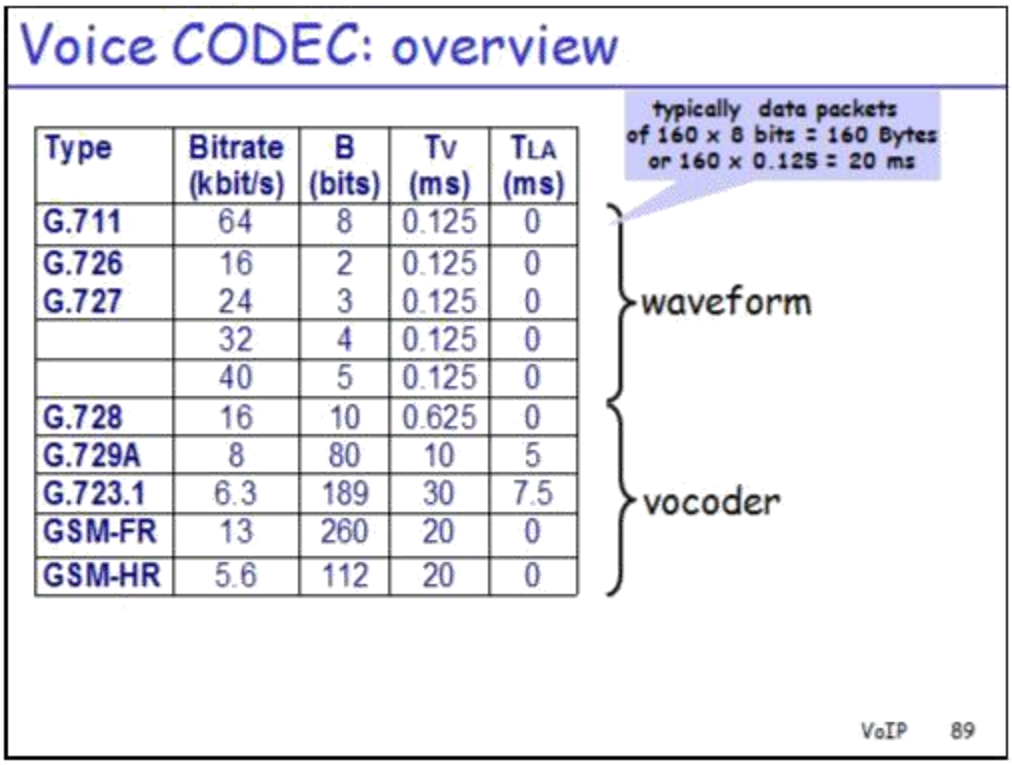
The G.828 is generating bitrates of 9.6, 12.8 or 16 kbit/s with a voice packet size of 2.5 ms containing 3,4 or 5 bytes of information.

The G.729A (E) is running at 8 (12) kbit/s and produces voice samples of 10 (15) bytes every 10 ms.



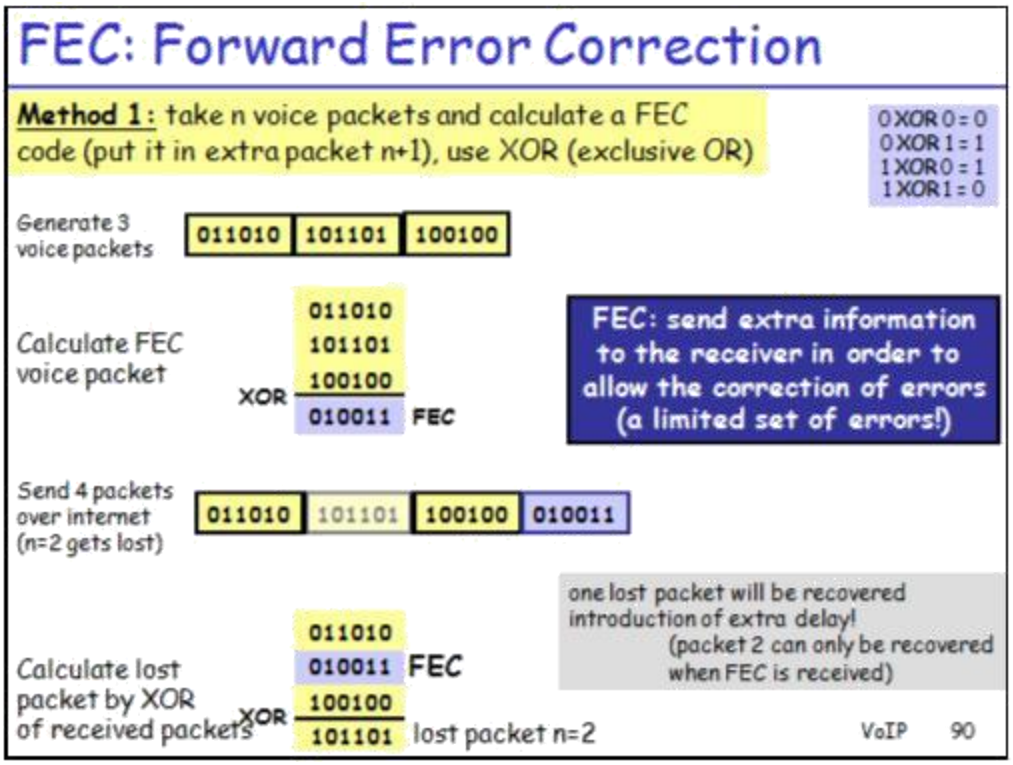
The figure shows the general operation of a voice codec. The analog voice signal will be split in voice samples (duration T_V) and each of these voice samples will be coded. In some cases one will also look ahead in order to code the voice sample. This look-ahead time is indicated by T_{LA}. It is only after the T_V and T_{LA} that the coding can start (at that time all the information is available to generate a voice packet). The time required for the encoding is T_{enc}. An IP packet may be constructed of a number of voice packets (in the example one is using 3 voice packets in one IP packet).

The example illustrates how long it takes to construct an IP packet for a G.729A codec. The voice samples are 10 ms long and the look-ahead time is 5 ms. If we group 6 voice packets in an IP packet, this results in a packetization time (delay) of 65 ms (note that this takes into account one look-ahead time).



Type	Bitrate (kbit/s)	B (bits)	Tv (ms)	TLA (ms)
G.711	64	8	0.125	0
G.726	16	2	0.125	0
G.727	24	3	0.125	0
	32	4	0.125	0
	40	5	0.125	0
G.728	16	10	0.625	0
G.729A	8	80	10	5
G.723.1	6.3	189	30	7.5
GSM-FR	13	260	20	0
GSM-HR	5.6	112	20	0

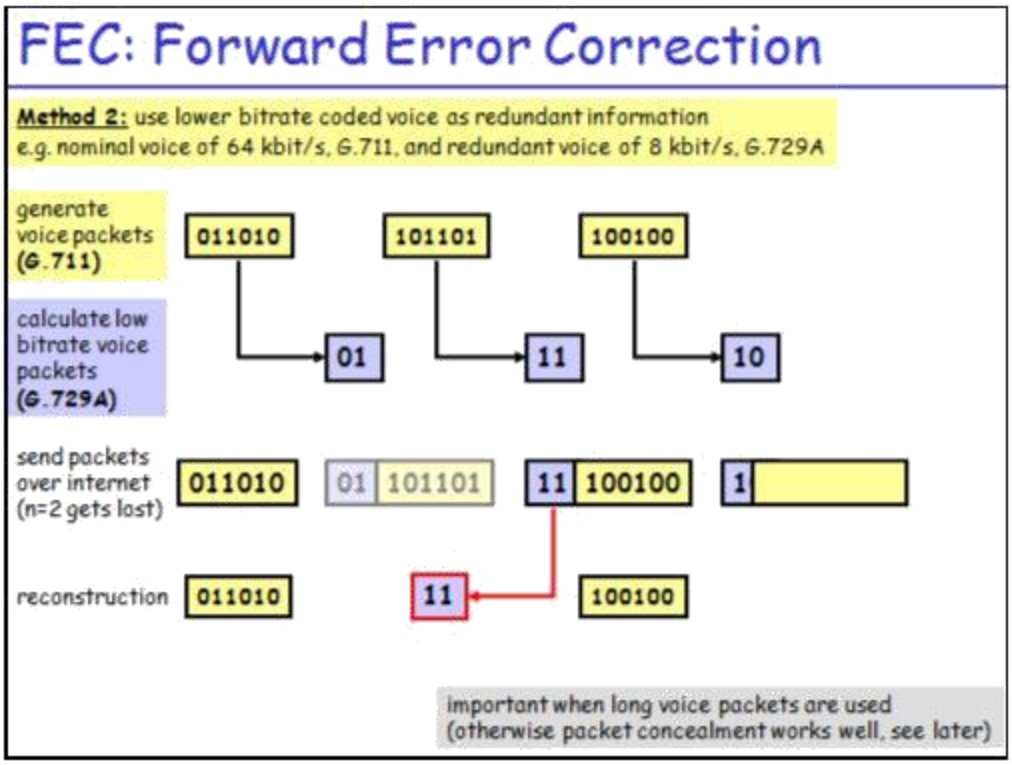
The table shows a number of codecs with their specific characteristics. The two lower ones are used in GSM (FR = Full Rate and HR = Half Rate).



Besides the coding, a sending terminal may also implement some error correcting measures (FEC = Forward Error Correction). Note that this may be considered as a part of the CODEC.

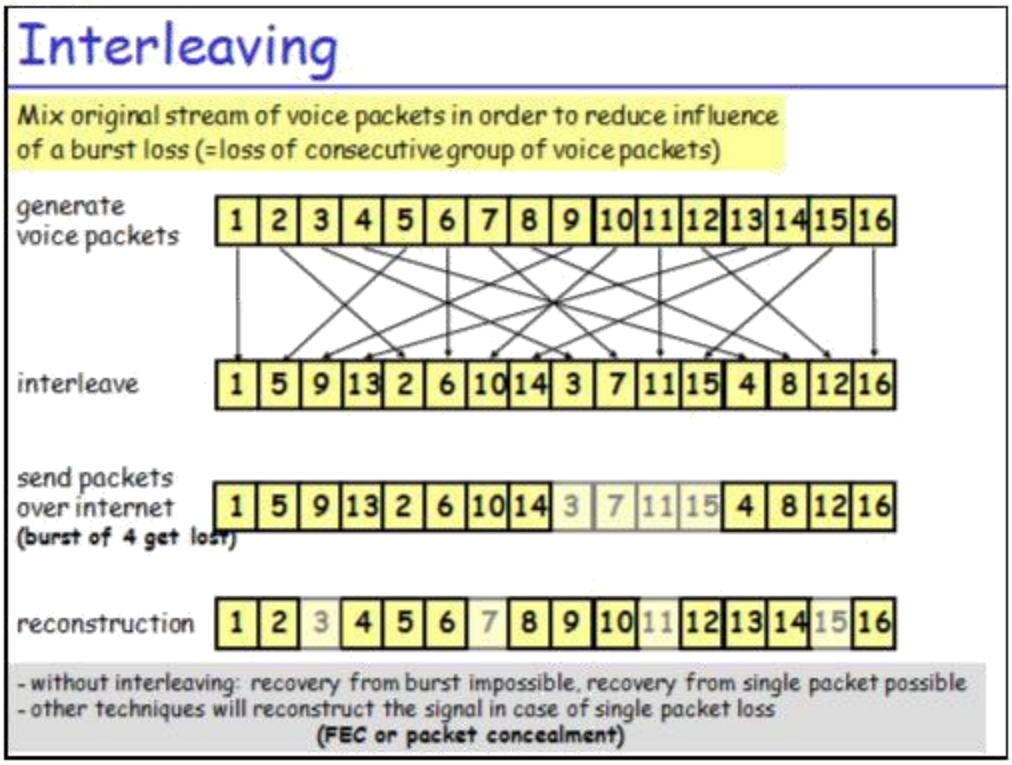
In a first method one will add some extra information to the outgoing datastream. One takes n voice packets and calculates one additional “FEC voice packet” by a XOR of the n voice packets. The n+1 voice packets will be transmitted to the receiver. If a voice packet is lost, the receiver will be able to recover it by replacing the missing voice packet by the “FEC voice packet”, again doing a XO R on these n voice packets. This is illustrated in the example.

Note that FEC will introduce some extra delay at the receiver side because recovery of a voice packet i (with i a value from 1 to n) will only be possible if the “FEC voice packet” is received. Recovery will be possible after rece ipt of the voice packets i, i+1, i+2, …, n.



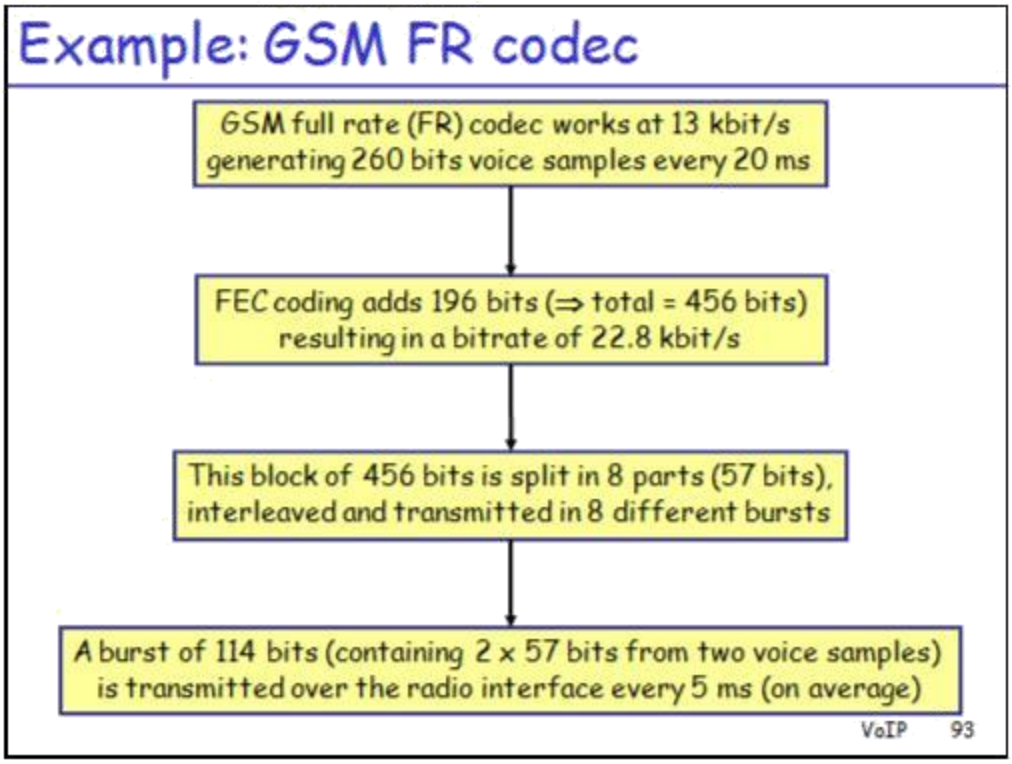
A second possibility of forward error correction (FEC) is to generate two compressed voice packet streams. The low bitrate (corresponding to lower quality) compressed voice stream will be used only in case the high bitrate codec voice packets are missing. The example shows a G.711 codec where G.729A voice packets are added for redundancy.

It is important to note that **FEC is in particular important whe n long voice packets** are used because the more easy packet concealment (see later) is not possible in that case.



A very severe problem that may drastically reduce the voice quality is the loss of a burst of voice packets. A burst is a group of consecutive packets (in the example four packets are lost). In order to reduce the influence of the loss of a burst of packets, interleaving may be used at the sender side.

Interleaving will mix the original voice packets. The figure shows a number of voice packets that are generated by the codec (from 1 to 16). Before sending them in the network, they will be reshuffled. The new order is now 1,5,9,13,2,6,10,14,3,7,11,15,4,8,12,16. The packets are sent over the network and a burst error occurs (e.g. voice packet 3,7,11,15 are lost). After putting back the samples in the original order, one observes that of course still 4 voice packets are missing but they are not consecutive. In this way they are much easier to recover (e.g. using FEC or packet concealment).

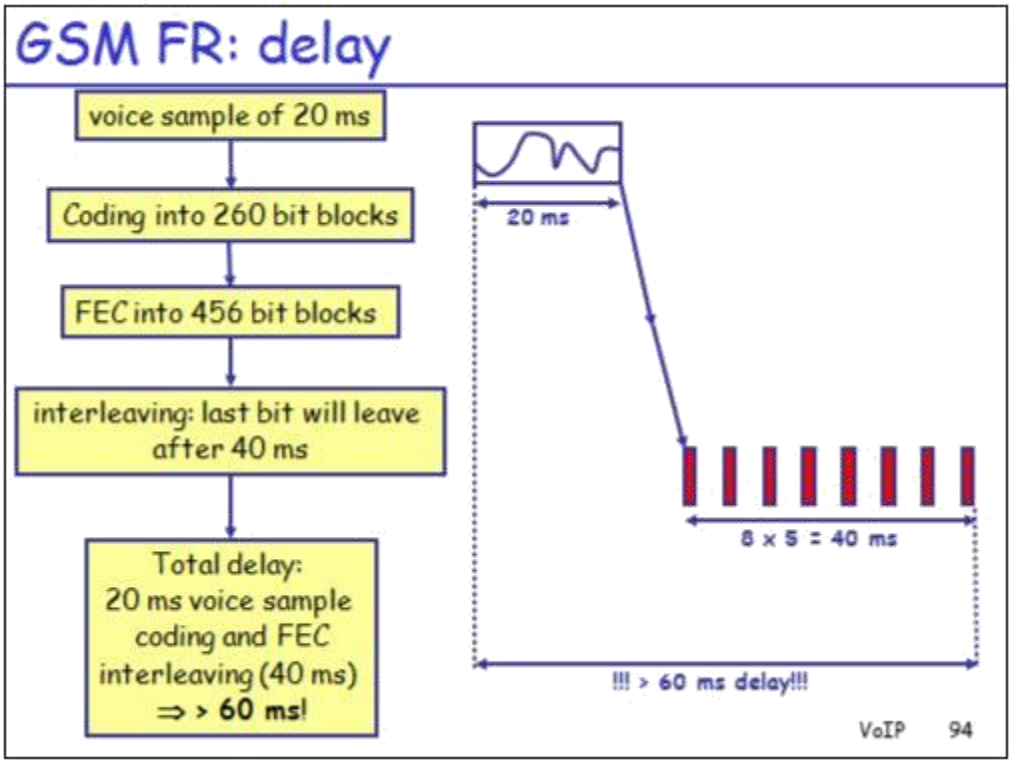


An example of a GSM codec with FEC and interleaving is discussed now. Note that this is not specific for VoIP but it is a very nice illustration.

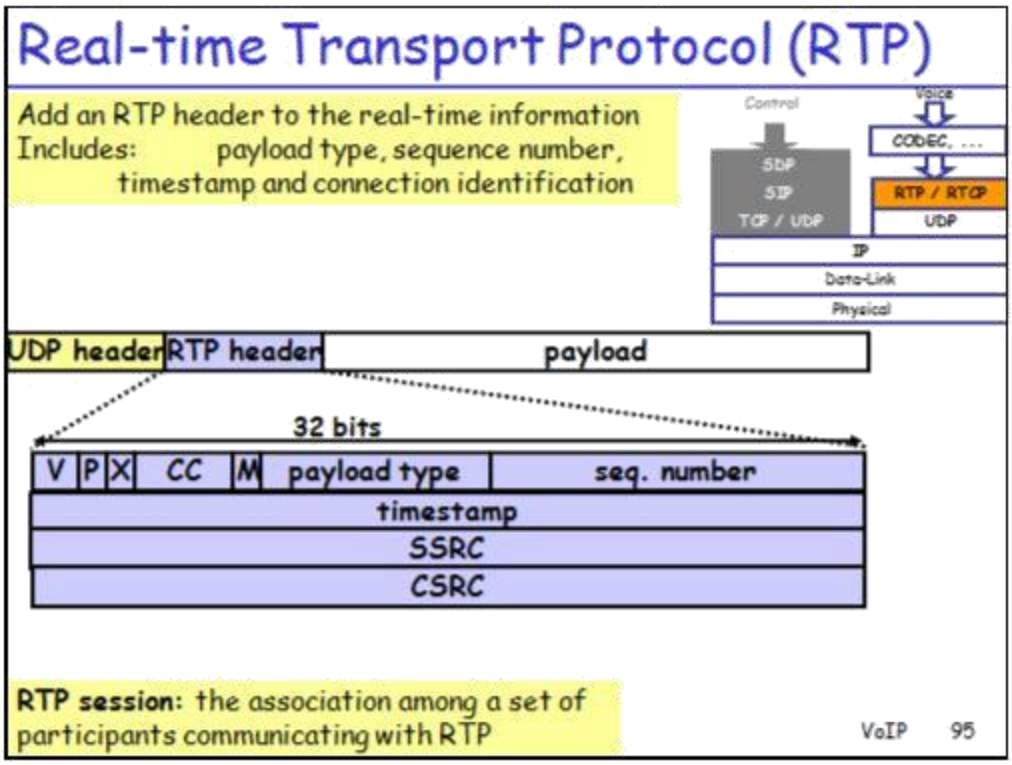
A GSM codec is using voice samples of 20 ms. These samples are coded in blocks of 260 bits generated every 20 ms (resulting in a bitrate of 13 kbit/s). FEC (more complex than the example given before) is used and adds 196 bits to it, resulting in a total of 456 bits every 20 ms (22.8 kbit/s).

These 456 bits are split into 8 equal parts of 57 bits and these are interleaved with other 57 bit parts (from adjacent voice samples). These 57 bit parts are transmitted in radio bursts on the air interface, every 5 ms. A burst contains 114 bits, 57 of one voice sample and the other 57 bits from an adjacent voice sample.

The principle is illustrated in the next slides.



In summary, a delay of more than 60 ms is observed before a voice sample is transmitted over the radio interface. This is a very large delay and it may degrade the voice quality (as is indeed the case with GSM). It is however of prime importance in GSM to use FEC and interleaving in order to counteract the large amount of bit errors at the radio interface.



Another feature a sender may add is the use of RTP (Real Time Protocol). This protocol will add some extra information between the UDP header and the voice data (payload). The RTP header consists of a number of fields (at least 4 x 32 bits), which allow the receiver to put packets in the correct order and provides timing information.

version (V): 2 bits: This field identifies the version of RTP.

padding (P): 1 bit: If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload. The last octet of the padding contains a count of how many padding octets should be ignored, including itself. Padding may be needed by some encryption algorithms with fixed block sizes or for carrying several RTP packets in a lower-layer protocol data unit.

extension (X): 1 bit: 1 if extension header between standard header and payload data.

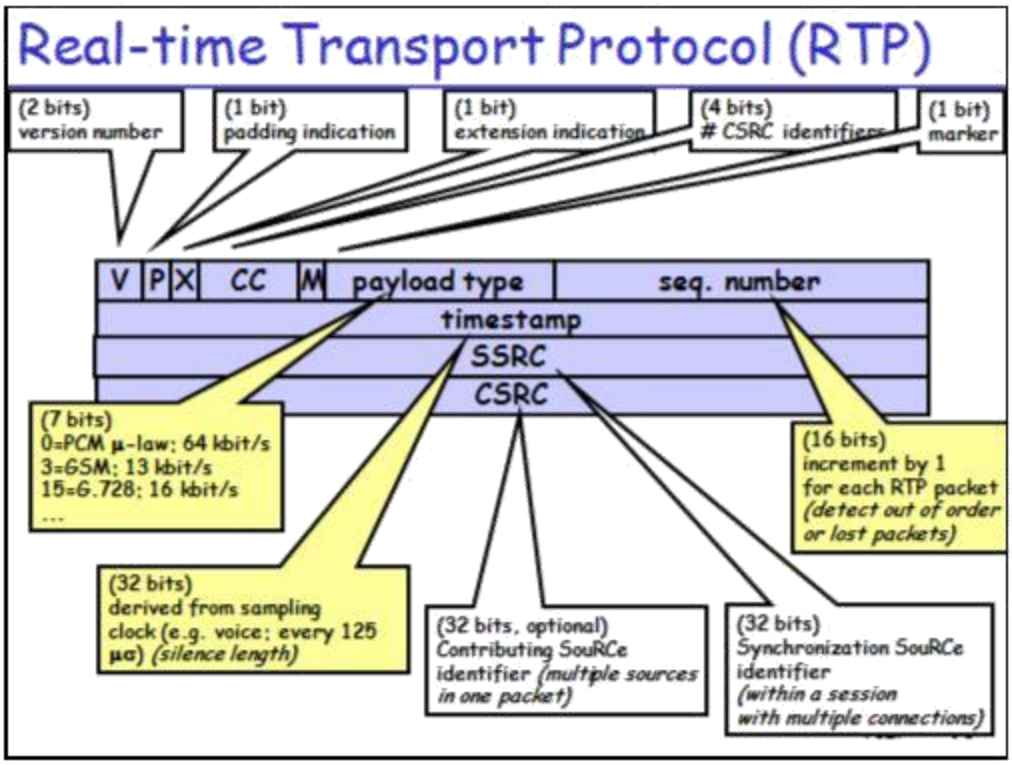
CSRC count (CC): 4 bits. The CSRC count contains the number of CSRC identifiers that follow the fixed header.

marker (M): 1 bit: (indication of some special relevance for the application).

payload type (PT): 7 bits: This field identifies the format of the RTP payload and determines its interpretation by the application.

sequence numbe r: 16 bits: Increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number SHOULD be random (unpredictable).

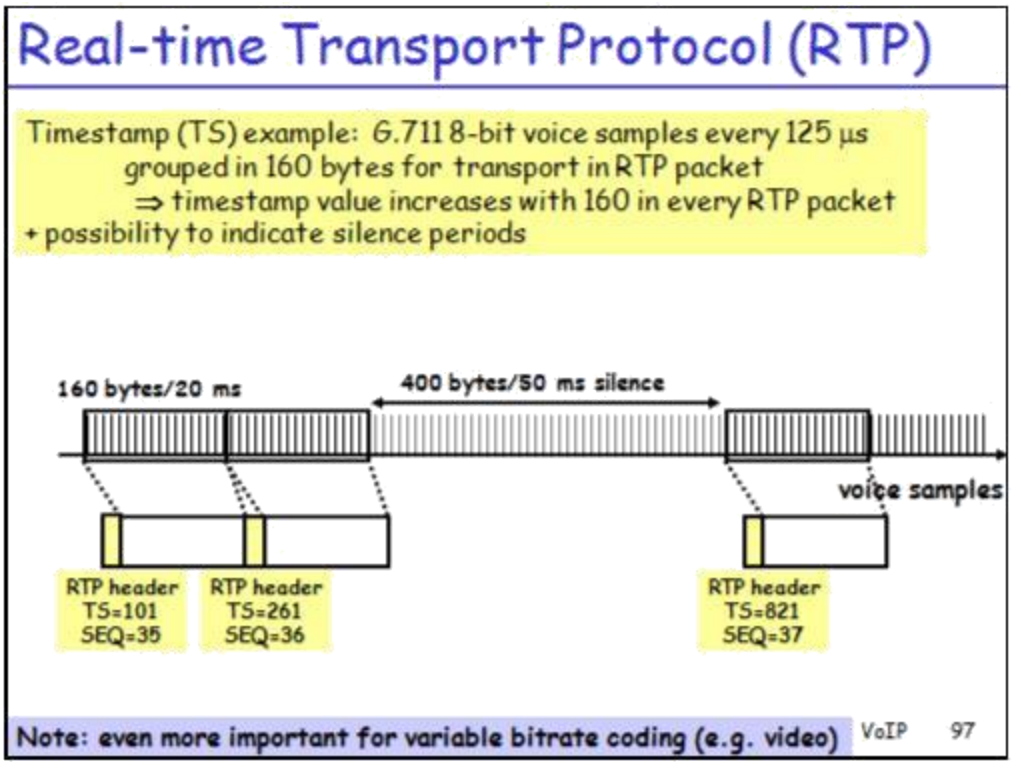
Note: **RTP session:** The association among a set of participants communicating with RTP. For each participant, the session is defined by a particular pair of transport addresses (one network address plus a port pair for RTP and RTCP).



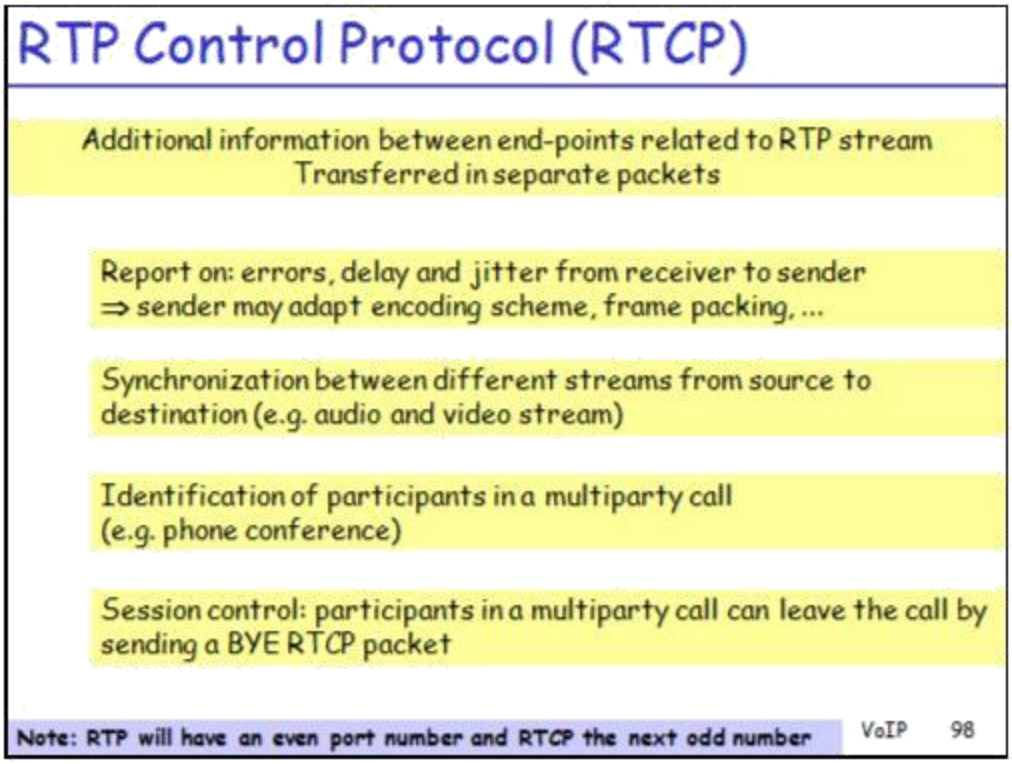
timestamp: 32 bits: The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations. The resolution of the clock must be sufficient for the desired synchronization accuracy and for measuring packet arrival jitter (one tick per video frame is typically not sufficient). The clock frequency is dependent on the format of data carried as payload and is specified statically in the profile or payload format specification that defines the format, or may be specified dynamically for payload formats defined through non-RTP means. As an example, for fixed-rate audio, the timestamp clock would likely increment by one for each sampling period. If an audio application reads blocks covering 160 sampling periods from the input device, the timestamp would be increased by 160 for each such block, regardless whether the block is transmitted in a packet or dropped as silent. The initial value of the timestamp should be random, as for the sequence number.

SSRC: 32 bits: The SSRC field identifies the synchronization source (SSRC = Synchronization SouRCe). This identifier should be chosen randomly, with the intent that two synchronization sources within the same RTP session can not have the same SSRC identifier.

CSRC list: 0 to 15 items, 32 bits each: The CSRC (Contributing SouRCe) list identifies the contributing sources for the payload contained in this packet. The number of identifiers is given by the CC field (maximum of 15). Example: for audio packets, the SSRC identifiers of all sources that were mixed together to create a packet are listed, allowing correct talker indication at the receiver.



The figure illustrates an example of the use of a timestamp for a constant bitrate voice stream (G.711). The G.711 is producing every 125 μs voice samples of 8 bits which are grouped in IP packets (e.g. 160 voice samples per IP/RTP packet). When starting to send out the voice stream, every IP/RTP packet will contain a timestamp (starting at some random value, e.g. 101). This value is incremented by 160 for every IP/RTP packet. If there is a silence period (resulting in no IP/RTP packets), the timestamp will be increased internally in the sender with the same speed (1 per 125 μσ) and when a next IP/RTP packet is sent out, it will have a much larger value (e.g. 821 in the example = 261 + 160 + 400). The sequence number is also illustrated (also starting at a random value, e.g. 35).



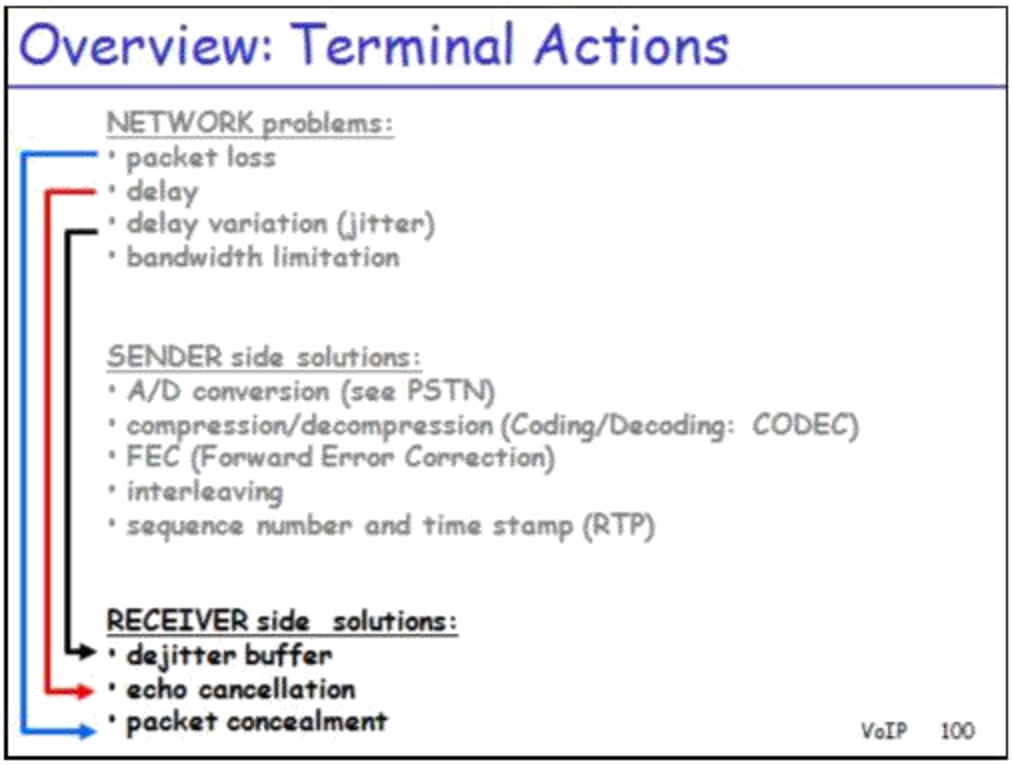
RTCP (Real-time Transport Control Protocol) will provide the possibility to exchange extra information related to the RTP streams. The RTP control protocol (RTCP) is based on the periodic transmission of control packets to all participants in the session, using the same distribution mechanism as the data packets. The underlying protocol must provide multiplexing of the data and control packets, for example using separate port numbers with UDP. RTCP performs several functions:

- The primary function is to provide feedback on the quality of the data distribution. The feedback may be directly useful for control of adaptive encoding. Sending reception feedback reports to all participants allows the observer of problems to evaluate whether those problems are local or global.
- RTCP carries a persistent transport- level identifier for an RT P source called the canonical name or CNAME. Since the SSRC identifier may change if a conflict is discovered or a program is restarted, receivers require the CNAME to keep track of each participant. Receivers may also require the CNAME to associate multiple data streams from a given participant in a set of related RTP sessions, for example to synchronize audio and video.
- Another function is to convey minimal session control informa tion, for example participant identification.
- RTCP will also allow to leave a session by sending a BYE RTCP packet.

RTP and RTCP relies on the underlying protocol(s) to provide demultiplexing of RTP data and RTCP control streams. For UDP and similar protocols, RTP should use an even destination port number and the corresponding RTCP stream should use the next higher (odd) destination port number

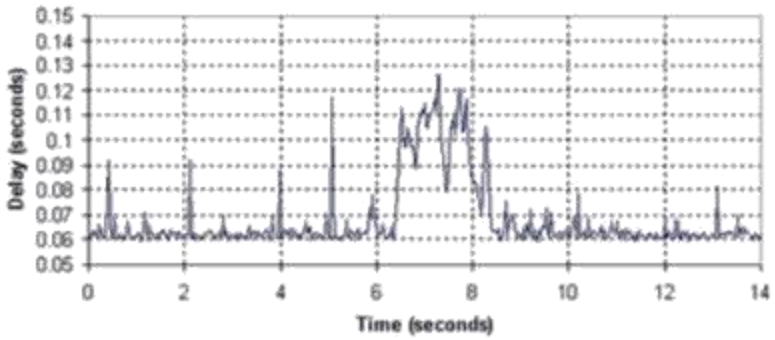
Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     - Sender side solutions
     - Receiver side solutions
     - Influence on voice quality
4. Multimedia over IP



This paragraph will discuss a number of measures used at the receiver side in order to reduce the influence of packet loss (use of packet concealment), delay (use of echo cancellation) and jitter (use of dejitter buffer).

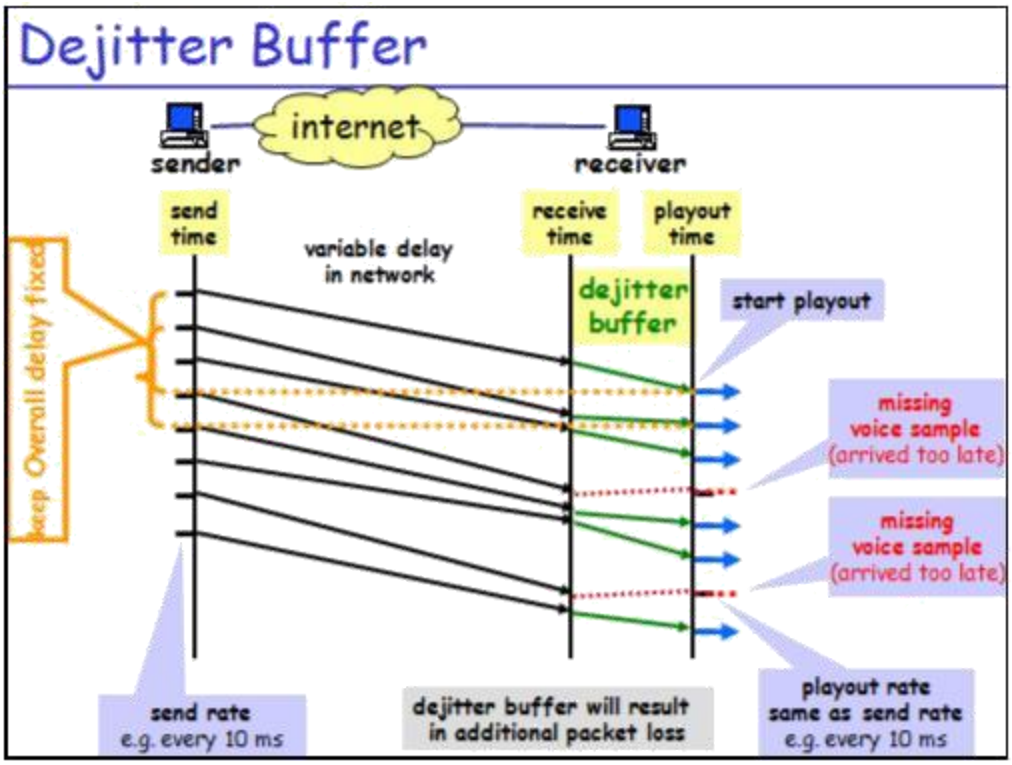
Jitter example: access link congestion



Example: 1500 bytes packet on 384 kbit/s uplink will take ~30 ms
(ISDN, DSL, cable modem)

VoIP 101

Access links are typically a major source of jitter as they represent one of the bottlenecks along the packet path. For example, the serialization delay for a 1500 byte IP packet sent through a E1 (2 Mbit/s) link is approximately 6 milliseconds therefore if five data packets are queued before a short voice packet then an additional 30 milliseconds of transient delay is introduced. This problem can be severe in the case of ISDN, ADSL or Cable Modems in which the upstream bandwidth can be even more restricted; for example if the upstream bandwidth is 384 kBit/s then each queued 1500 byte IP packet would introduce an extra 30 milliseconds of delay. An example is shown in the figure.

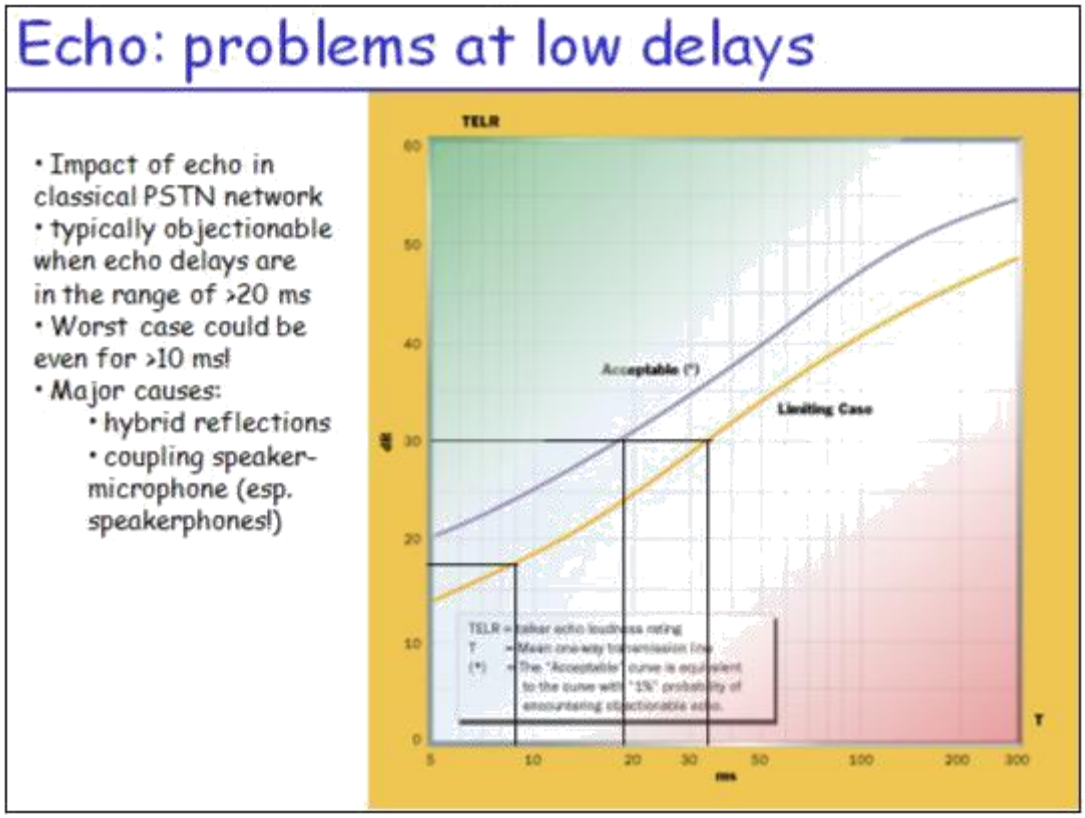


The major solution used to counteract jitter in the network is the use of a dejitter buffer. This buffer allows to reduce drastically the influence of jitter in the network, at the expense of an extra delay and some possible packet loss. The dejitter buffer works as follows:

A sender will send packets at a constant rate to the receiver (e.g. every 10 ms). When transiting the network, delay variations will occur resulting in a degraded signal when played out without counter measures. When the first packet arrives however in a receiver with a dejitter buffer (e.g. voice RTP stream), it will be delayed for a certain time (it will be kept for some time in the dejitter buffer, e.g. 15 ms) before playout. From then on, a timer will start that will play out the next arriving voice packets at a fixed rate (e.g. every 10 ms). The next packet may observe a higher delay in the network and therefor it will be delayed in the dejitter buffer for a shorter time in order to keep the playout rate constant.

It may happen that a packet arrives after its scheduled playout time (e.g. packet 4). In that case it will be discarded and regarded as a lost packet.

In general the use of a dejitter buffer will keep the overall delay constant, from sending the packet on the network till the playout. Packets arriving too late will be discarded.



The Figure from ITU-T G.131 (control of talker echo) shows the variation in acceptable perceived echo loudness (as level difference between voice and echo) versus total delay.

•TELR = talker echo loudness rating = SLR + RLR + R + T + Lr

•SLR = speaker loudness rating = 7dB nom, 2dB min for most telephones

•RLR = receiver loudness rating = 3 dB nom, 1 dB min for most telephones

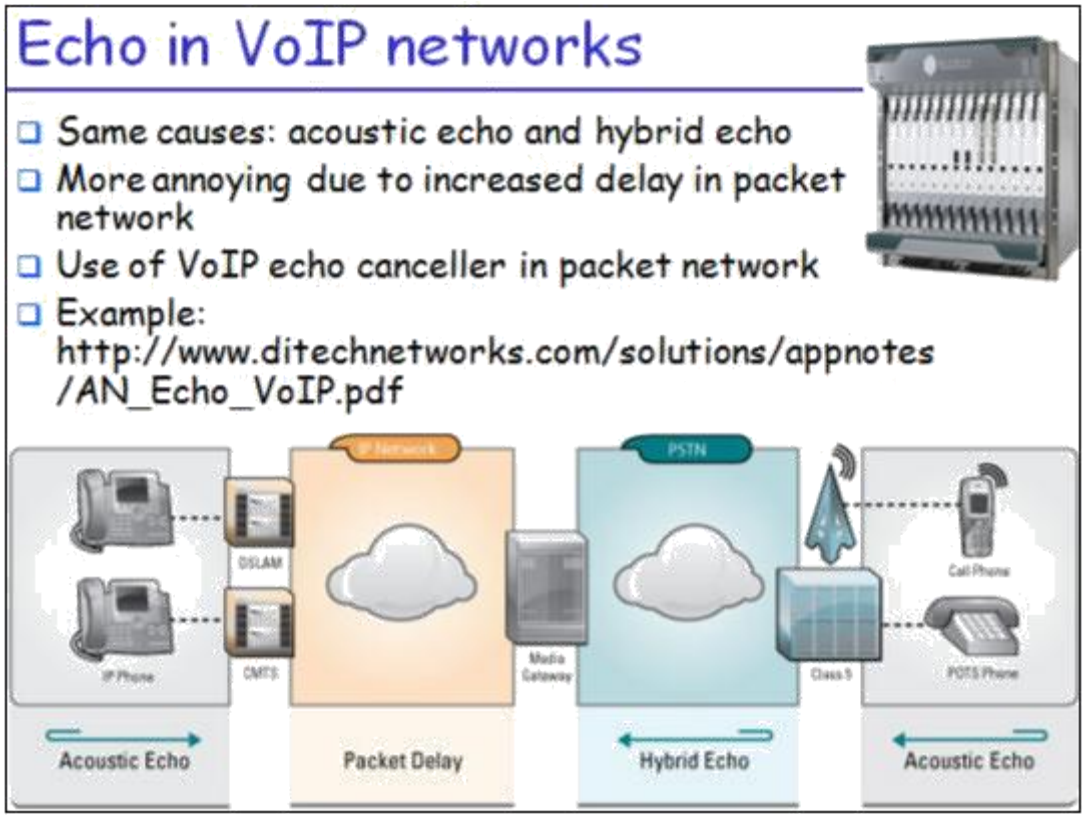
•R = receive loss

•T = transmit loss

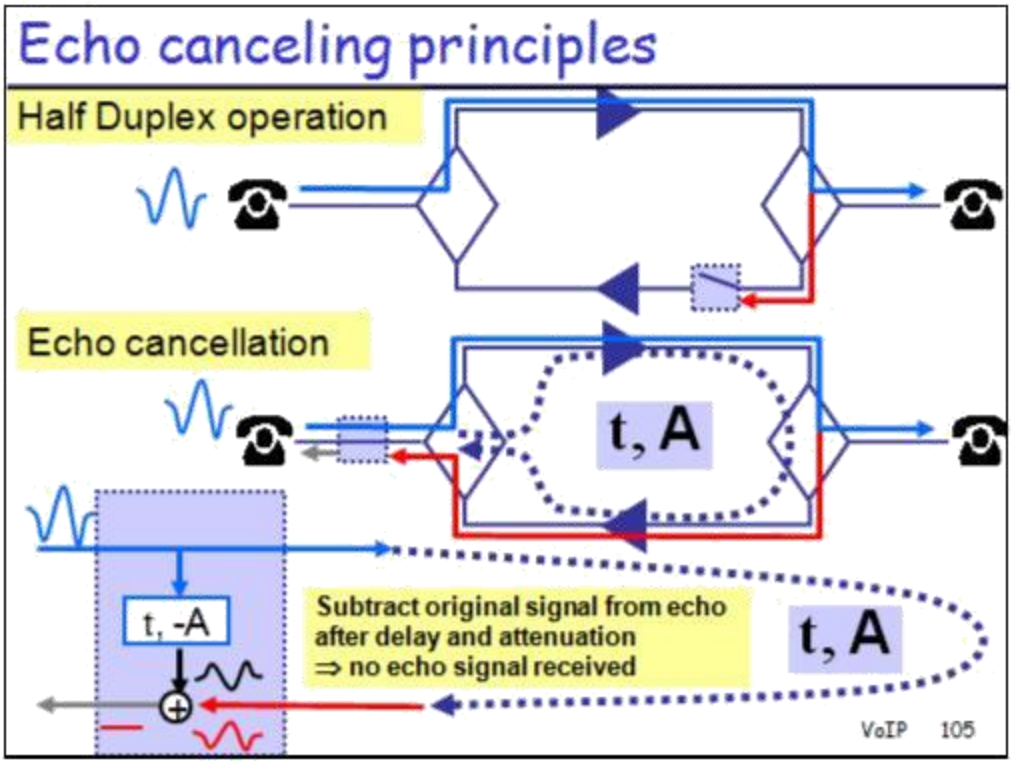
•R+T = 6 dB is introduced in most calls in the US for echo control

•Lr = return loss or hybrid balance = 14 dB nom, 8 dB min for line cards that subtract a constant fraction of the signal being sent due to the variation in telephone and loop impedance.

Adding all of these variations gives a 17-dB minimum TELR and 30-dB nominal. The worst case TELR of 17 dB would be objectionable for most subscribers at >8 ms of delay, and objectionable to some subscribers starting at less than 5-ms delay. The average TELR of 30 dB would be objectionable to most subscribers at >35 ms of delay, and could be objectionable to some subscribers at >18 ms of delay.



Echo is a severe problem in VoIP networks due to the increased delay in the packet networks. Solutions are introduced in the packet network itself (e.g. http://www.ditechnetworks.com/solutions/appnotes/AN_Echo_VoIP.pdf)

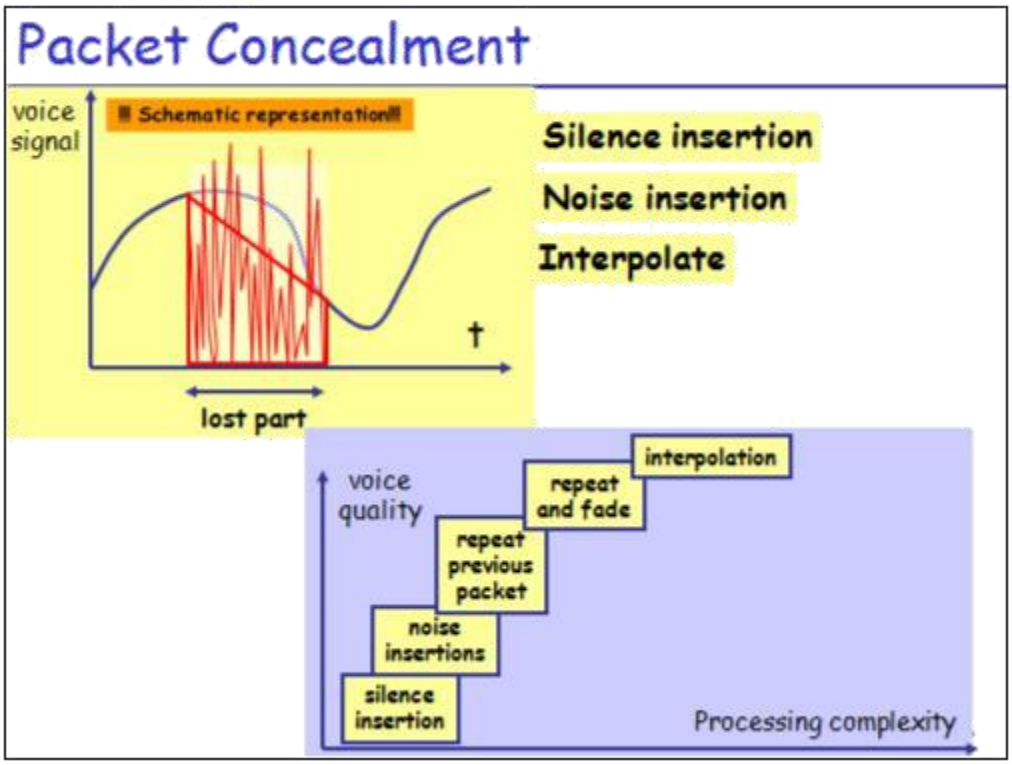


There exist two major solutions to resolve the echo problem:

• Half duplex operation: in this case one is transmitting a voice signal only in one direction at a time (one is blocking the other direction, as indicated in the figure). In this way one will of course also block the echo signal. Because in the end one is interested in talking in both directions, a voice activity circuit will be used which detects who is speaking. This circuit will activate the correct transmission direction (one at a time). Note that the use of half duplex makes the conversation much more difficult in terms of "co-ordination": only one person can tal k at a time (walkie-talkie type of operation).

• Echo cancellation: in this case an echo cancellation block is used in order to suppress the echo. This is done by "recording" the original sig nal and "replaying" it at a later time (t) with an appropriate attenuation (A). By inverting this signal and adding it from the received echo, one cancels effectively the echo (as indicated on the figure).

Remark: One should avoid to use several echo cancellations in series (e.g. at different places in the network) because this degrades the perceived voice quality.



If a part of the voice stream is lost (e.g. packet loss in the network or packet discard in the dejitter buffer), the receiver may try to resolve the problem. There are several ways to do this, as illustrated in the figure. Silence insertion will insert silence during the time when no information is available. Noise insertion will insert noise and interpolation will send a signal that is an interpolation between the last received value (before the lost part) and the first received value after the lost part. Another possibility is to replay the last received part (with or without fading).

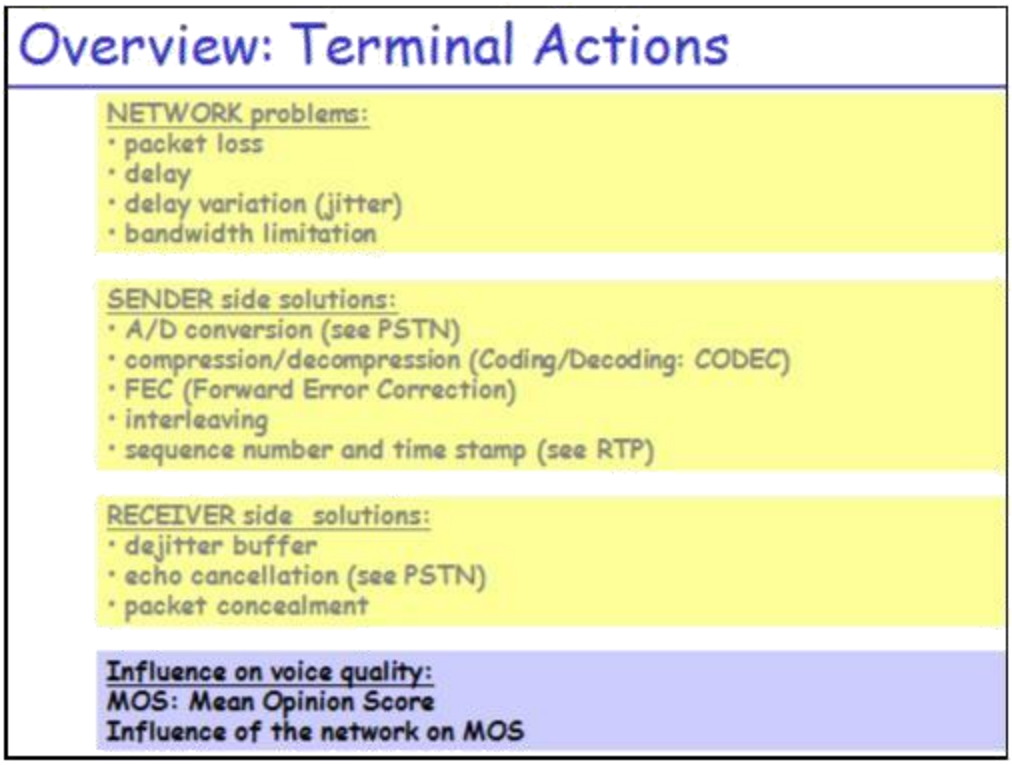
The processing complexity and voice quality of the different techniques is illustrated in the lower graph.

Note that packet concealment is only useful when there is only a small part of information missing (not a burst).

Of course FEC and interleaving will also be used at the receiver side to recover from packet loss.

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     - Sender side solutions
     - Receiver side solutions
     - Influence on voice quality
4. Multimedia over IP



A last part of this paragraph on terminal behavior will be devoted to the influence of different degradation effects on the perceived voice quality.

Perceived Quality: MOS

MOS (Mean Opinion Score): the opinion of an average public on the perceived voice quality under well controlled conditions
(e.g. background noise, mix of public, ...)

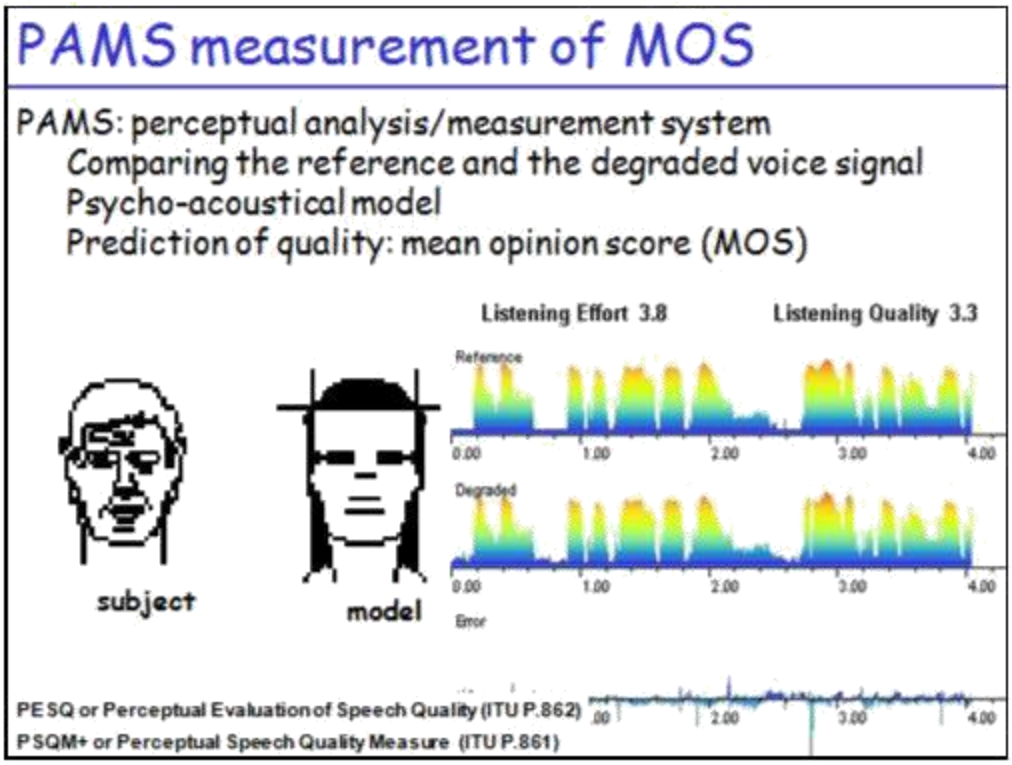
!!! difficult and time consuming measurements!!!

MOS-scale: from 1 to 5

VoIP 109

The characterization of perceived voice quality is very difficult because it is a subjective matter. One has defined a measure to represent the voice quality: MOS or Mean Opinion Score. This will be based on the opinion of a heterogeneous public listening to a voice stream under well controlled conditions (e.g. noise in the room). From the individual scores (ranging from 0 to 5), an average will be calculated giving a measure for the voice quality.

Note that MOS measurements are very difficult (special room environment is required and very good audio equipment) and time consuming (audience to be invited, tests themselves take also some time, …).

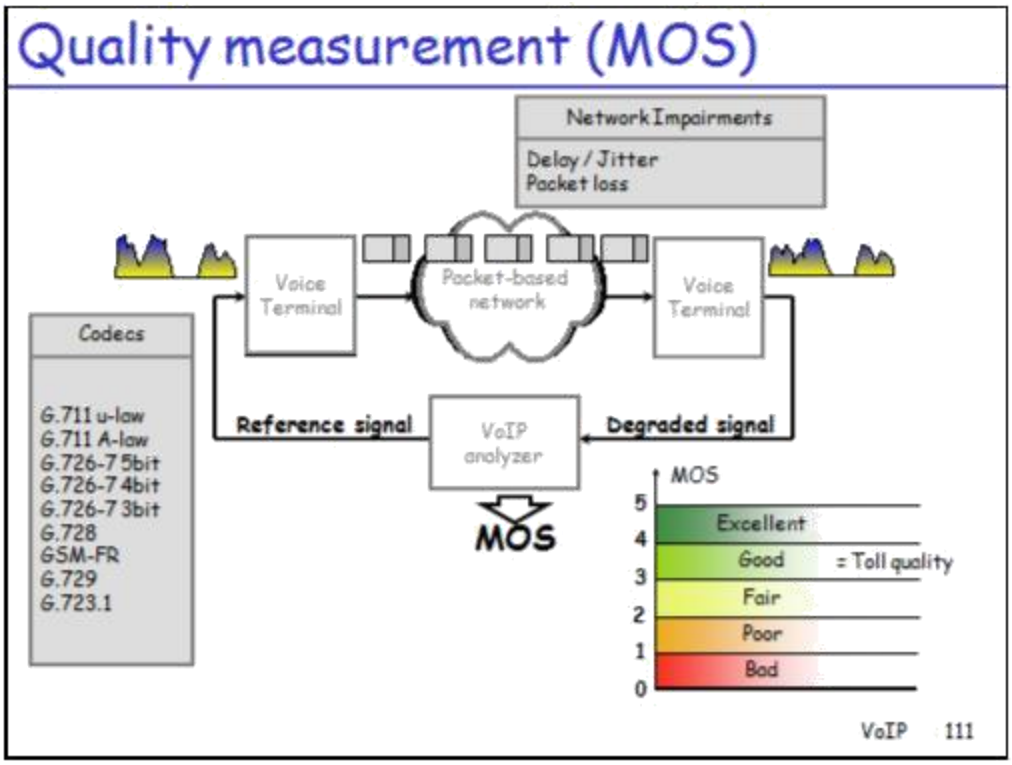


In order to reduce the burden of MOS measurements, some mathematical models (psycho-acoustic) have been proposed that will form a model for the human perception. This will be very useful for the fast characterization of MOS because it may run on a computer platform. An example is PAMS: Perceptual Analysis / Measurement system. The set-up will generate a voice stream (few seconds) and this will be transmitted over e.g. the Internet. The voice stream after transmission will be compared with the original voice stream and from this difference a MOS value will be calculated.

Other recent measurement techniques are: PESQ or Perceptual Evaluation of Speech Q uality (ITU P.862) and PSQM+ or Perceptual Speech Quality Measure (ITU P.861)

Psychoacoustics is based heavily on human anatomy, especially the ear's limitations in perceiving sound:

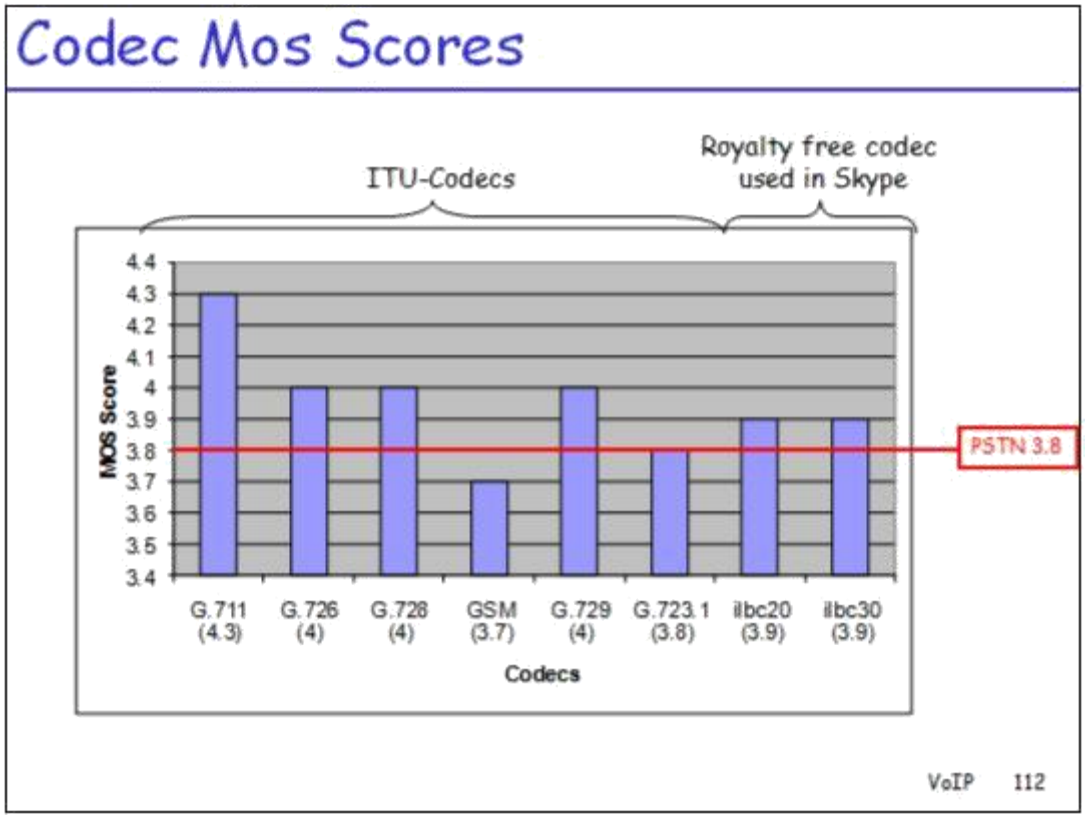
* High frequency limit
* Absolute threshold of hearing
* Temporal masking
* Simultaneous masking



A typical measurement set-up is shown in the figure. A number of measurements will be illustrated in the next slides, illustrating the problems encountered when sending voice over internet.

A VoIP analyzer will generate analog voice streams. From this reference signal, a voice terminal will generate digital voice streams using different codecs (see list). The packets will be transmitted over an internet (sometimes emulated by using an impairment node*). At the receiving side the digital voice stream will be decoded and the analog (degraded) voice signal will go to the analyzer. The VoIP analyzer will generate the MOS value.

* An impairment node will introduce artificial packet loss, delay and delay jitter.

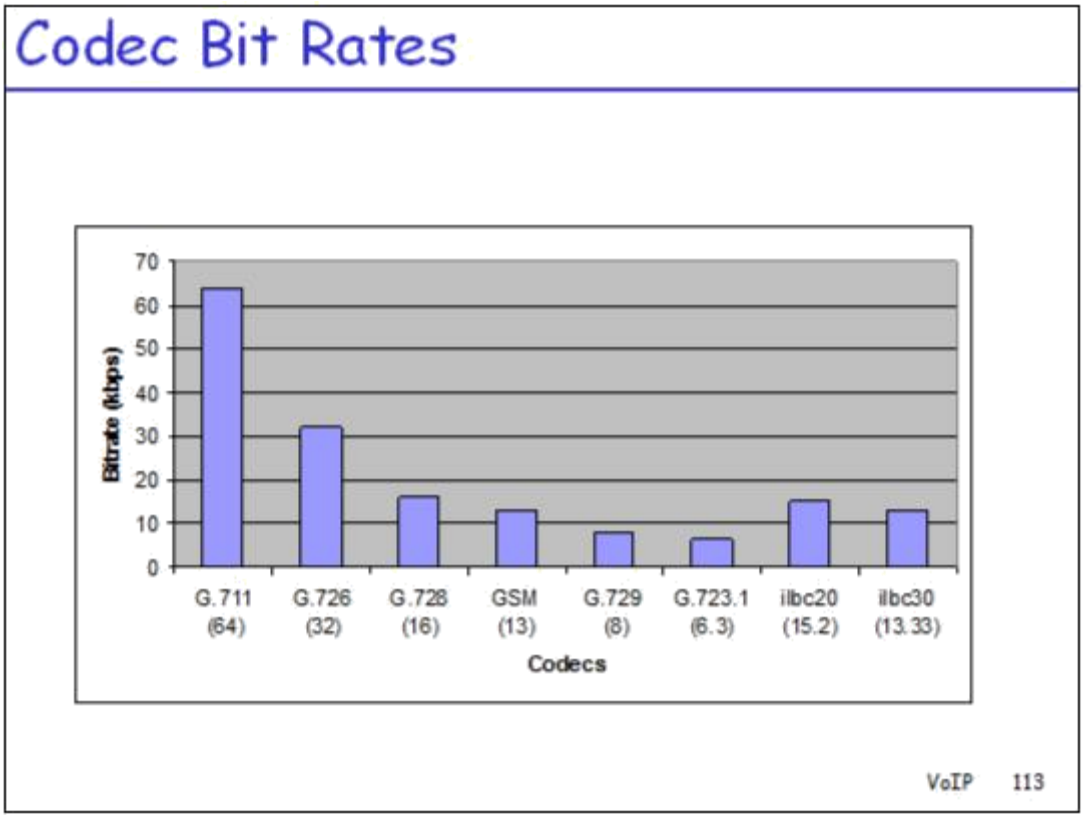


This slide shows the MOS values for different codecs when no network impairment is present (back-to-back measurement: no network delay, jitter or packet loss). A standard telephone call over the PSTN network results in a MOS value of 3.8. The difference in quality is thus caused by the compression used (the compression is not lossless).

The digital PSTN (ISDN or trunks) makes use of the G.711 codec, so normally (G.711 = 4.3 in the graph) a higher MOS Score is expected than 3.8. The typical telephony band is limited from 300 to 3400 Hz. This explains the lower MOS score obtained with PSTN. G.711 is a wideband codec (up to 8Khz).

Ilbc is described in IETF RFC 3951.

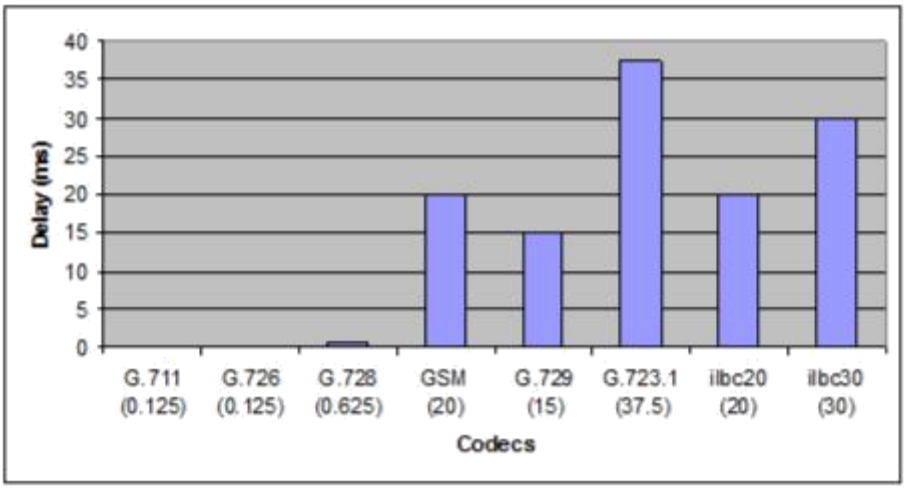
These numbers come from literature. PSTN 3.8 was confirmed with own measurements.



The graph illustrates the bitrates used by the different codecs. If this graph is compared with the 'Codec MOS scores', one can see that some codecs (e.g. G.729, which is commonly used) can combine a low bitrate with a high MOS score.

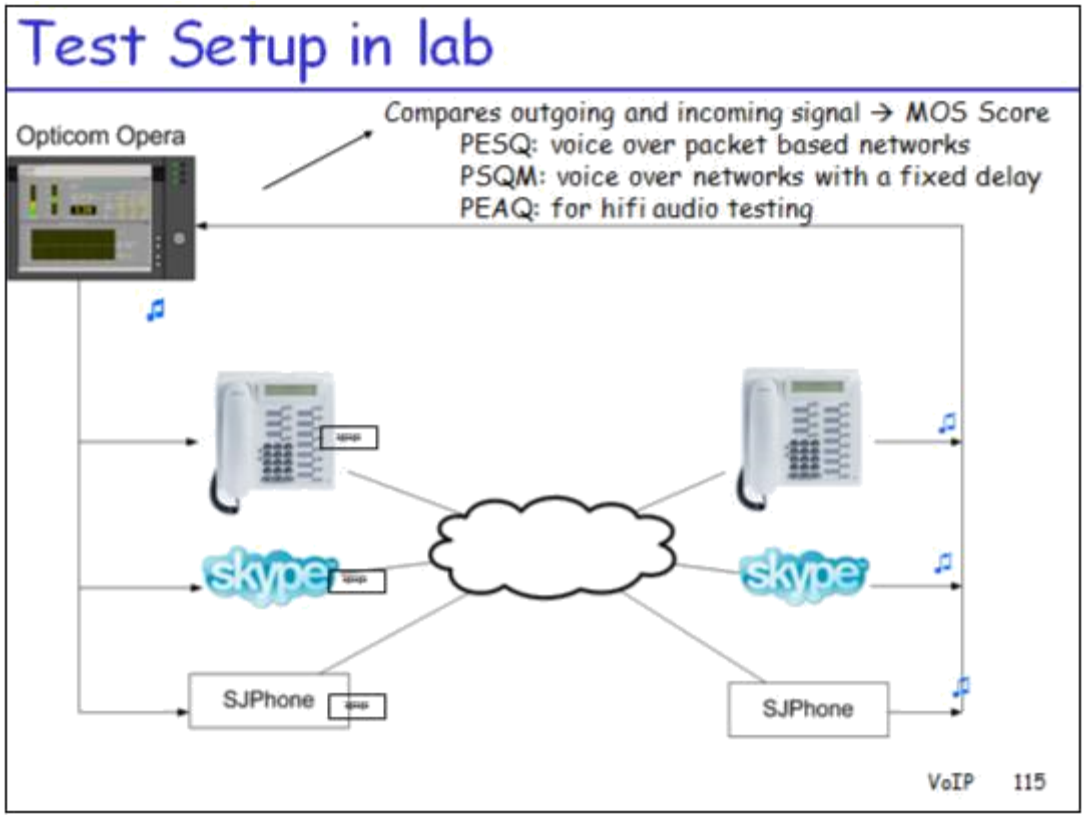
These numbers come from literature.

Coding Delays



VoIP 114

These values are reference values from literature and show the typical delay introduced by the use of a compression/decompression algorithm (codec). A low end-to-end delay (coding delay, network delay, buffering in the terminals) is important to have a good interactive call.



A typical measurement set-up is shown in the figure. A number of measurements will be illustrated in the next slides, illustrating the problems encountered when sending voice over data networks or the Internet.

A VoIP analyzer will generate analog voice streams. From this reference signal, a voice terminal will generate digital voice streams using different codecs. The packets will be transmitted over a network (sometimes emulated by using an impairment node*). At the receiving side the digital voice stream will be decoded and the analog (degraded) voice signal will go to the analyzer. The VoIP analyzer will generate the MOS value by comparing the reference and the recorded degraded signal.

The algorithms based on psycho-acoustic used for quality analysis are:

PESQ = Perceptual Evaluation of Speech Quality (ITU-T P.862)

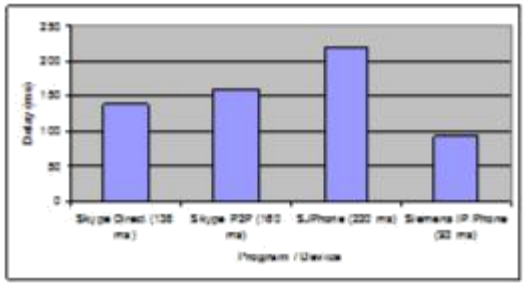
PSQM = Perceptual Speech Quality Measure (ITU-T P.861)

PEAQ = Perceptual Evaluation for Audio quality (ITU-R rec. BS.1387)

* An impairment node will introduce artificial packet loss, delay and delay jitter in a controlled way.

Acceptable Delays - typical delays

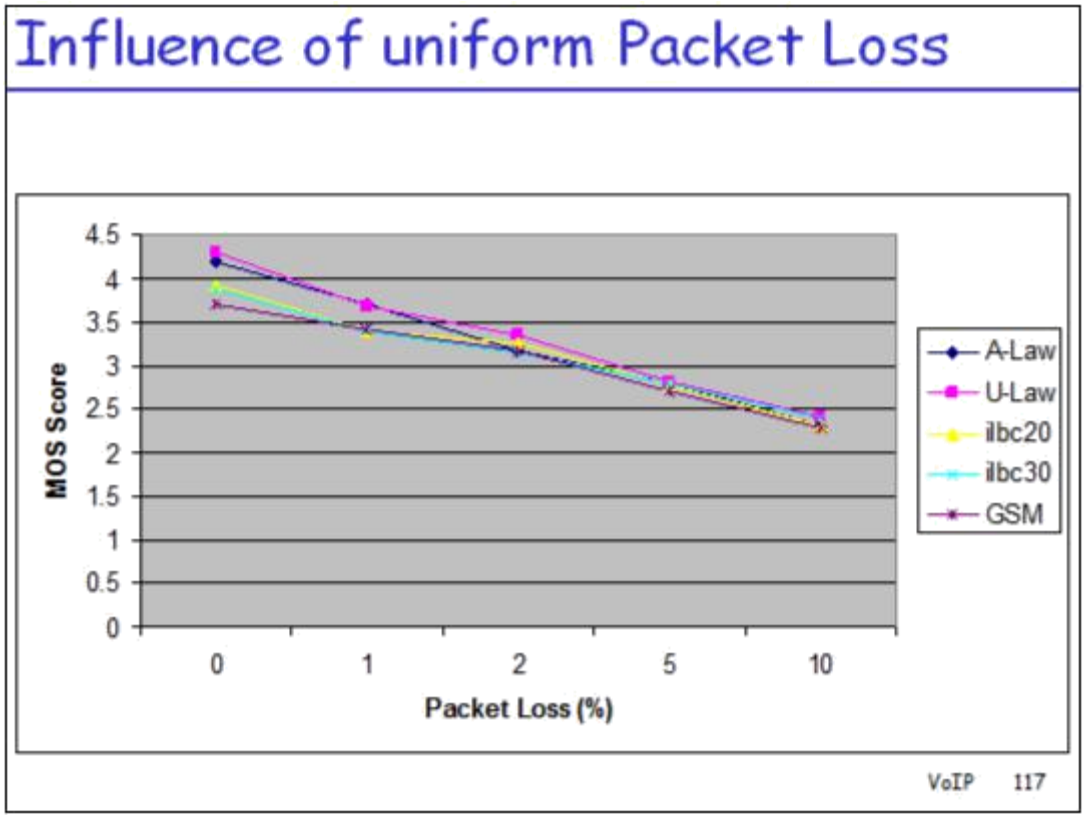
- The maximum acceptable one-way delay set by the ITU for a good quality VoIP call is 150 ms
- In general this maximum delay is very hard to obtain using software (due to buffering) and therefore 150-400 ms is considered acceptable by users.



VoIP 116

We notice that software (Skype, SJPhone) in general has higher delays than a hardware solution (Siemens IP Phone). This is due to the buffering on several levels (soundcard, audio drivers, dejitter buffer, …). Skype direct and skype peer-to-pee r (P2P) are further discussed in more detail.

.



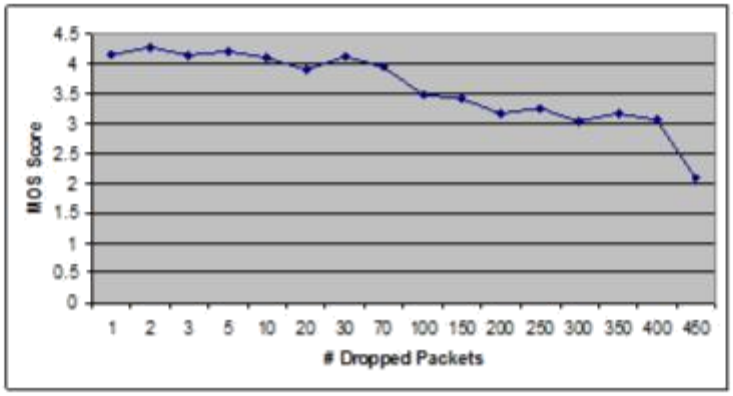
From 0-1 % packet loss we see a variable MOS score for different codecs (some codecs are more robust), but from 1% on, the MOS scores converge.

Bursty packet loss

- More realistic than uniform packet loss when considering congestion behavior in networks/the Internet.
- A number of consecutive packets are dropped in a burst.
- When the burst loss occurs in a silent part of the sample, there is no degradation of the MOS Score

Influence of Burstloss

- The complete voice sample is around 2000 packets
- Degraded signal:
  - Quality of audio sample is impeccable over the whole BUT some parts are dropped (silence is heard)
  - Minor influences of the dropped parts on the PESQ score
- Tests were done using SJPhone and the G.711 codec



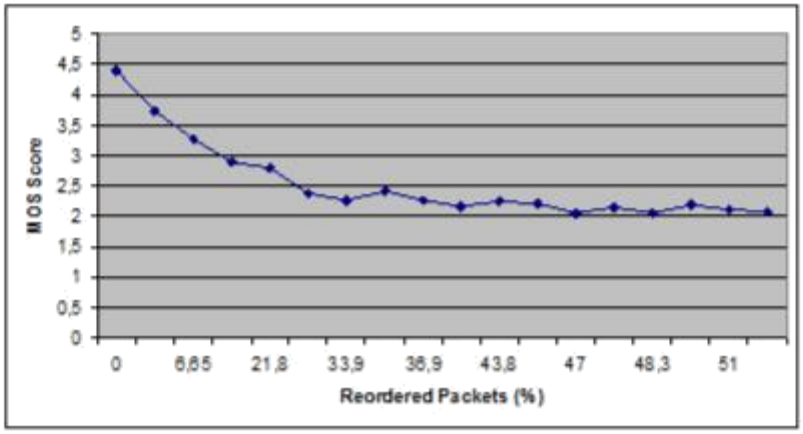
VoIP 119

100 packets = around 5% loss

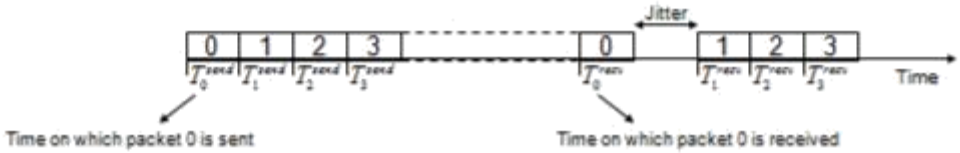
Influence of reordering

Tests + Results:

- The packet delays are spread within a gaussian distribution. The amount of reordered packets (%) is indicated on the X-axis
- Conclusion: Packet reordering is not very well handled by multiple VoIP programs
- Tests were done using SJPhone and the G.711 codec



Jitter



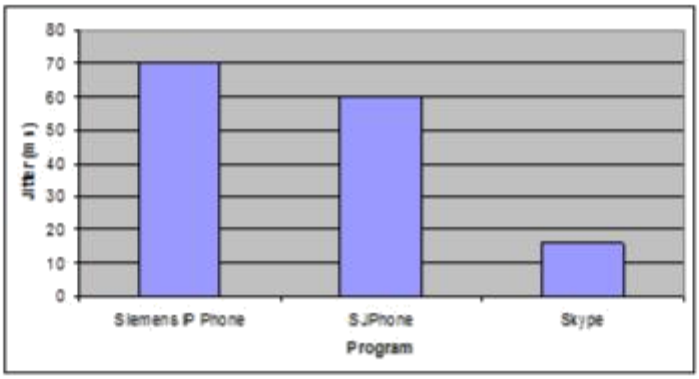
- Jitter: multiple definitions
  - Basically: variation in delay of packets in the internet
  - We use: The absolute value of the difference between two successive packet delays
  - $abs[(T_i^{recv} - T_i^{send})-(T_{i+1}^{recv} - T_{i+1}^{send})]$
  - = IETF standard RFC 3393: IP Packet Delay Variation Metric
- Caused by:
  - Network congestion changes
  - router internal queue behavior
  - routing changes
  - ...
- Solution: (de)jitter buffers

VoIP 121

Dejitter buffers (playout buffers) buffer the received data before they play it. Jitter is removed by this as all data can be played with the right timing.

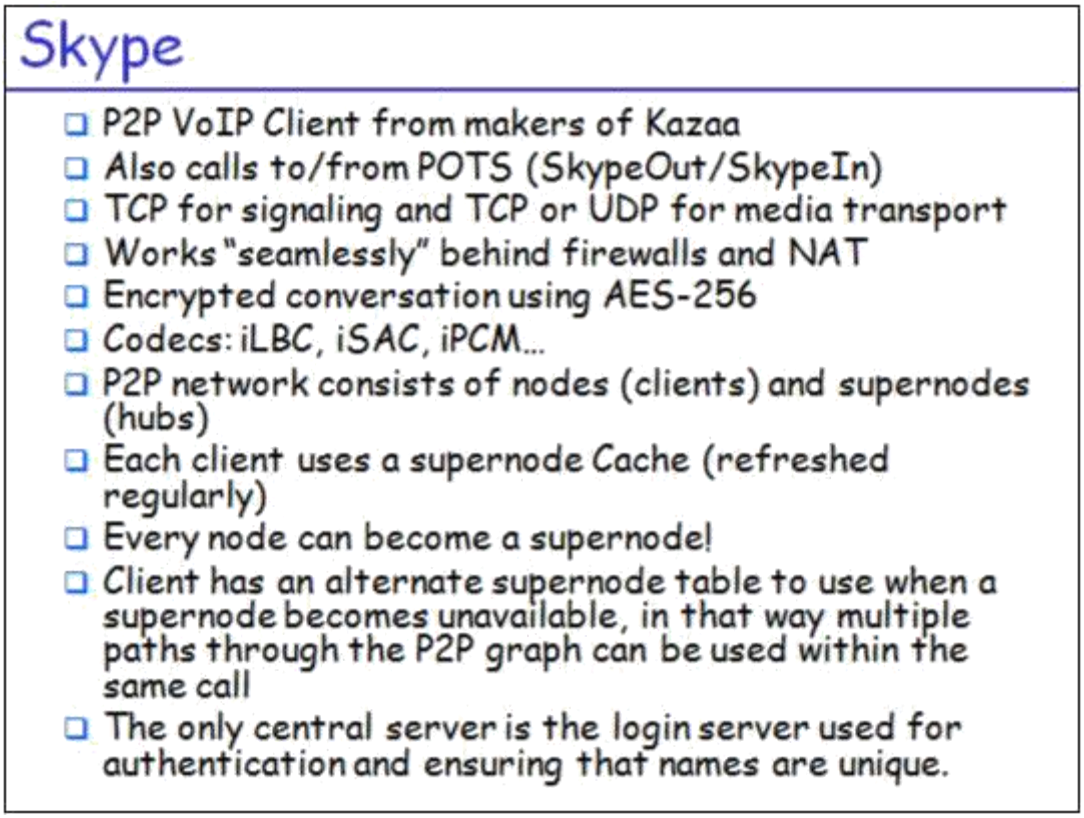
Influence of Jitter (Example)

- The graph shows the maximal amount of jitter that can be handled with the default settings of a certain program/device without losing any audio quality



VoIP 122

When considering the audio quality with increasing jitter, we see a sudden drop of the MOS Score. This means the jitter capacity of the current buffer settings has been exceeded.



Skype is a software program created by the Swedish and Danish entrepreneurs Niklas Zennström and Janus Friis. Skype allows users to make teleph one calls over the internet to other Skype users free of charge and to landlines and cell phones for a fee. Additional features include instant messaging, file transfer, short message service, video conferencing and its ability to circumvent firewalls.

Codecs:

- iLBC (Internet Low-Bitrate Codec): starts from 16-bit PCM, 8kHz =>constant bitrate 13.3 kbit/s for 30ms samples; 15.2 kbit/s for 20 ms samples; free codec, see RFC3951.

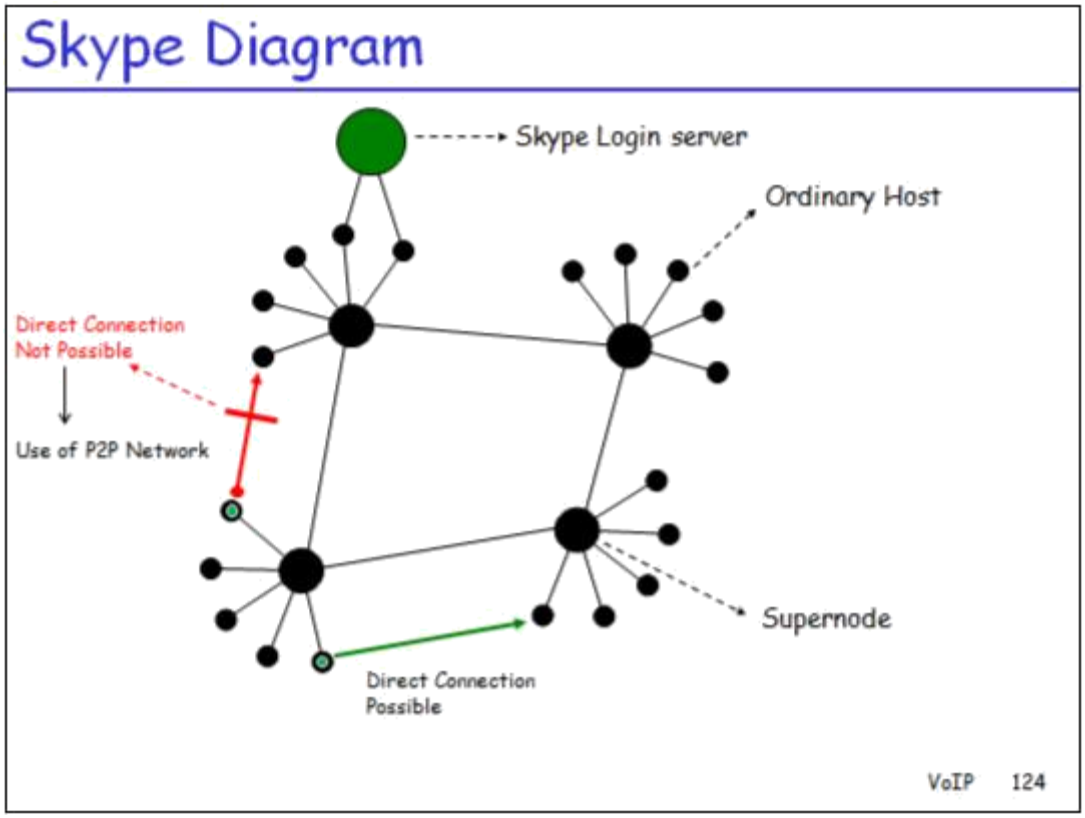
- iSAC (Internet Speech Audio Codec): starts from 16kHz => variable bitrate, 10-32 kbit/s; for adaptive 30-60ms packets; proprietary

- iPCM: wideband codec, 80kbps

POTS = Plain Old Telephone Service

AES = Advanced Encryption Standard

P2P = peer-to-peer



There are three node types involved for PC-to-PC calls:

- Ordinary hosts: Skype Client (SC)

- Super nodes (SN): any SC with sufficient CPU, memory, bandwidth can become super node. (User has no control over it.) The SNs are used for routing login messages, signaling messages and media.

- Login server: the only central component provided by Skype, which keeps track of all the users (e.g. to keep names unique, to store passwords, …).

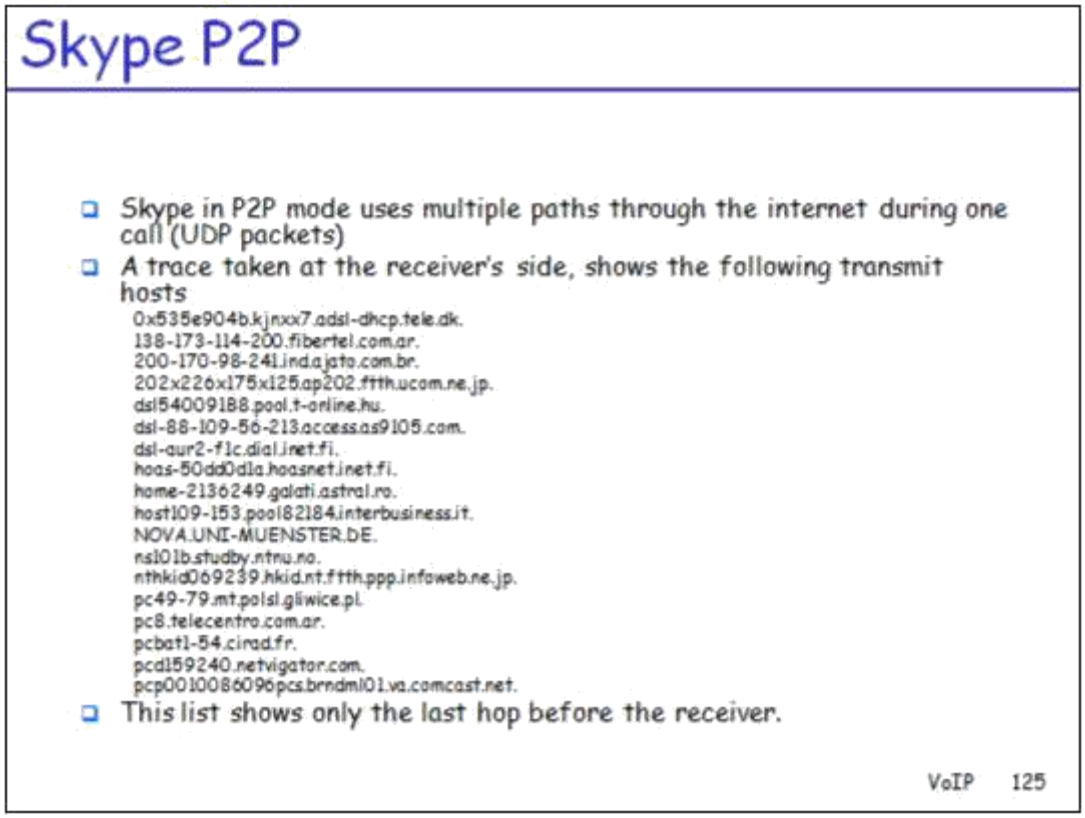
Other servers are provided by Skype for calls from/to the PSTN.

A SC stores a list of SN addresses in a host cache. It is bootstrapped from a set of (hard-coded) IP addresses at startup if the cache is empty.

The online/offline user information is stored in a decentralized fashion.

There are two possible operations of Skype VoIP calls: in the first case there is a direct connection between the two users (no NAT), in the second case there is NAT in between the two users (in that case one has to use supernodes).

A more detailed and interesting (reverse engineered) analysis may be found at http://www1.cs.columbia.edu/~salman/publications/skype1_4.pdf (written by the inventors of SIP).



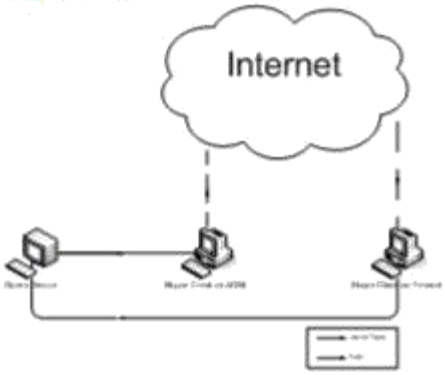
Skype is a software program created by the Swedish and Danish entrepreneurs Niklas Zennström and Janus Friis. Skype allows users to make teleph one calls over the internet to other Skype users free of charge and to landlines and cell phones for a fee. Additional features include instant messaging, file transfer, short message service, video conferencing and its ability to circumvent firewalls.

Skype measurement

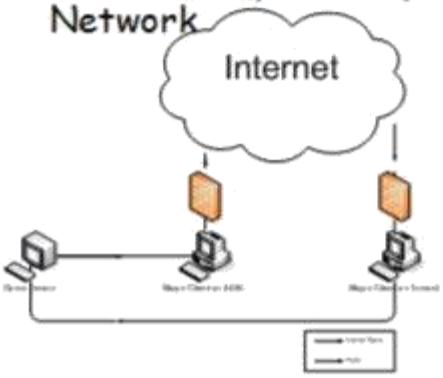
- A long term measurement (> 10h) was performed.
- Each 10 minutes a Skype call was initiated
- This call was measured using the opera equipment.
- Following are graphs of:
  - the MOS score obtained per call
  - The minimum, maximum and average delay per call

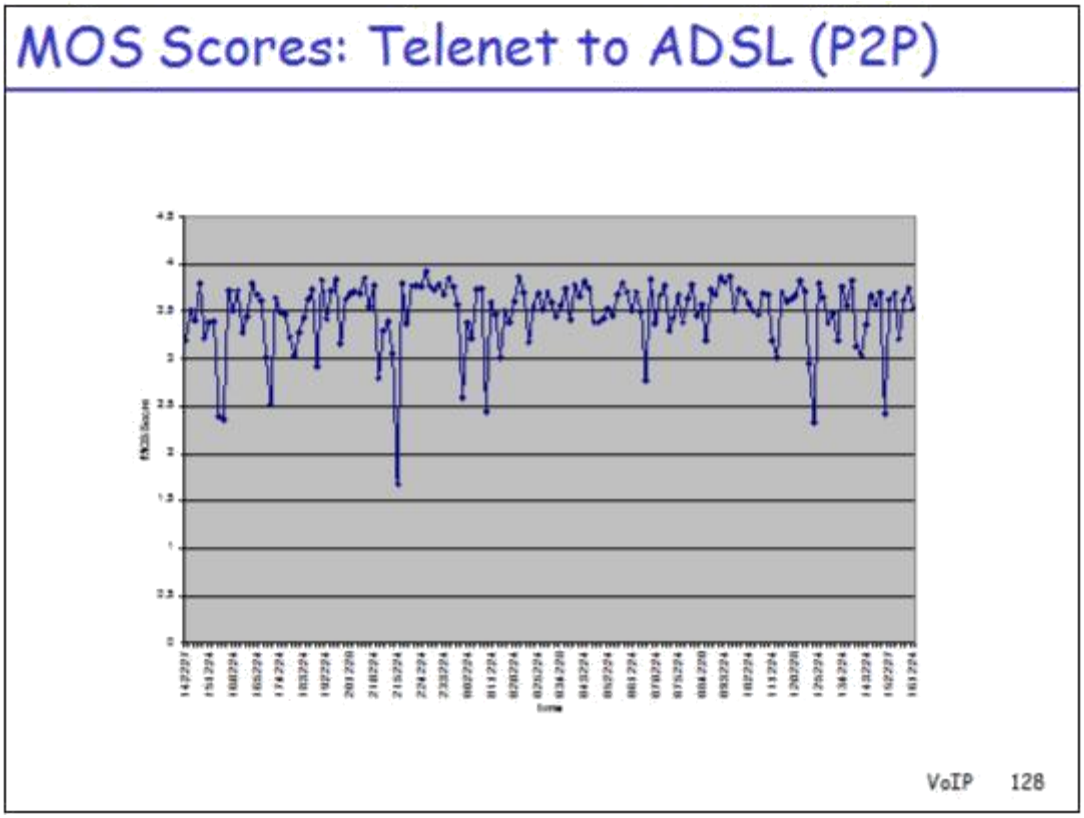
Test Setup

- No firewall/NAT device is used
- Direct Connection



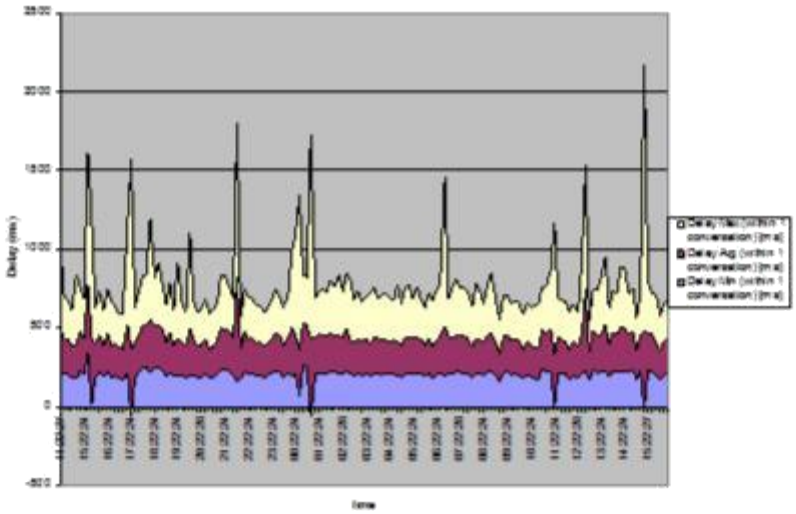
- Computers are behind firewall/NAT
- Use of P2P (peer-to-peer) Network

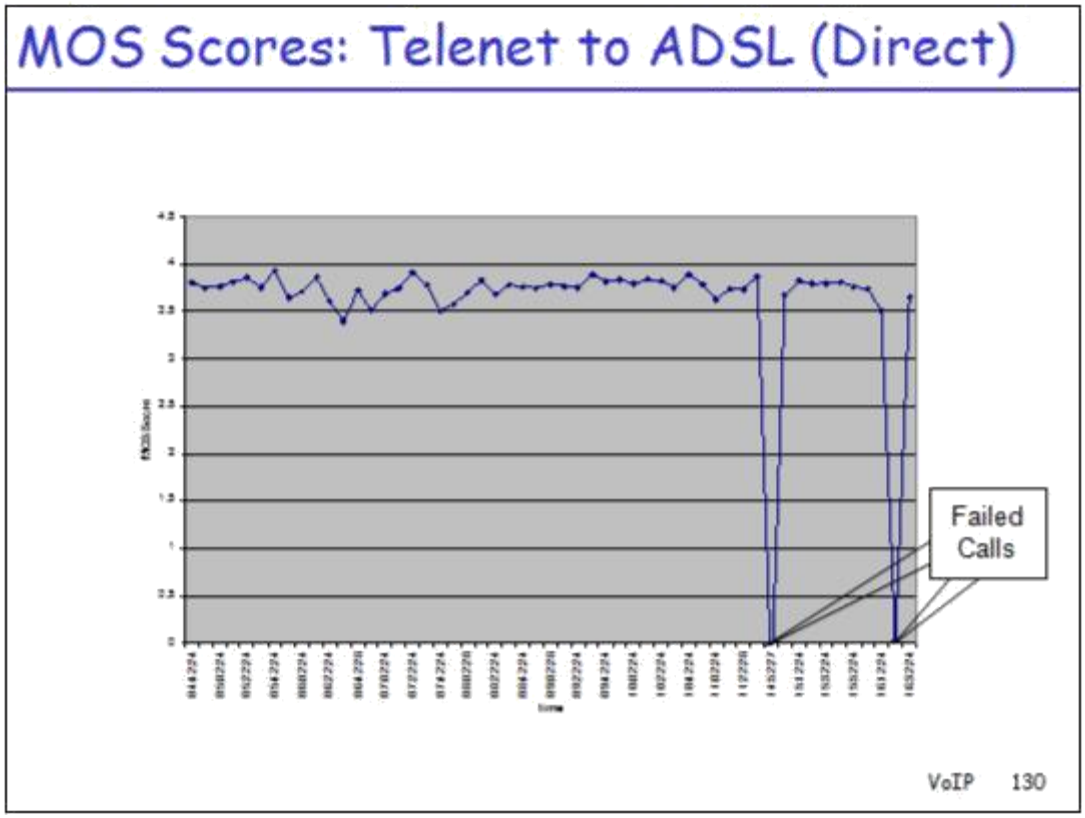




In this graph we clearly notice a very variable MOS score. Average is about 3.5.

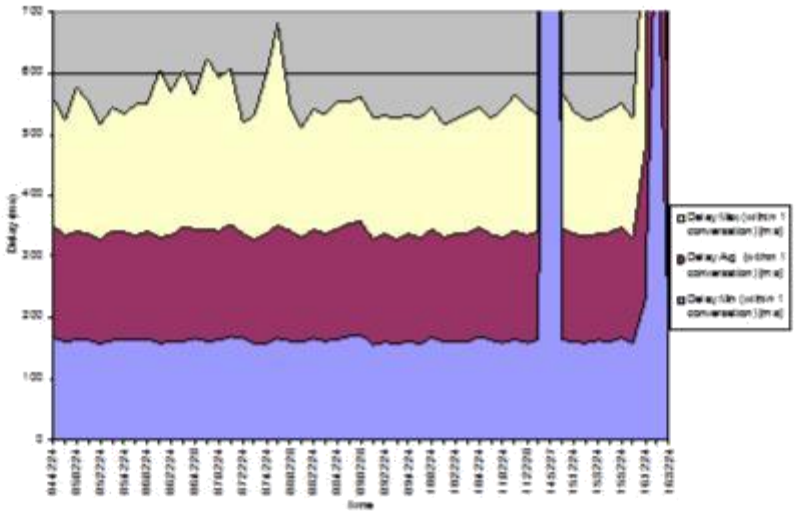
Delays: Telenet to ADSL (P2P)





When having a direct connection between the two endpoints, we can see that the MOS score is more stable and averages to about 3.7-3.8. Some calls were unsuccessful as can be seen on the graph. This means that opening up e.g. company firewalls for Skype means that better quality can be reached.

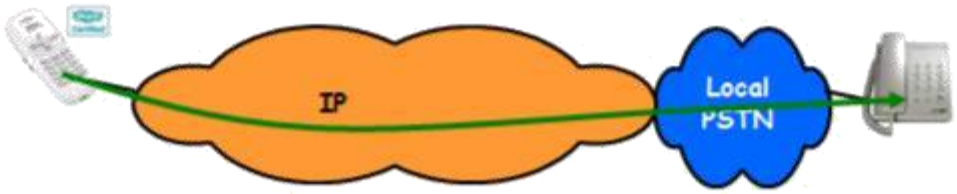
Delays: Telenet to ADSL (Direct)



Conclusions of the Skype Test

- In case of direct connection:
  - More or less stable MOS Score
  - Sometimes failed calls
- When P2P network is used:
  - More variable MOS Score
- When delays rise (P2P), a longer network path is used, so more jitter and packet loss is introduced. Therefore MOS scores are lower.
- Voice Quality is (in good circumstances) better than PSTN because a wideband codec is used. PSTN is limited to 4 kHz. The voice quality is limited by this narrowband coding in PSTN.

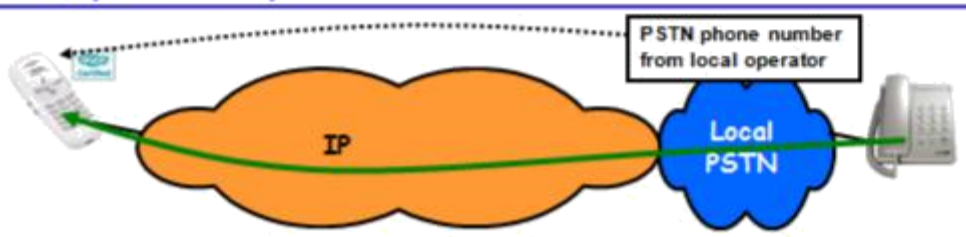
SkypeOut operation



- *SkypeOut* allows Skype users to call traditional telephone numbers, including mobile telephones, for a fee. This fee is as low as EUR 0.017 per minute for most developed countries. Beginning January 2007, Skype also charges an fixed connection fee of EUR 0.039 for each SkypeOut call, in addition to the ordinary rate. After 180 days of not making a SkypeOut call the Skype balance expires. As of January, 30th 2007, SkypeOut calls to Canada and The United States are no longer free.

SkypeOut operation

SkypeIn operation



- *SkypeIn* allows Skype users to receive calls on their computers dialed by regular phone subscribers to regular phone numbers. Permits users to subscribe to numbers in 21 countries (not including Belgium so far), among which: Denmark, Finland, France, Germany, Hungary, Ireland, Poland, Romania, Sweden, Switzerland, UK...
- For example, a user in San Francisco could create a local telephone number in Helsinki. Callers from Helsinki would pay only local rates to call that number.
- $50 a year (Feb 2009)

VoIP 135

http://about.skype.com/paidfeatures.html > "Online number"

http://www.skype.com/intl/en/allfeatures/onlinenumber/

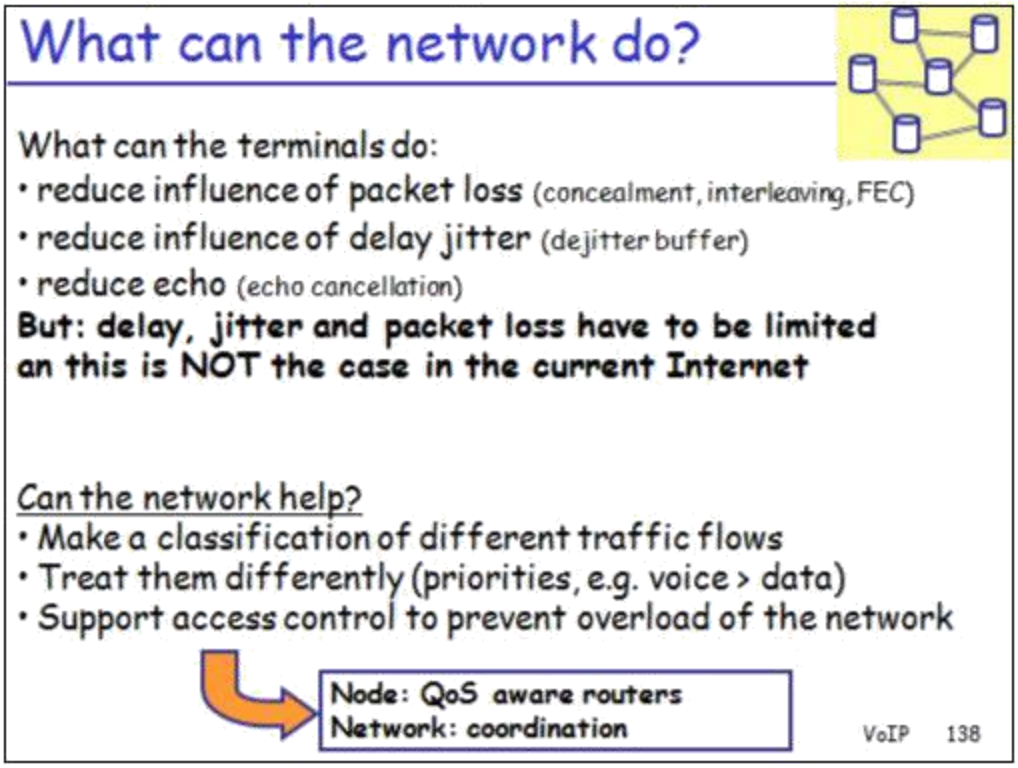
Conclusions

- Different codecs
  - High bitrate: almost no degradation (0.1 MOS)
  - Low bitrate: important decrease of quality (0.3 - 0.7 MOS)
- Packet loss
  - Faster degradation for higher compression (low bitrate codecs)
  - Larger packet sizes results in faster degradation
  - Typ. keep packet loss < 1%
- Delay
  - Typ. keep one way delay below 150 ms
- Delay jitter
  - Without dejitter buffer: devastating influence
  - With dejitter buffer: similar to packet loss
  - Typ. keep one way jitter < 30 ms
- Combination of packet loss and delay jitter

  tight margins to obtain acceptable quality

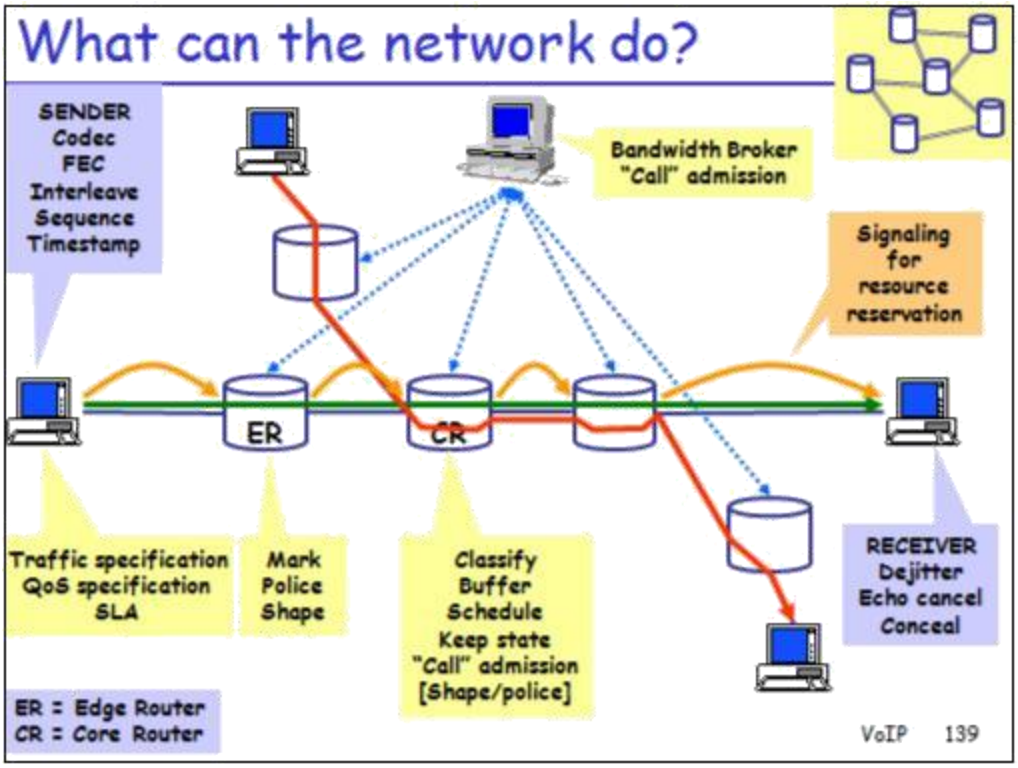
Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     Node (Qos Router) and overall network issues
4. Multimedia over IP



Because the terminals can only partly resolve the problems with delay, delay jitter and packet loss, other measures have to be taken by the network. The network will classify the traffic and treat the traffic classes differently (e.g. data traffic is not sensitive to delay or delay jitter, therefore voice traffic may have priority compared to normal data services). The network control may also limit the access to the network (C AC or Customer Access Control) in order to avoid excess delay or packet loss due to congestion in the network.

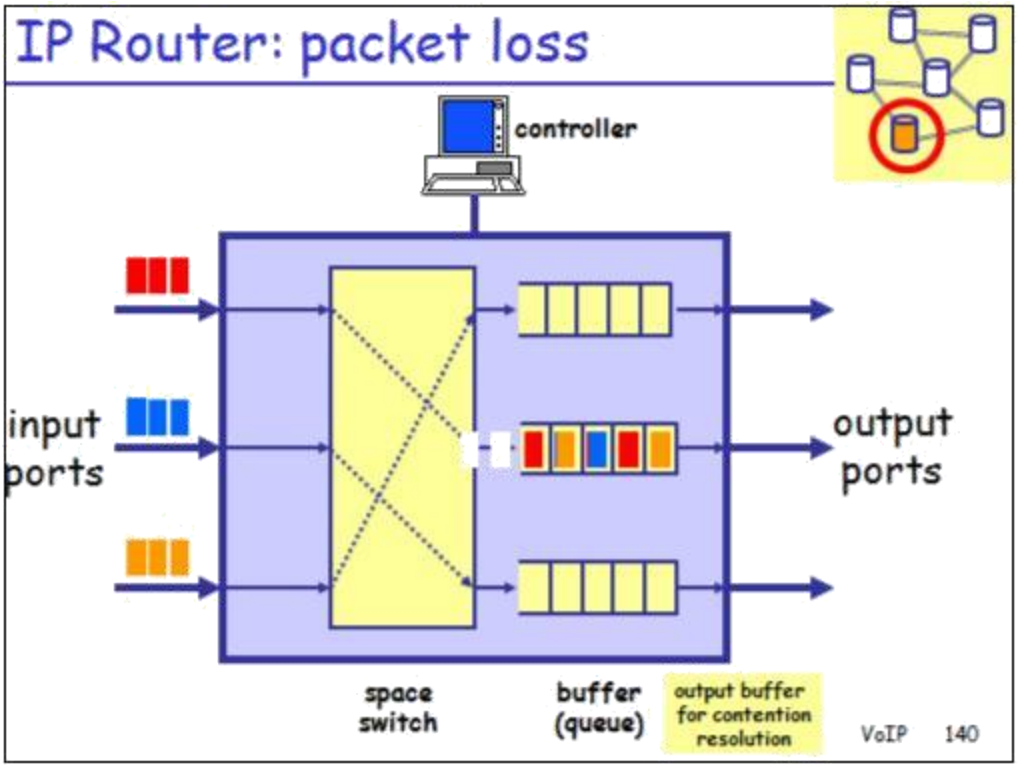
Two important aspects will be discussed: specific supporting techniques in the network nodes (routers) and global aspects related to network coordination.



The support of QoS re quires a lar ge number of basic buildi ng blocks s pread o ver t he net work and i n the terminals. The pre vious part has ill ustrated the major buil ding blocks used in the terminals (coding/ decoding, for ward error correction, interleaving, sequence and timestamp number, dejitter buffer, echo cancellation, packet conceal ment). Also the net wor k will (has to) contribute to the support of QoS. The important building blocks are:

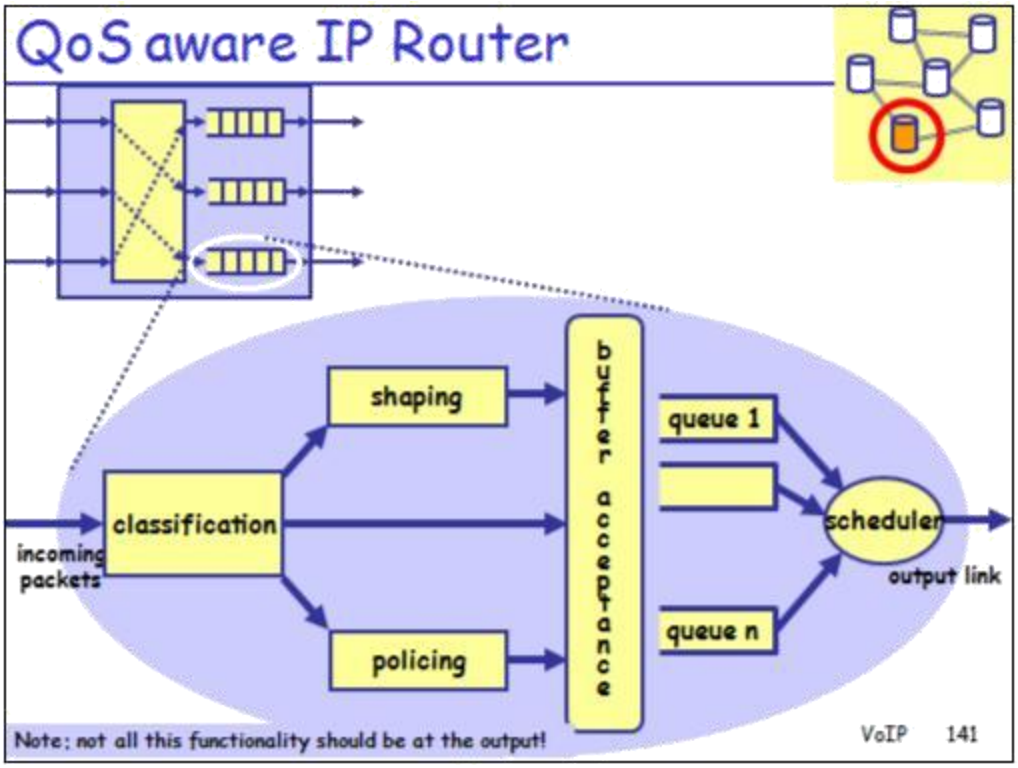
- The terminal has to specify the characteristics of the traffic that will be sent over the network (the Tspec or Traffic specification). This could e.g. include average bandwidth, maximum bandwidth, burst length. The terminal will also specify the requirements towards the network in order to support the specific flow, e.g. maximum delay, bandwidth (Rspec or Reservation specification). These specifications could be "written down" in a contract or SLA (Service Level Agreement). Note that such an SLA could be made up in a dynamic way (fast and eventually for a short period) or more statically (for a longer period).
- In the edge router (access router) packets may be marked (could also be done in the terminal!) in order to indicate the traffic class. Policing will check if the source is sending packets in accordance to the traffic contract (e.g. is the bandwidth used not larger than the specified bandwidth) and if not, actions may be taken. The edge router may also try to smoothen the traffic (using a shaper) in order to make the traffic more "network friendly" (e.g. bursts of traffic will degrade the network performance).
- Inside the network (in the core routers) classification is required in order to distinguish different traffic flows. Based on this classification, the flows will be treated differently. This is possible by using different queues that will be served with different priorities (scheduler). Also the buffer management strategy is important in that respect (this is the way packets are dropped when the buffer is getting full). The nodes have to keep state (information) about the different flows and eventually they may refuse new traffic flows ("call" admission). Shaping and policing may also be applied.
- In some cases there will be a central system ("Bandwidth Broker") that will take care of the acceptance of new flows. This will be important in order to optimize the network resources and in order to guarantee the quality in the network
- Finally a signaling protocol will be necessary to reserve the resources in the network.

Note: A distinction may be made based o n a limited number of traffic classes (used i n DiffSer v) or based on a l arge number of i ndi vidual flo ws i n the net work (used i n IntSer v) . The latter will have scalability problems in the network.



This figure illustrates (see also previous slide) the basic architecture of an IP router. It consists of incoming interfaces (input lines or ports) and outgoing interfaces (output lines or ports), a space switch (to transfer an incoming packet to the correct output interface) and a buffer in order to resolve contention (multiple packets contending for the same output link will have to wait for each other because only one packet is allowed at a time on the output line). In addition there is a controller that will take care of the control of the space switch, the buffer management, the forwarding (including routing table look- up), the routing table update (using routing protocols), etc.

The example illustrates the problem of packet loss: multiple packets from different input ports go to the same output port (middle) and this for several (3) consecutive timeslots. This will result in output buffer filling and at a certain point the buffer will be full and additional arriving packets will be discarded (lost).

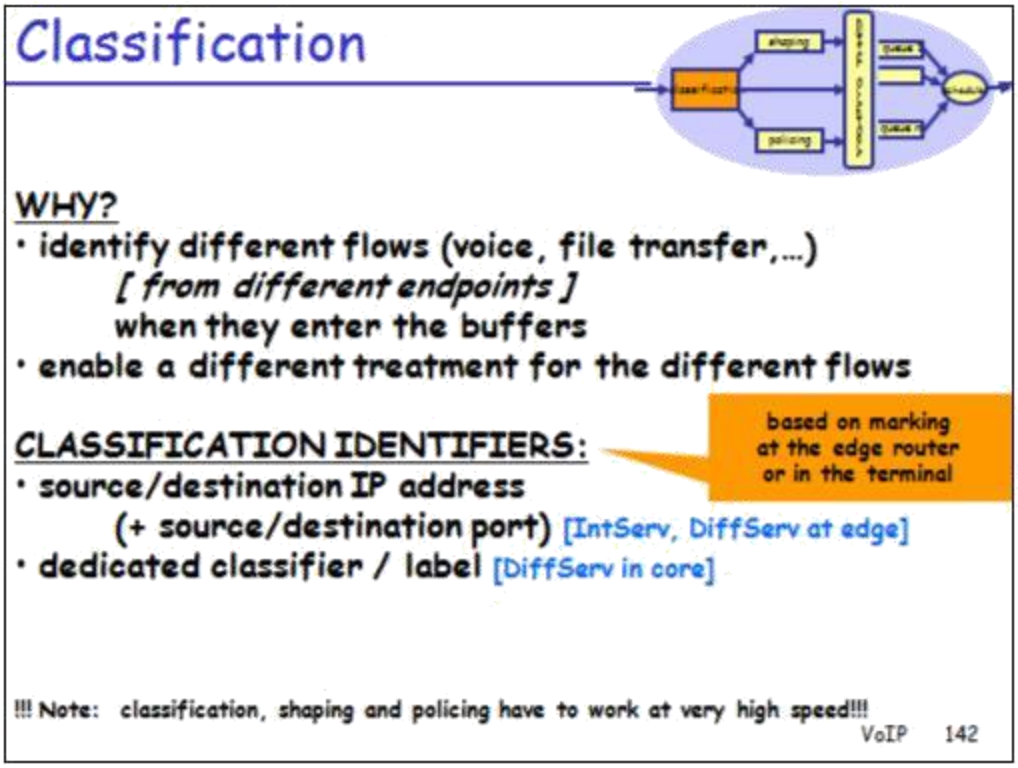


The typical building blocks of a QoS-aware IP router are depicted in the figure above.

Classification means the identification of the flows. The flows will be either shaped or policed (or go directly to the buffers). The packets are then stored in one of the queues depending on the desired QoS. Each queue can for example correspond to a service category (e.g. real time streaming video or non real time data). The scheduler then determines which queue is allowed to send packets to the output port. Buffer acceptance will be important in case the buffers are filling up. It will be responsible for discarding some packets.

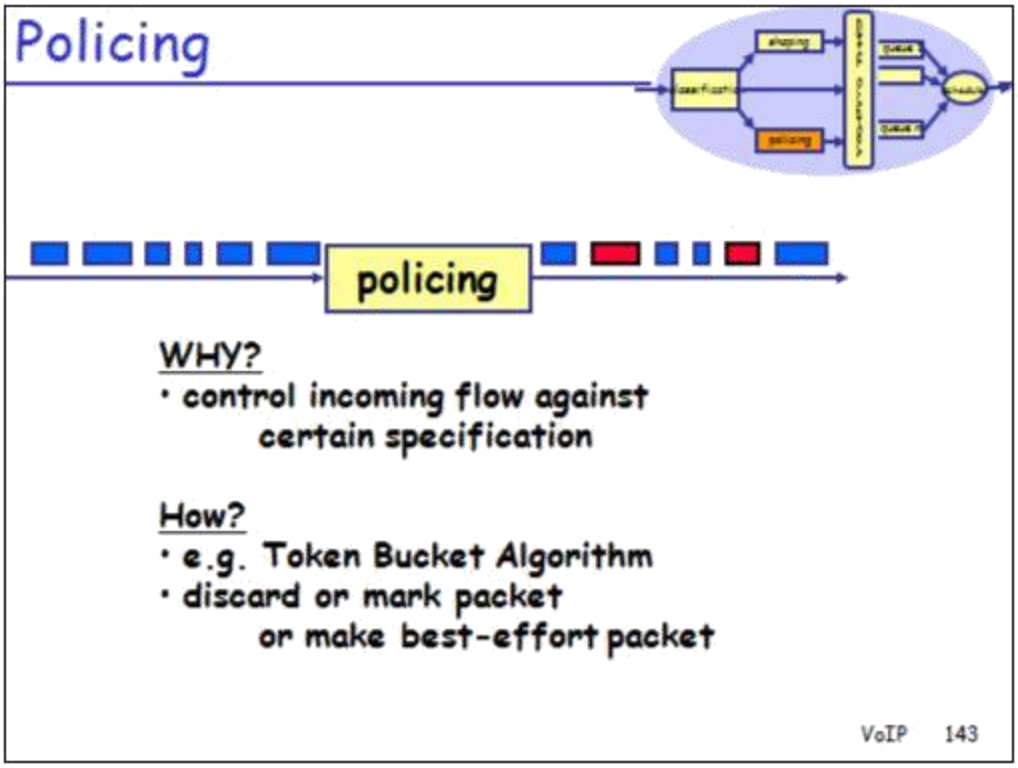
The next slides will illustrate each of these building blocks.

Note: the building blocks do not necessarily have to be at the output, part of the classification could be done at the inlet (e.g. if forwarding decision is dependent on traffic class).



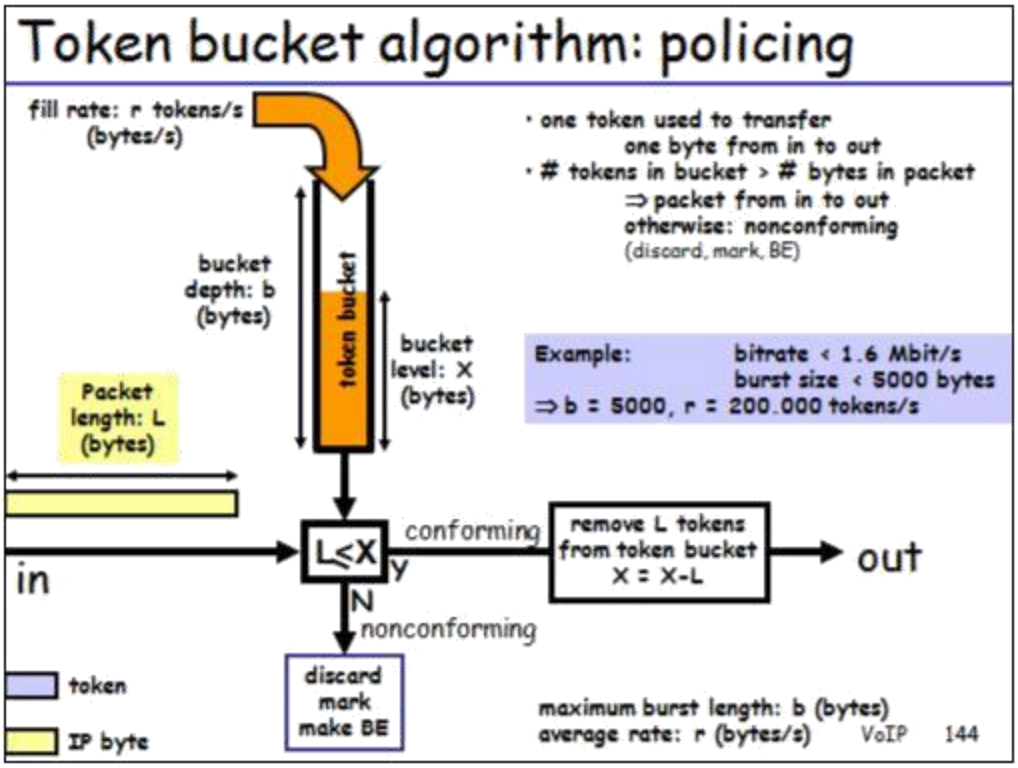
Classification is very important in order to identify different packet flows in the network. Differentiation could be based on the type of traffic (traffic classes, e.g. voice or data) but could also be based on a differentiation between the endpoints. In this way the flows can be treated differently inside the router (e.g. priority).

Classification can be done using a specific classifier (e.g. in the TOS byte of the IP header) or label, as will be discussed in the Diffserv case. Marking (= filling in the correct classifier value) may be done at the edge of the network. It is also possible to classify based on the IP address(es) (of source and/or destination), on the port number(s) (of source and/or destination) or on the combination of address and port numbers.



Policing will be important in order to check if an incoming flow conforms to a certain specification. A user may ask the network a certain bandwidth and the network may accept this (e.g. if there is still enough bandwidth available).

Policing can be done using the token bucket algorithm. If IP packets are not conform to the specification (e.g. bitrate too high which means that packets are arriving too fast), they may be discarded or marked (with a lower priority) or just made best-effort packets (= lowest priority).



The token bucket algorithm works as follows:

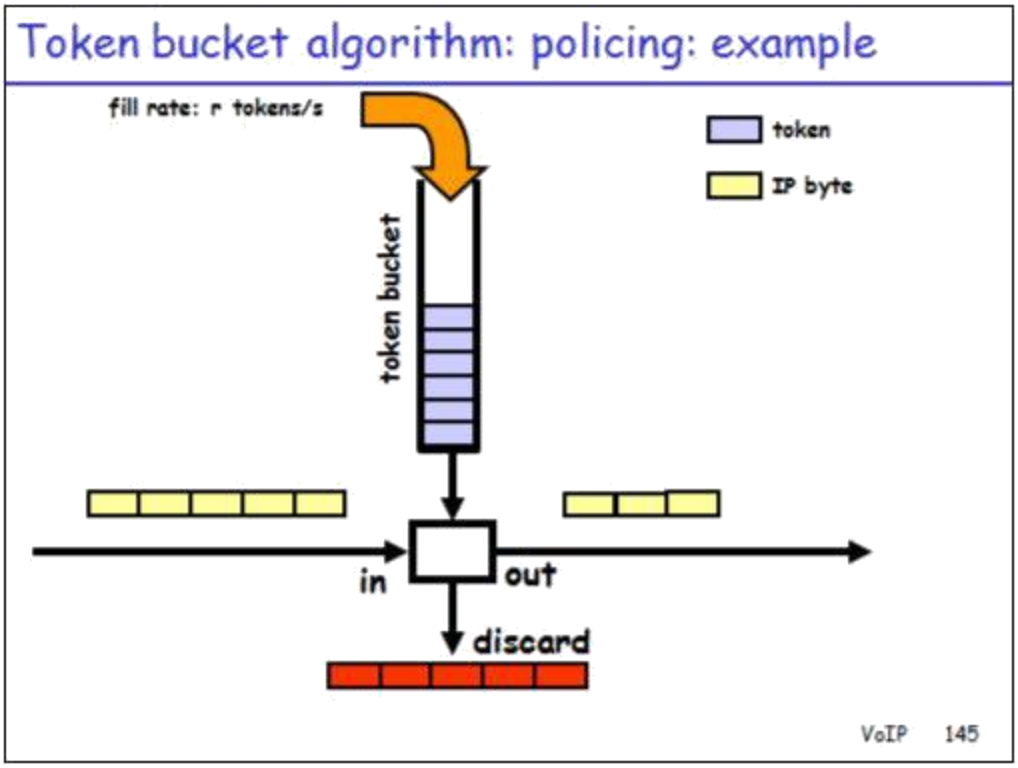
A token bucket is filled up at a certain rate (e.g. r tokens/sec) and it has a certain depth. The tokens will be used to control the transfer of incoming IP packets (with a byte scale resolution). When a packet arrives with a length of 1000 bytes and there are 1500 tokens in the bucket (bucket level X = 1500), the packet is transferred to the output and the number of tokens in the bucket is reduced by the amount of bytes transferred to the output (1000 bytes). The new value is now X=500. The token bucket is however filled again. When a packet arrives with a length of 2000 bytes and the bucket has only 1500 tokens (X=1500), then the packet will be discarded or marked or made best-effort. In this way it is possible to control the average rate of a certain flow.

Note that the bucket depth will limit the maximum number of tokens available (to a maximum b). This will also limit the maximum burst length that will be allowed. Suppose that for some time no packets had arrived and the bucket is completely full (the bucket depth b is 10000). Suppose also that a burst of 15 packets arrives with a length of 1000 bytes for each packet. In that case the first 10 packets will be transferred without any problem (corresponding to a total of 10000 bytes) but the next 5 packets will be discarded or marked or made best effort.

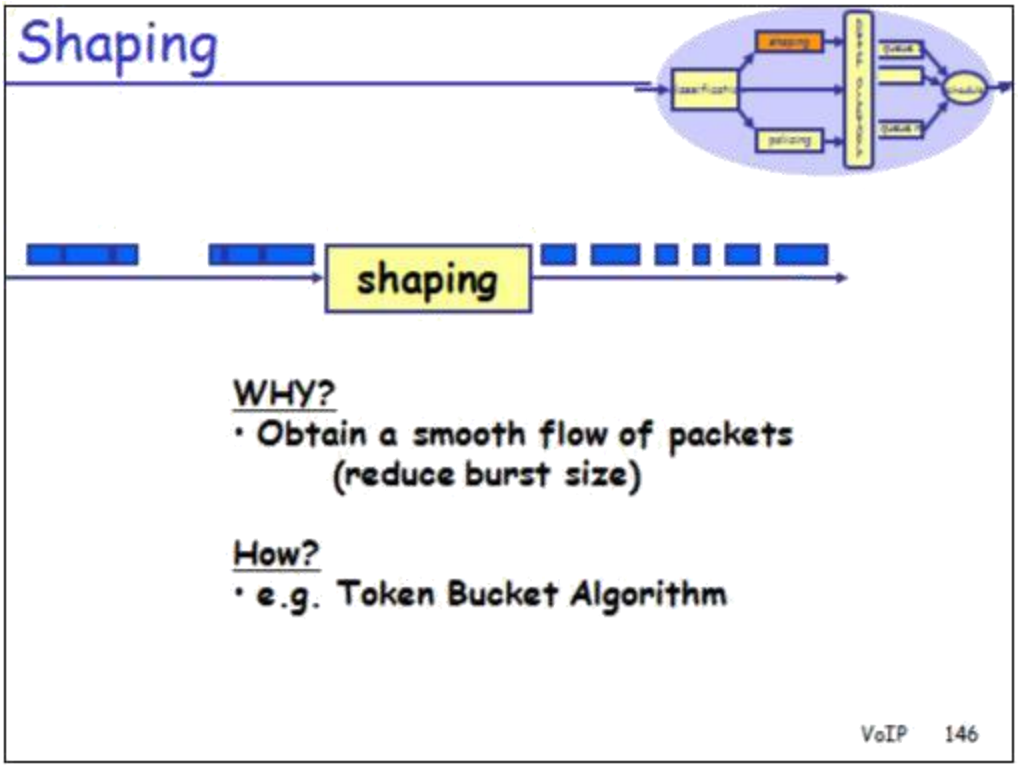
Example: suppose one wants to limit the speed of a certain flow to 1.6 Mbit/s and the maximum allowed burst size to 5000 bytes. In that case one will use the following parameters:

r = 1.6 Mbit/s / 8 bits = 200.000 tokens/s

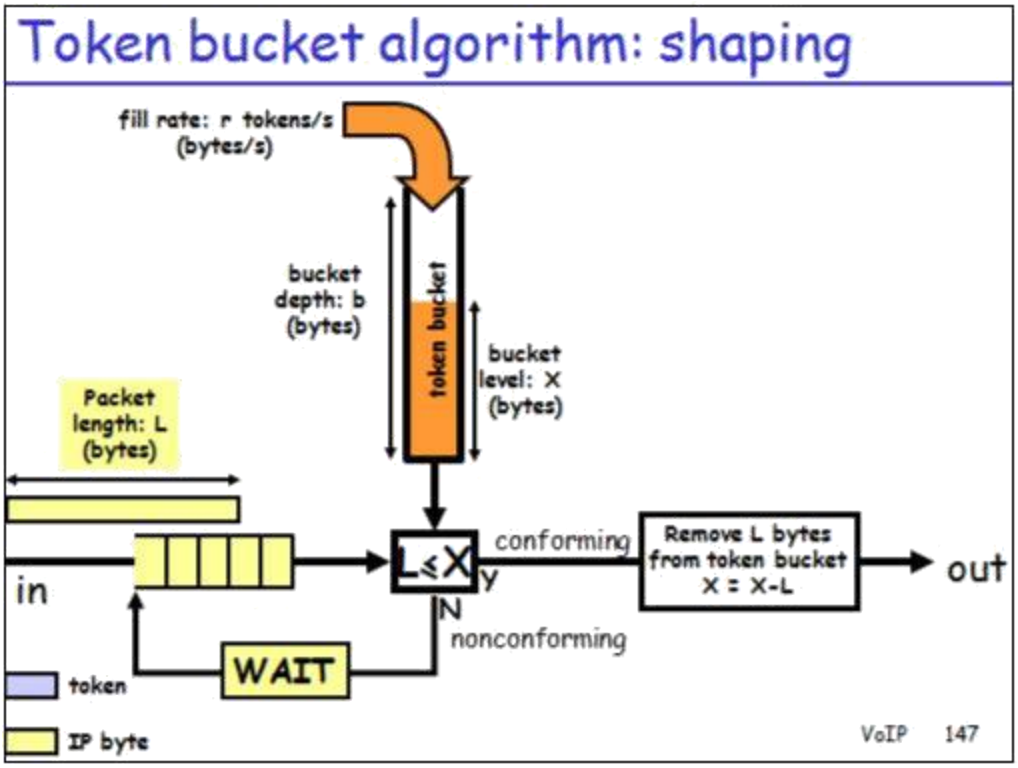
b = 5000



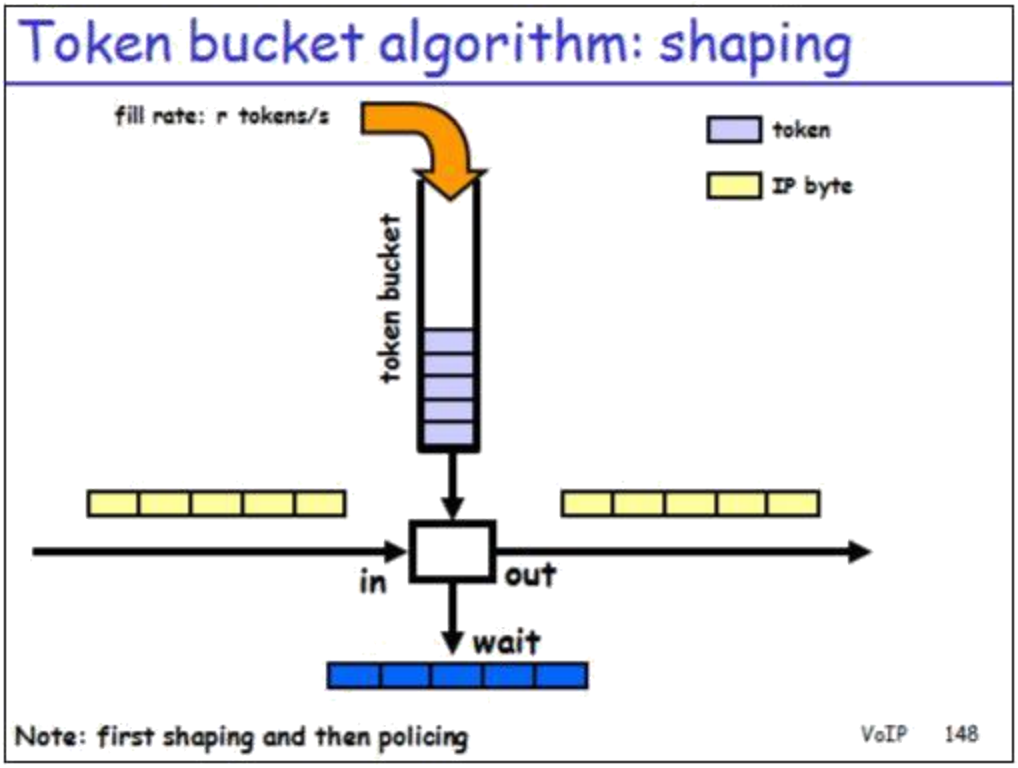
An example of policing is illustrated.



Another mechanism used in a router is shaping of the traffic flows. This will reduce the possible strong variation in packet rate or bitrate (e.g. it may limit the burst size, without however discarding any packets). This will also be based on the token bucket algorithm.

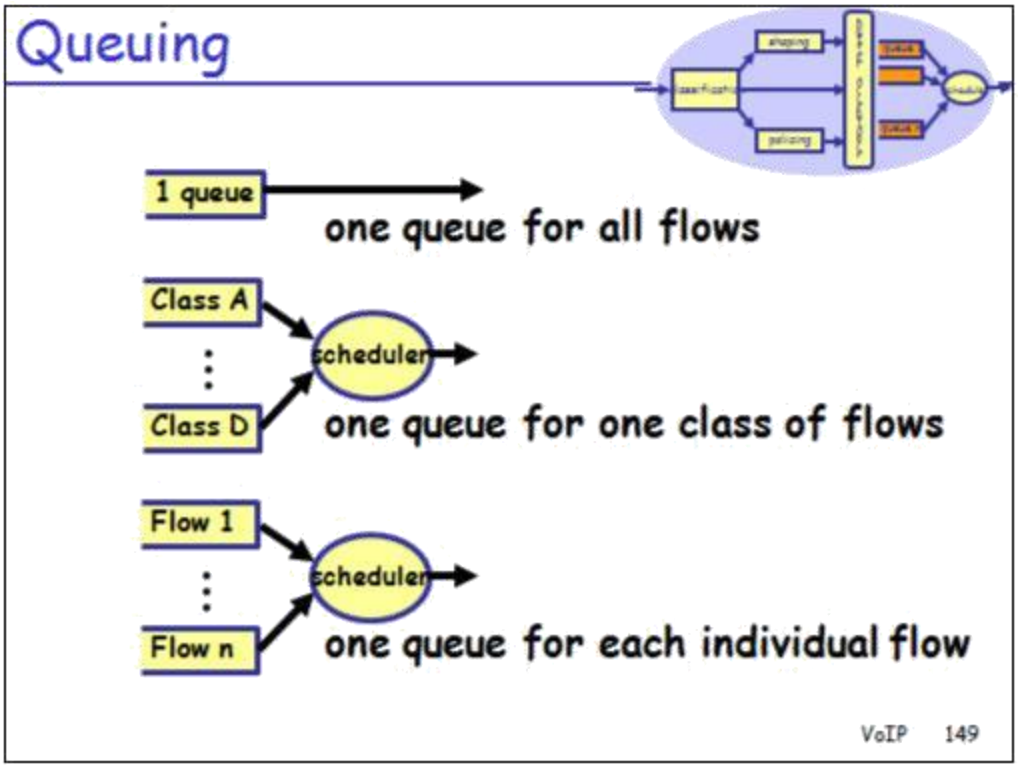


The major difference with the token bucket algorithm used for policing is that non conforming packets are delayed until enough tokens are available in the token bucket. This will effectively smooth out the traffic.

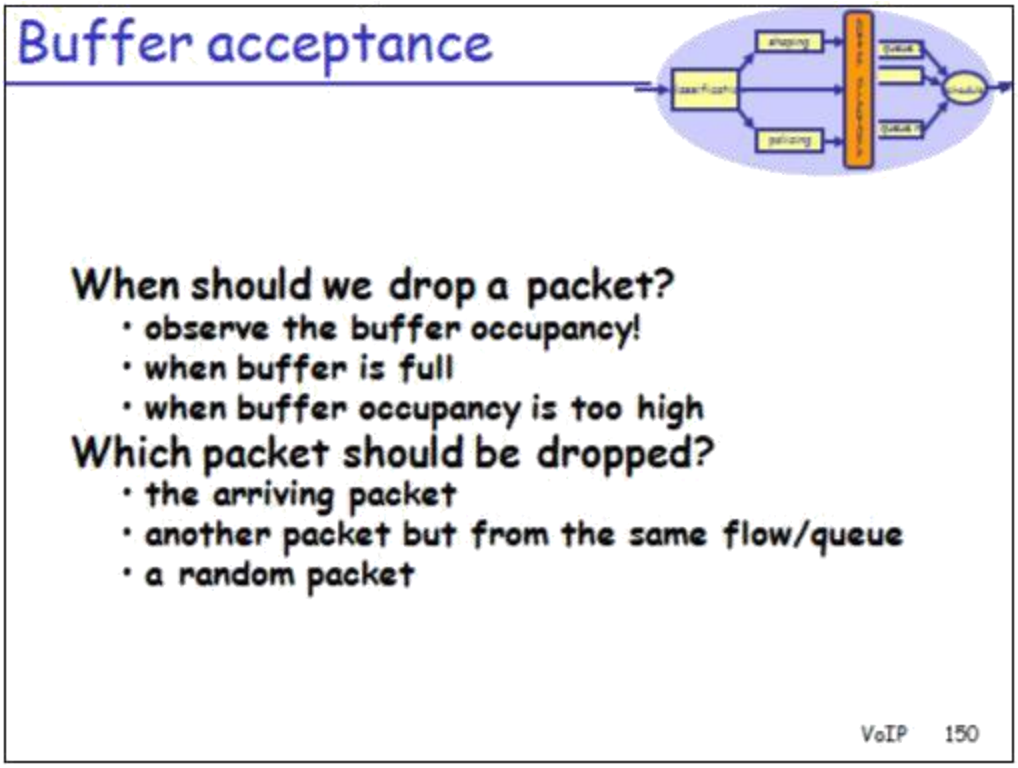


An example of shaping is illustrated.

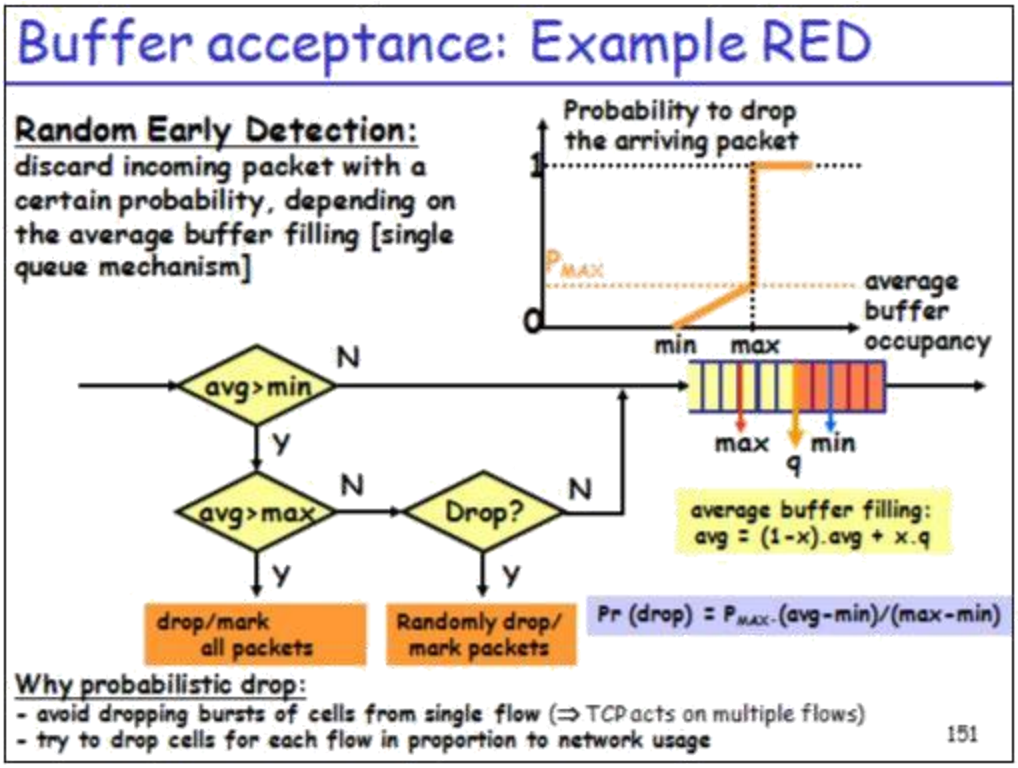
In many cases a traffic flow will first be shaped before entering a policer.



After policing and/or shaping (note that the two may be combined!), the packets have to be buffered. Different possibilities exist: use of a single buffer (queue), use of a queue per class of information flow (e.g. voice versus data) or use of a queue for each individual flow (e.g. for each voice connection transiting the router). When multiple queues are used for the same output, a scheduler is required to decide which packet (from which queue) will be sent to the output line.

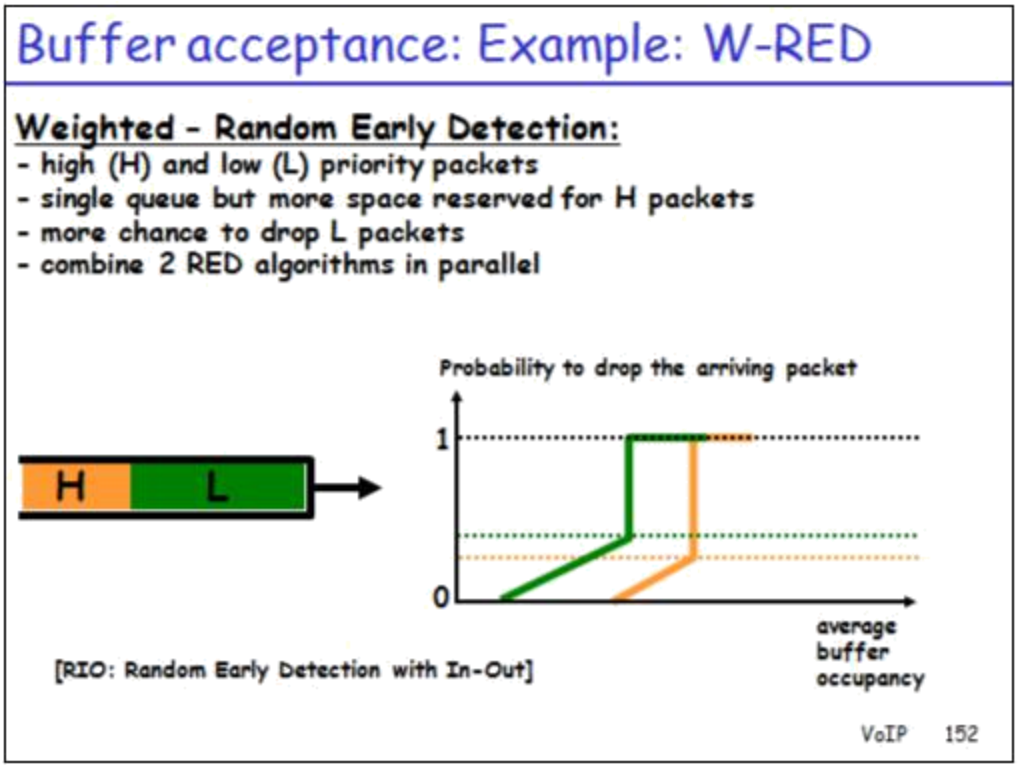


When too many packets arrive in the buffers, measures have to be taken (discard packets). The decision on when a packet should be discarded is based on the observation of the buffer occupancy (buffer completely full or above a certain threshold value). This can be done in a straightforward manner: drop the arriving packets when the buffer is full. But it may be much better to drop a random packet or a packet from the same flow or queue.



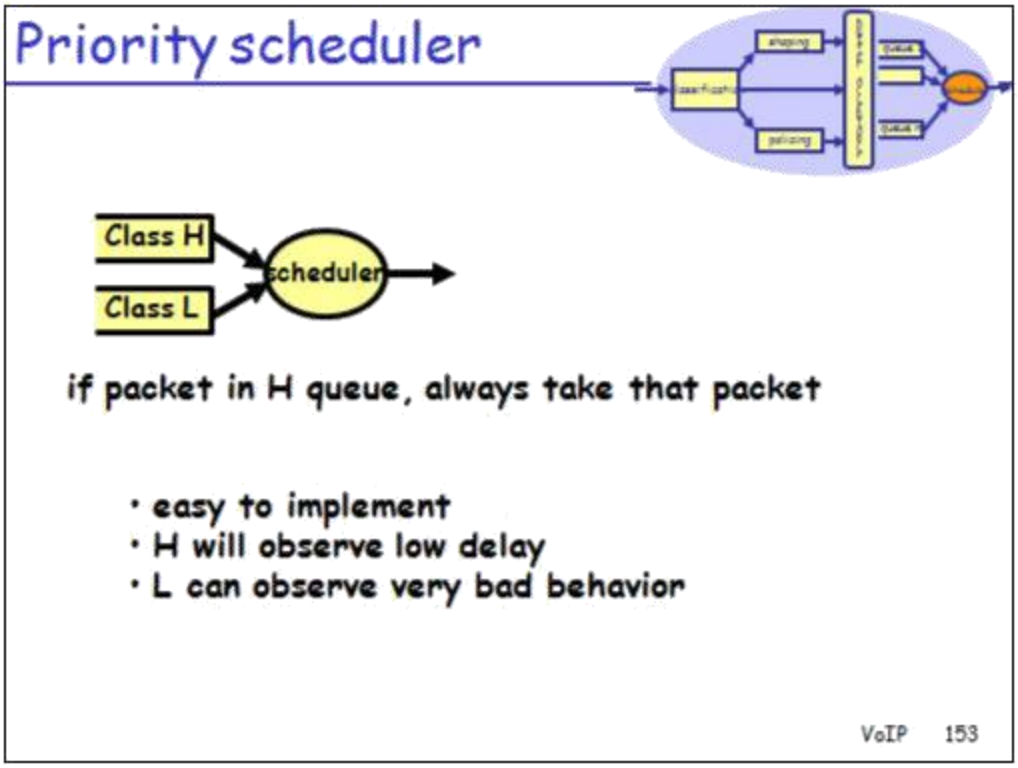
A first example of buffer acceptance strategy is RED or Random Early Detection. In this case the buffer filling will be measured continuously (an average value (avg) will be calculated). Two thresholds will be used: min and max. The incoming packets will be accepted if the average buffer filling is below the min threshold. The incoming packets will be discarded with a certain (increasing) probability if the average filling is between min and max. If the average filling is above the max threshold, all packets will be discarded.

This probabilistic dropping will improve the network performance. By randomly dropping packets from different flows, different sources will reduce their sending rates due to the TCP behavior (slow start, fast retransmit). Random dropping will also result in a dropping rate proportional to the network usage of the different flows.



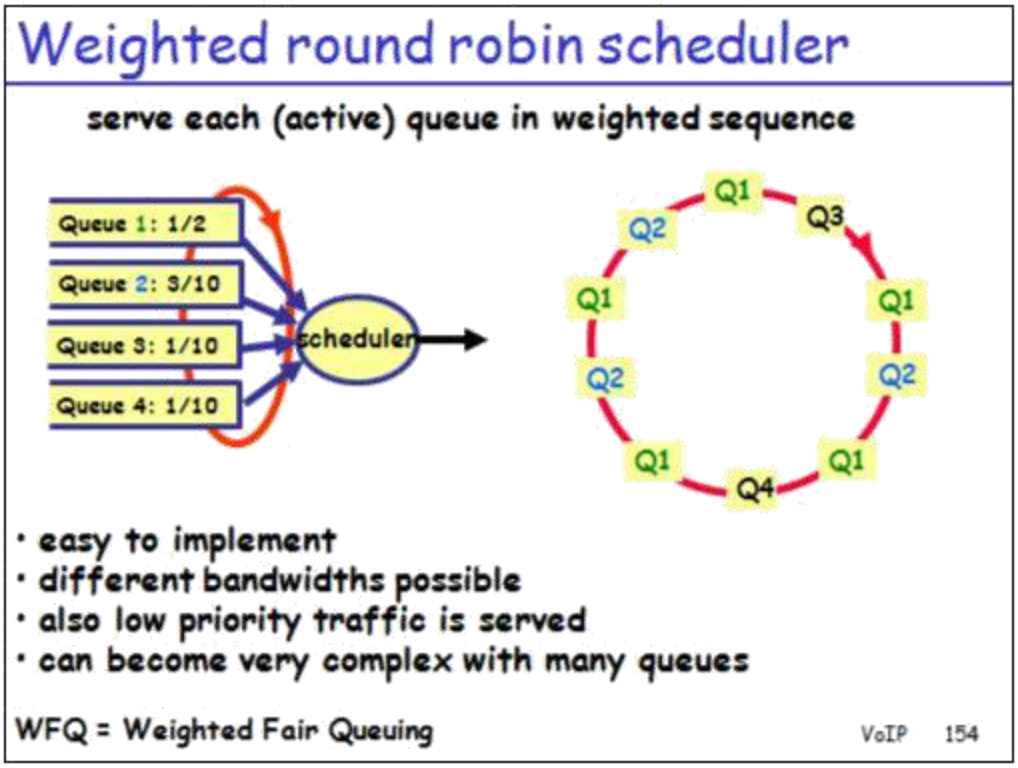
Weighted RED will (e.g.) use two traffic classes: high and low priority packets. Both will have different min and max values, different PMAX and different slopes. The net result is that low priority packets have a higher chance to be dropped.

[RIO : Differentiates between in/out-profile packets. It maintains two average queue length values, one considering only in-profile packets, and one with both in- and out-profile packets. As in W-RED it has different tresholds, namely for both in- and out-profile classes.]



The scheduler will indicate which queue is allowed to send a packet to the outlet (in case of multiple queues).

The example illustrates a very simple scheduler for two queues (high and low priority). If there is a packet in the H queue, the scheduler will always give priority to that queue. It is only in case the H queue is empty that the L queue is allowed to send packets to the outlet. The major advantage is that this is a very simple scheduler and that the high priority traffic will observe a very good treatment. The low priority traffic on the other hand may observe a very bad behavior. In case of a large volume of high priority traffic, the low priority traffic will be completely blocked, resulting in packet loss (because the buffers have a finite depth).

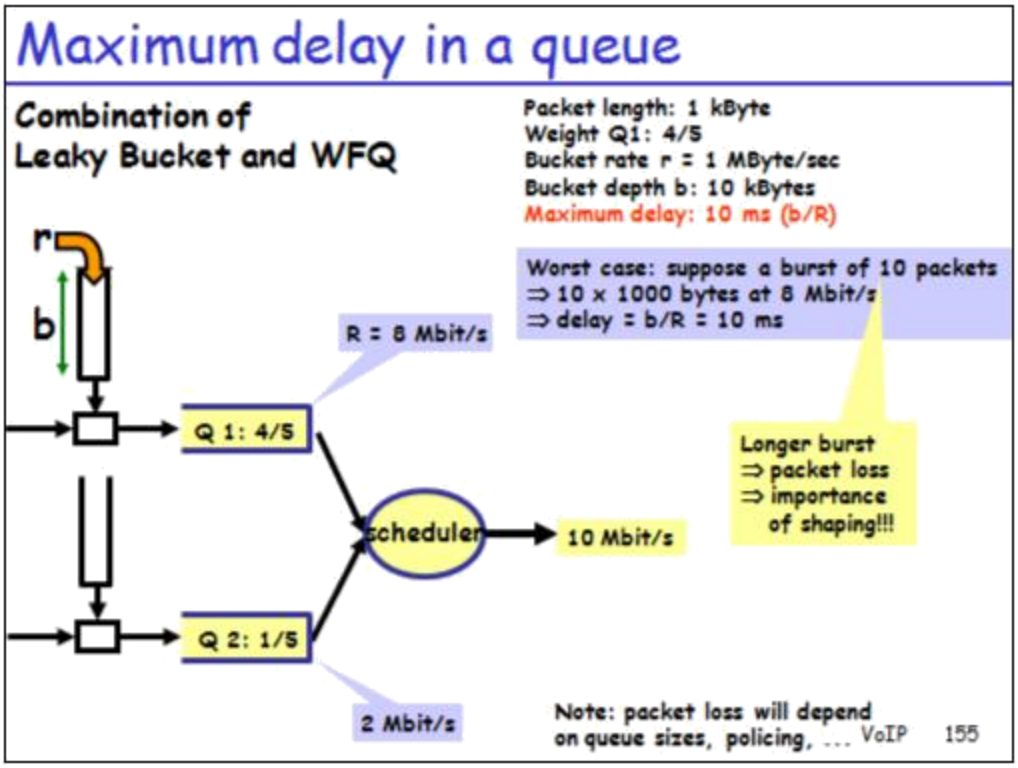


A more advanced scheduling algorithm is weighted round robin. Each queue will have a certain weight (corresponding to its relative priority) and the queues will be served based on their weighting factors. The example illustrates 4 queues with weight: 5/10, 3/10, 1/10 and 1/10. It is a scheduling algorithm which is easy to implement if the number of queues is limited and it allows to differentiate different traffic flows. Also the low priority traffic will be served.

A special case is the round robin scheduler where all the weights are equal.

Note: if there is a queue without a packet then the scheduler will immediately move to the next queue ("work-conserving" mode of operation).

WFQ: Weighted Fair Queuing: more advanced scheduling mechanism with same goal (i.e. allocating each stream a share of available bandwidth), but designed for variable length packets (whereas WRR originated for fixed size-packets).



(Note: if R<r, then max delay is dependent on Q1 length)

It is very important to be able to guarantee certain parameters in the network. A very important parameter is the maximum delay in the network. An upper- limit on delay is obtainable when combining the leaky bucket algorithm with the weighted fair queuing scheduling discipline.

An example is given in the figure. It shows an output line with a bandwidth of 10 Mbit/s. Using WFQ it is possible to assign a bandwidth of 8 Mbit/s (=1 Mbyte/s) to the flow in queue 1 (Q1). The use of a leaky bucket in front of the queue will result in a maximum delay through this node. Suppose the leaky bucket uses the following parameters: token rate r = 1 Mbyte/s and bucket depth = 10 kByte. We assume that all packets have a length of 1 kByte. The worst case is when a burst of 10 packets arrives and at that time the bucket is full. In this case all packets will be transferred to the output queue (Q1) and in this queue the last packet has to wait till all other packets are transmitted on the outgoing link (with a bitrate of 4/5 of 10 Mbit/s). The last bit of that last packet will be delayed by 10 times 1000 bytes transmitted at 1 Mbyte/s, corresponding to a delay of 10 ms.

By adapting the parameters, it is possible to change the maximum delay in the node.

Note that the leaky bucket may be used as a shaper in order to avoid dropping of the packets.

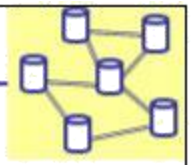
Note that packet loss is another important parameter in the network. This will depend on many factors such as queue lengths, buffer management, leaky bucket parameters, "call" acceptance, etc.

Note that the example is simplified because packets have different lengths, burst are not arriving in zero time, etc.

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     - Node and Overall network issues
       - Intserv
       - Diffserv
4. Multimedia over IP

Remaining questions!!!



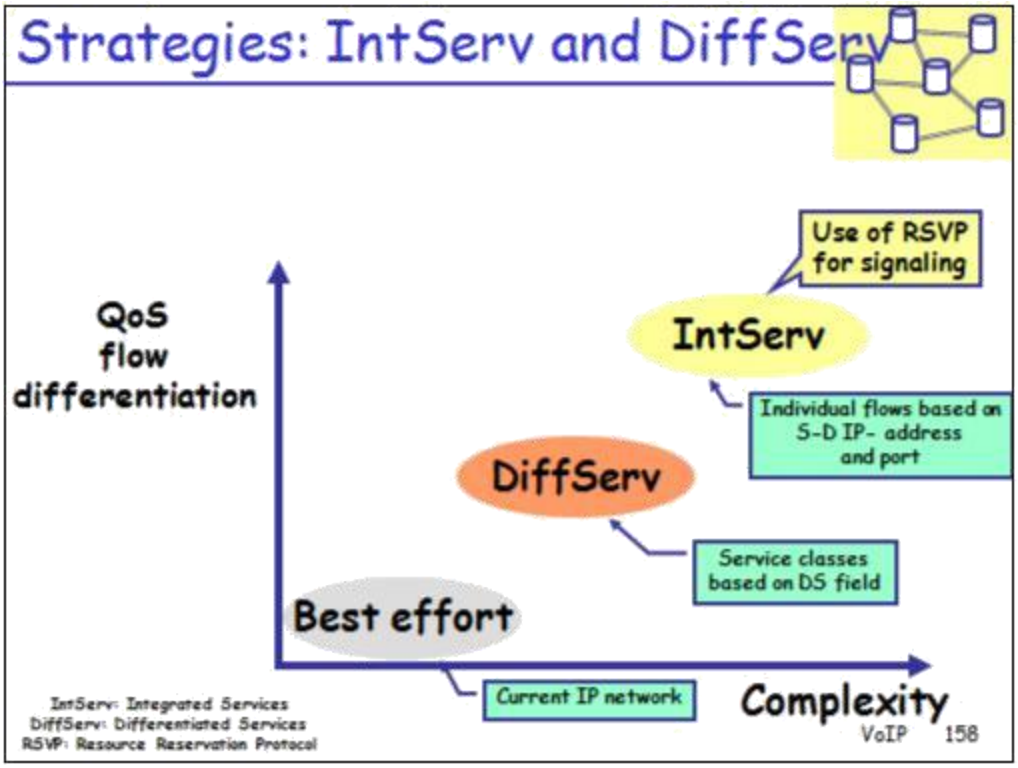
How to combine all these basic building blocks!

How to specify resources?
How to reserve resources?
How to admit additional flows?
How to distribute this information in the network?
How to assign the markers?
How to configure the routers?
(scheduling strategy, queue lengths, leaky bucket parameters, ...)
...

Two IETF architectures:
Integrated Services and Differentiated Services

VoIP 157

A number of important questions are still not resolved. Two architectures will be discussed: Integrated Services and Differentiated Services.



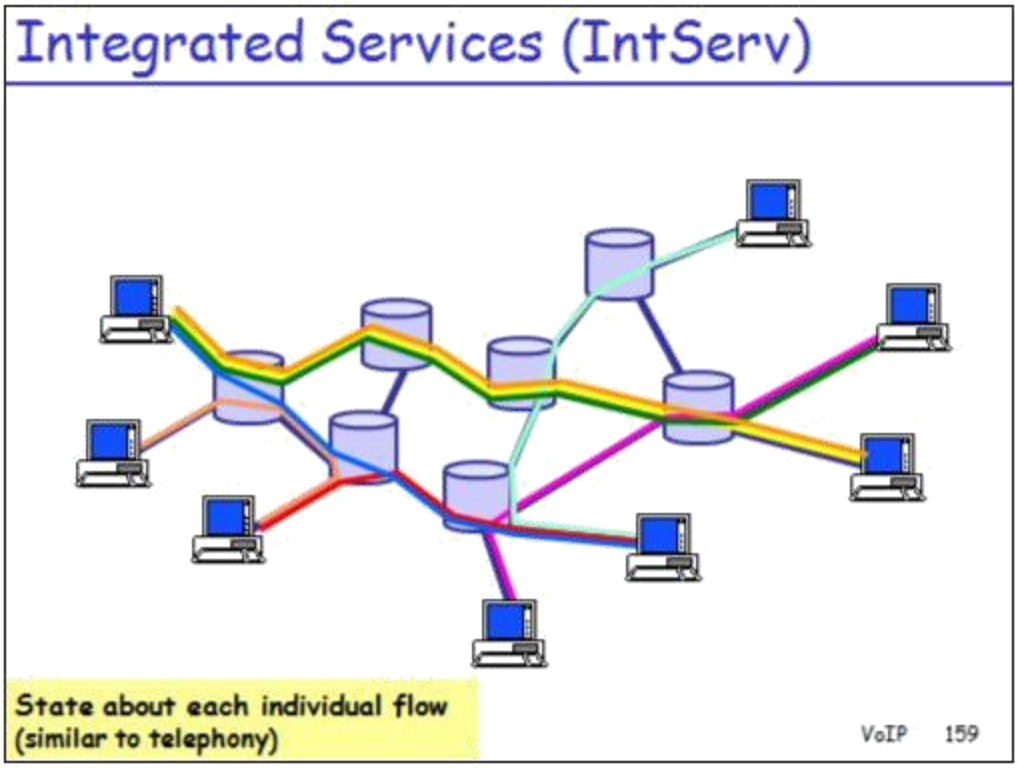
There are two architectures presented in the IETF in order to support QoS in the network: IntServ (Integrated Services) and DiffServ (Differentiated Services). The figure illustrates the complexity and the flow differentiation.

In case of a Best Effort (BE) internet, no differentiation is made between the different traffic flows. They are al treated in the same way and complexity is very low.

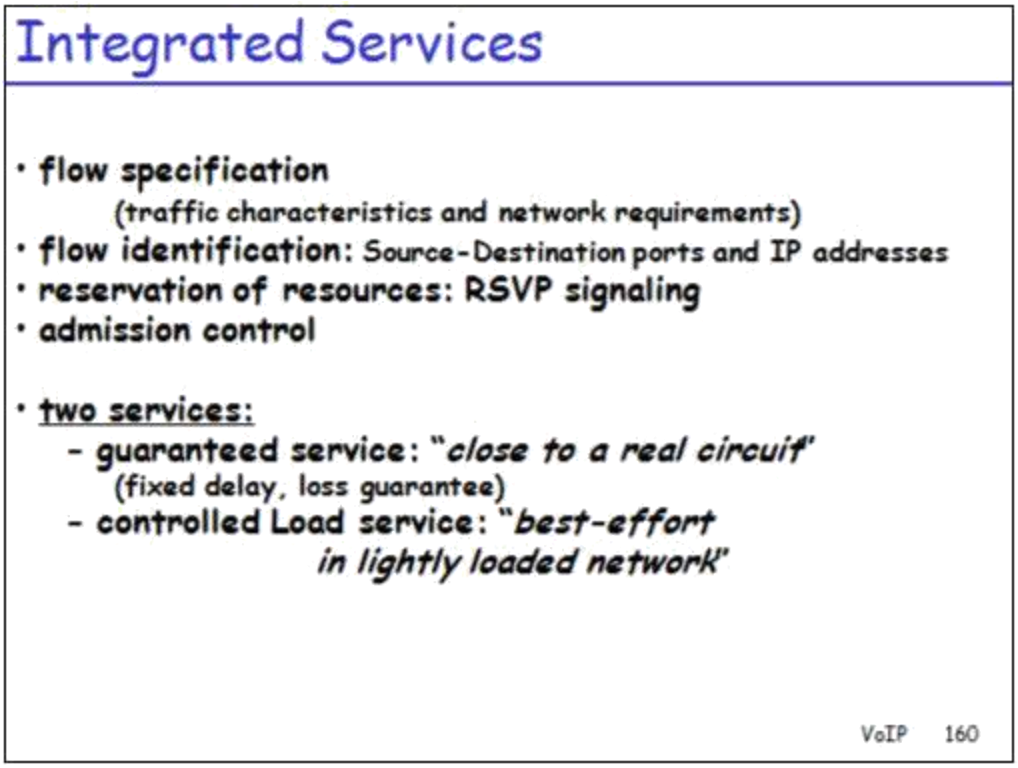
In case of Differentiated Services (DiffServ) one will introduce a number of service classes with different requirements. The different service classes will be identified by using the TOS (Type Of Service) byte in the IP header (the DSCP or DiffServ Code Point). This approach requires only a limited amount of state in the routers (state related to each service class).

In case of Integrated Services (IntServ) one will treat the individual flows separately, resulting in a large amount of state in the routers (for each flow the router has to maintain state). This architecture results in general in more flexibility and tailoring towards specific needs of individual flows, but on the other hand it has a big problem with scalability (due to the large amount of state). IntServ will require extensive signaling in the network. This will be based on RSVP (Resource Reservation Protocol).

Note: Today, IntServ is only considered as a viable solution for small (typical local or access) networks. Most interest goes to the DiffServ architectures. From a conceptual point of view, it is interesting to describe both.



This figure illustrates the IntServ approach: all individual flows have to be treated separately in the network. Each router where the flow is transiting will keep some information (state) about this specific flow.



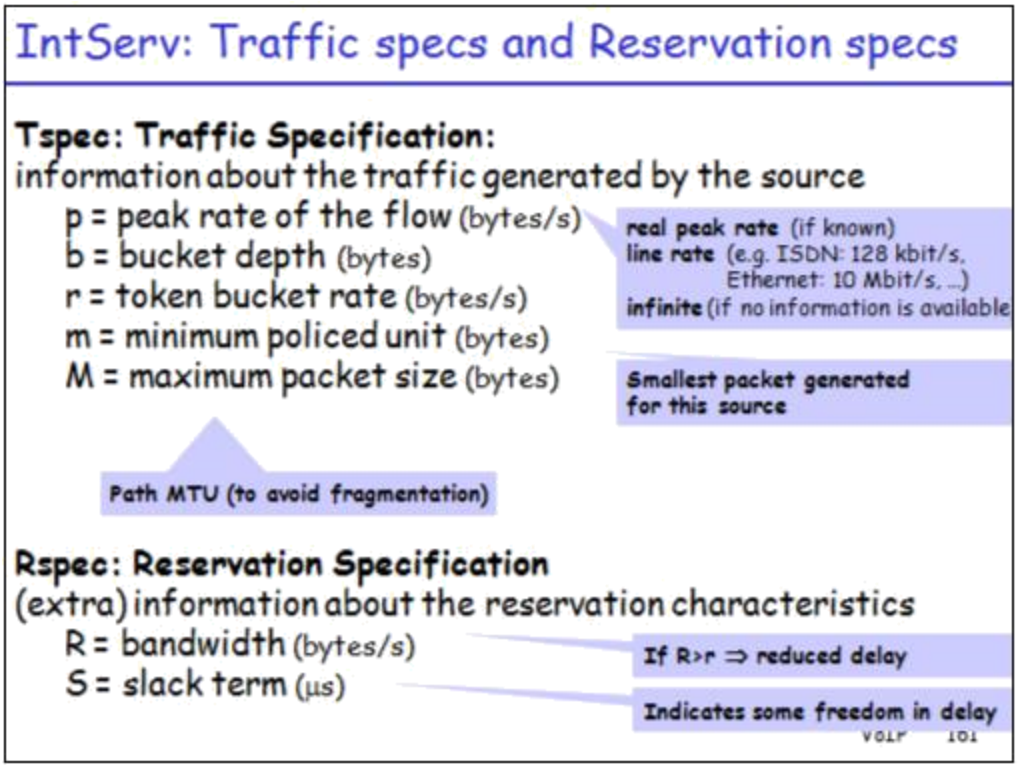
Before resources can be reserved in the network, it is important to specify them. This will be done by specifying the traffic characteristics on one hand and the network requirements on the other hand.

A flow will be identified by the source and destination IP address and the source and destination port number.

Using the specifications for the required QoS, a signalling protocol is required in order to set up and configure the queues and scheduling mechanisms in the different routers along the path between source and destination.

The routers along a path between source and destination have to decide whether they still have enough resources to accept a certain reservation.

In IntServ two important service classes have been defined: guaranteed service and controlled load service. Each traffic flow will have the choice between these two service classes (and each traffic flow will be able to set some parameters specific to their needs). If none of the service classes is used, traffic is considered as best-effort (as in the current internet).

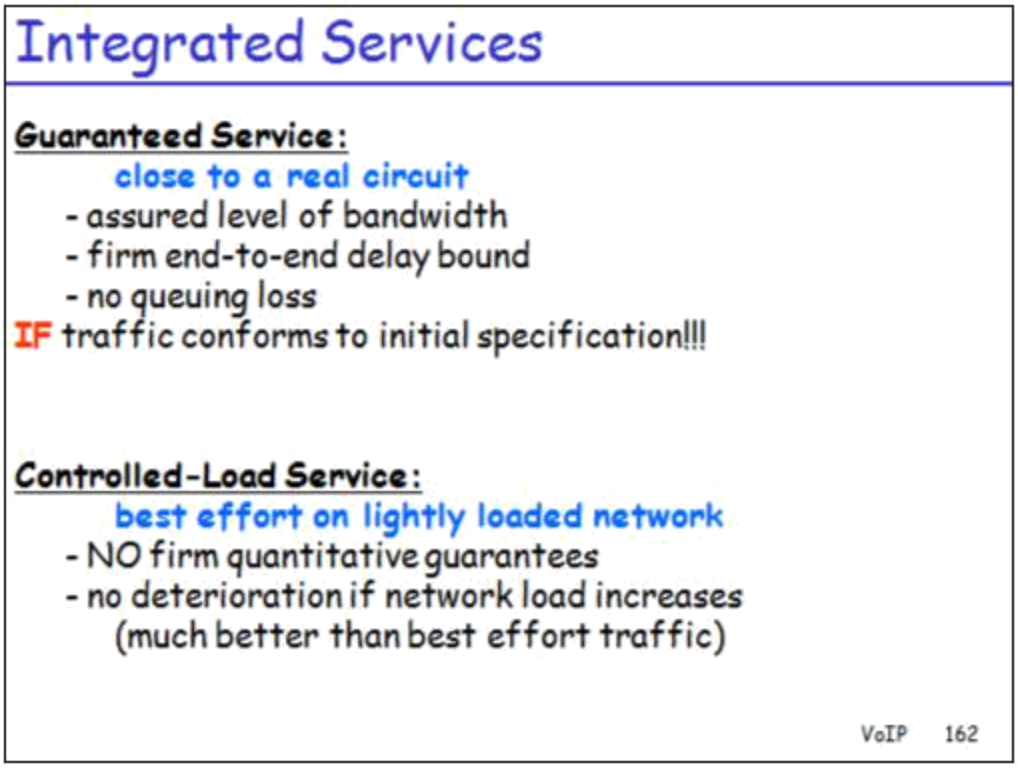


Traffic is specified by using the Tspec. This Tspec has the following parameters:

- p = peak rate: this could be the real peak rate of the application, it could be the bitrate of the outgoing link from the terminal (e.g. 10 Mbit/s for Ethernet or 128 kbit/s for ISDN) or it could be infinite (if no reasonable value is available)
- b = bucket depth (bytes): this forms a measure for the maximum burst length of the generated traffic (note that generated traffic that is not conform with this maximum burst length may be smoothened using a shaper with the appropriate parameters)
- r = token rate (bytes/s): this forms a measure for the average sending rate of the source
- m = minimum policed unit (bytes): this is the minimum packet size that will be used by the application. This is important because large amounts of small packets are much more loading the network (due to the header processing overhead) than a smaller amount of long packets (with the same overall bitrate)
- M = maximum packet size (bytes): this is the maximum packet size that will be sent (it will be important to keep M smaller than the MTU or Maximum Transfer Unit in order to avoid fragmentation).

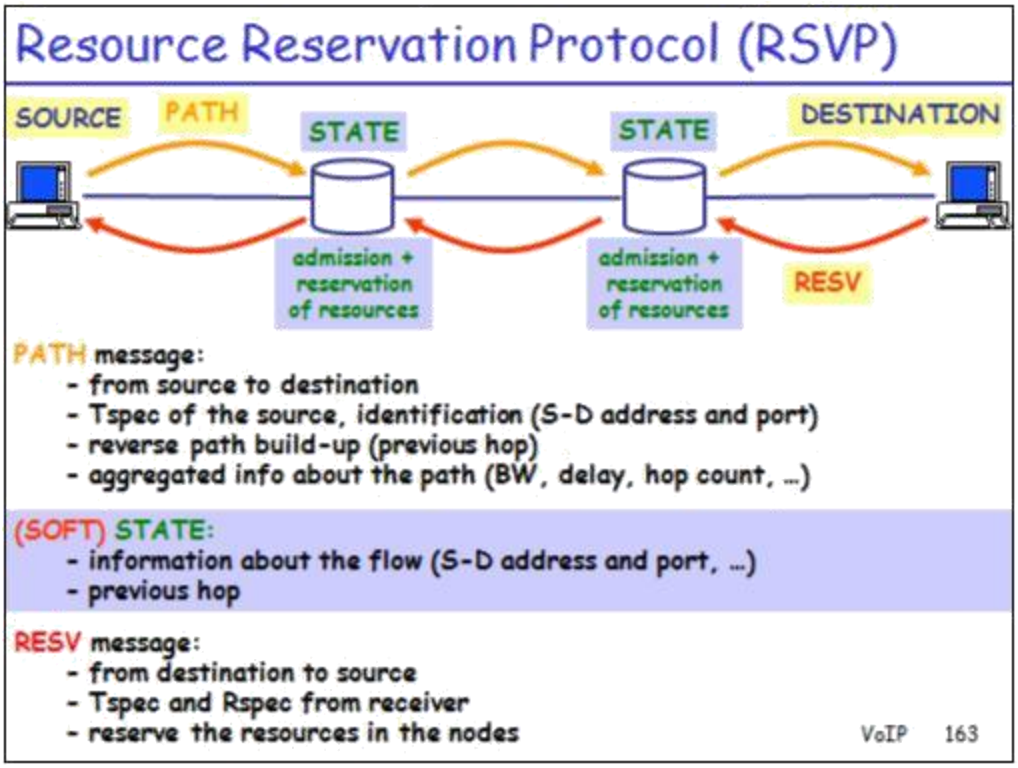
The reservation specification (Rspec) is used (in addition to the Tspec) to inform the network about the required network resources related to delay and bandwidth. R (>=r) will specify the desired bandwidth (could be omitted because r and p also give information on bandwidth). If R > r, the delay will be lower than required for the service (the desired delay depends on b/r, the actual delay depends on b/R which is smaller than b/r). If R>r there is some freedom in the delay which is indicated by the slack time S (in ms). S is the difference between the desired delay and the (lower) delay obtainable with R (>r). This will allow some freedom in the network elements to relax the resource reservation.

For more detailed information see RFC2212 (“Specification of Guarant eed Quality of Service”).



The guaranteed service will provide a performance close to a real circuit (compare to telephone circuit). There will be an assured level of bandwidth, a firm end-to-end delay bound and in general there will no queuing losses. This is of course only true for traffic that conforms to the initial specifications.

Controlled-load service is offering a best effort service but on a lightly loaded network (="not heavily loaded or congested"). There are no firm quant itative guarantees but the performance perceived by the flow will resemble a lightly loaded network, even if there is a large amount of best-effort traffic.



A signalling protocol is required to set up a flow through the network with a certain service level: RSVP or Resource ReSerVation Protocol. This protocol is using two phases:

• Path messages are sent from sender to receiver. They carry information about the source (identification, Tspec, previous hop IP address, aggregate information about the path characteristics).

• State will be maintained in the routers where the PATH message is transiting. This information includes the flow identification, the previous hop in the path, etc.

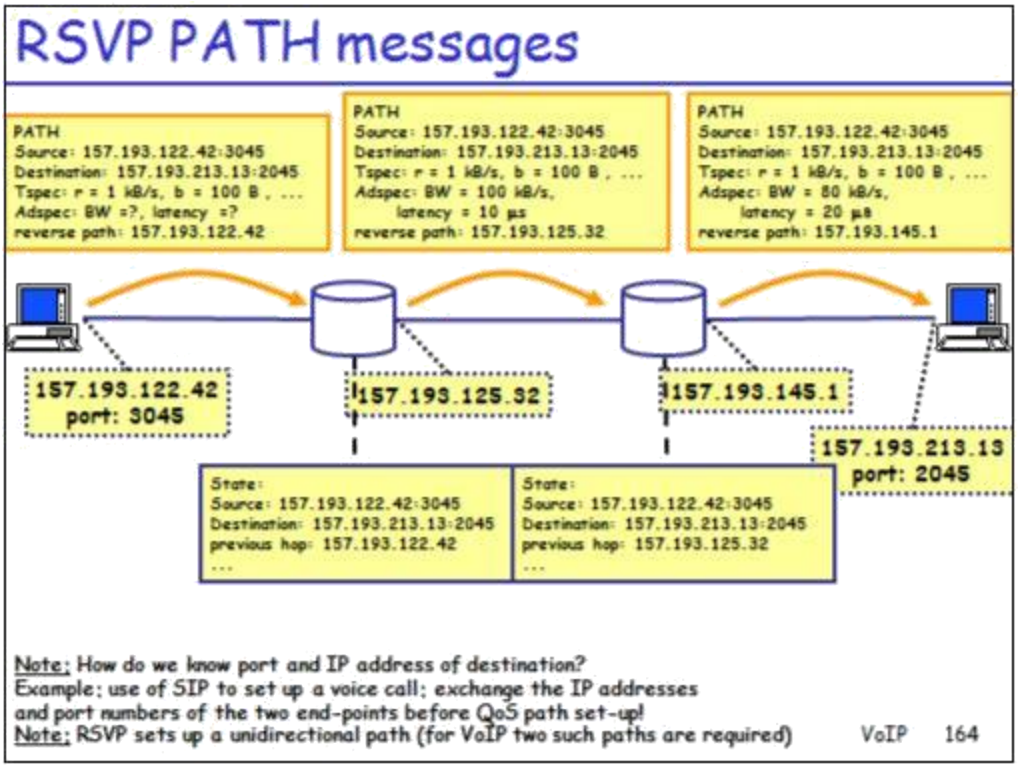
• RESV (reservation) messages are sent from destination to source with Tspec and Rspec. They will be responsible for the reservation of the resources (if accepted by the router).

• The QoS traffic will be sent along this path.

Note: RSVP is a **soft state** protocol: PATH message are sent out at regular times. When the source stops sending PATH messages, the state in the routers will disappear and the path will disappear.

In a hard state protocol, specific messages have to be sent to tear down a path (typical example is a telephone connection).

Note: the RSVP messages follow the (shortest path based) routing table entries (as established via e.g. OSPF). When the topology changes, the routing tables will change and the PATH updates will follow the new route. This results in an automatic rerouting of the path.



This figure gives an example of the PATH messages used in RSVP.

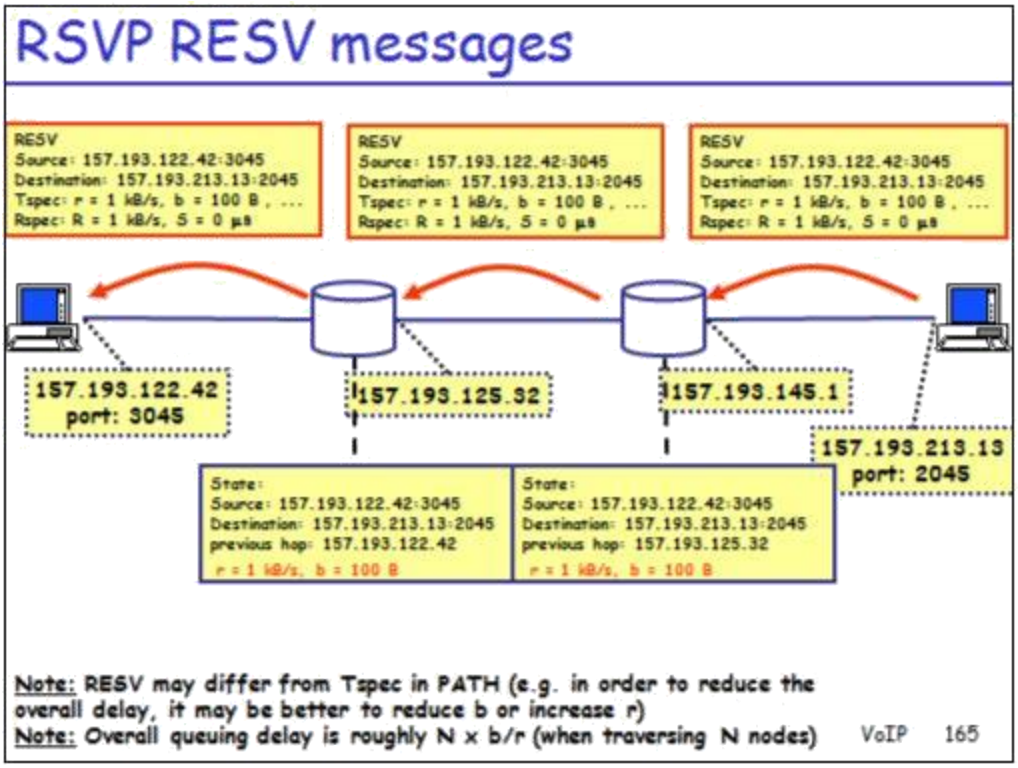
The PATH messages will leave from the source and contain the identification of the flow (using S-D IP addresses and ports). Also the Tspec of the source is included. The example uses a token rate of 1 kbyte/s (=this could correspond to a voice codec of 8 kbit/s) and the bucket depth is 100 Bytes (which will result in a maximum burst size of 100 Bytes).

We also observe the Adspec: this is aggregate information (so-called *One Pass With Advertising (OPWA)* information) about the path from source to destination. When starting, no information is available. In the first router the available bandwidth (e.g. 100 kByte/s) and the latency (e.g. 10 µs) are filled in. Note that the latency is the fixed latency (NOT including possible delay in the buffers), meaning that it will be a lower bound. When traversing a next router, the bandwidth may be reduced ("minimal measure") and the latency will be increased ("additive measure").

A last important part is the reverse path indication (*RSVP_HOP* object). This will allow to set up the reverse path for the RESV messages (from destination to source) following the same route as the shortest route from source to destination. Note that this is necessary because the shortest route from source to destination is not necessarily the same as the shortest route from destination to source (and we want that the RESV messages go back over the same route as the PATH messages!).

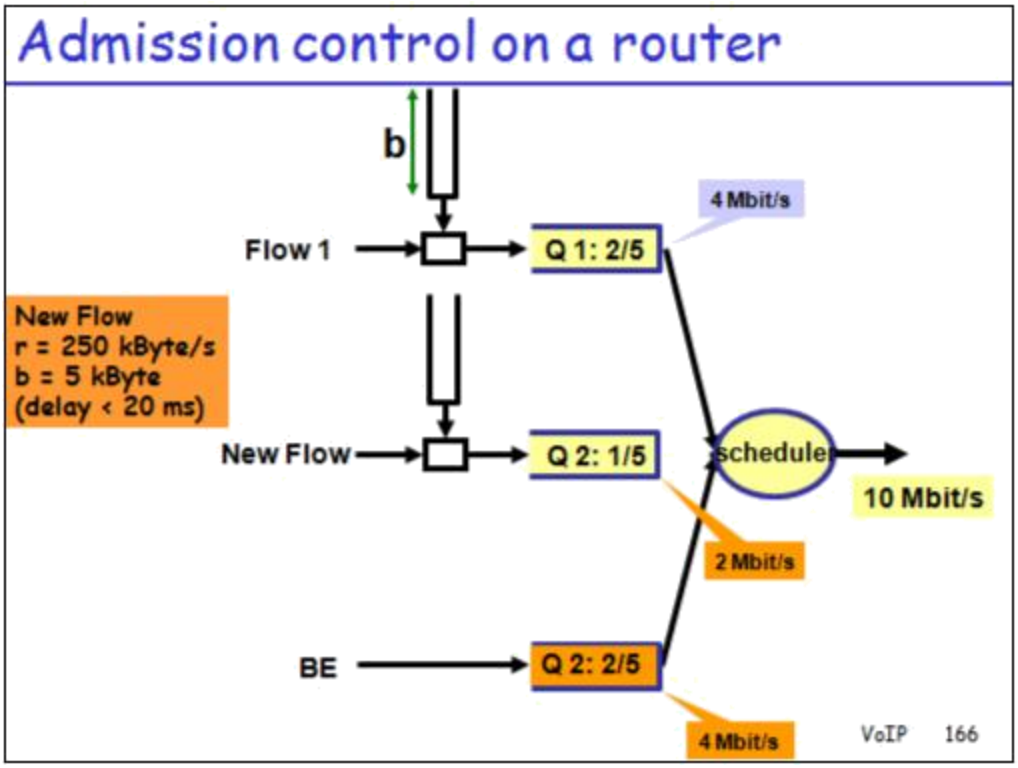
Note: When a PATH message is sent out, we observe that the port of the destination is already known. This is because (for example) in a previous phase SIP was used to set up the call between to IP phones (see earlier). SIP will indeed exchange the IP address and port number of the two end-points.

Note: RSVP is unidirectional. This means that for a VoIP service using QoS reservation, the two sides have to set up (separately) a path supporting the required QoS.



When the receiver receives the PATH message, it will compose a RESV message (Reservation). This message will effectively (try to) reserve the resources in the different routers along its path. The routers will use admission control in order to see if enough resources are available to support the requested flow. The RESV message is using both the Tspec and the Rspec. The state in the routers will change because they have to set the r and b values and the weight factor w (use of w in WFQ and r and b in token bucket in the nodes).

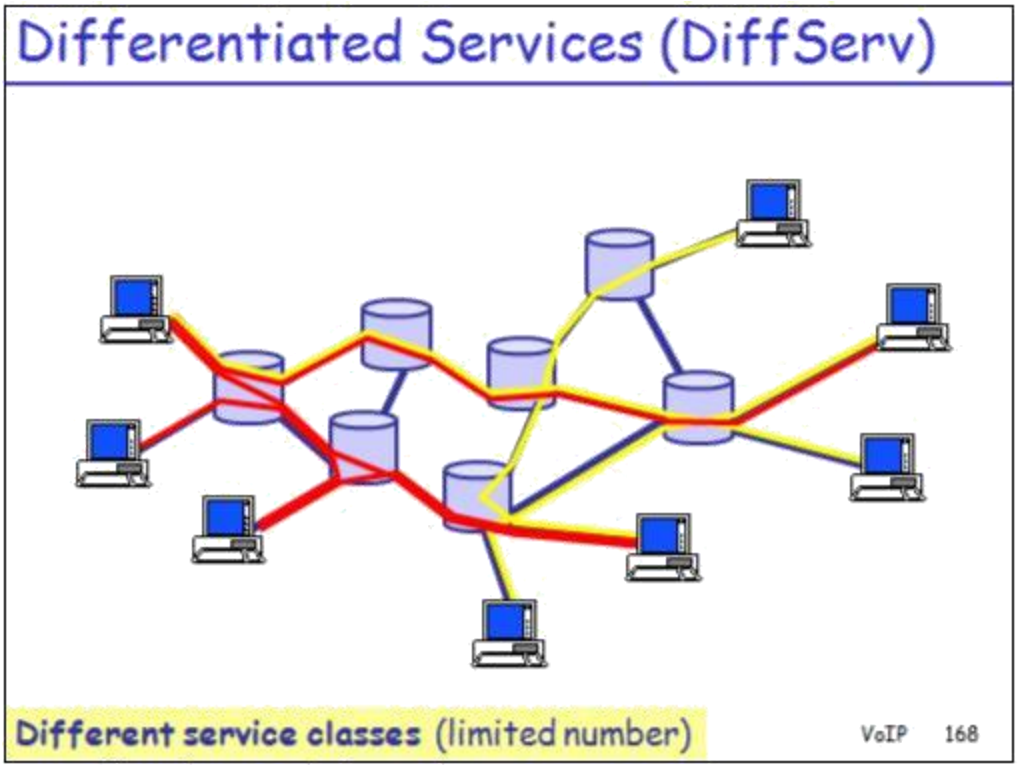
Note: It is possible that the RESV message is not using exactly the same Tspec as the sender. Suppose that the latency (=delay) obtained from the Adspec is relatively high, then it may be useful to reduce the possible delay in the queues. This can be done by reducing the bucket depth b or increasing the token rate r, because this will reduce the delay (b/r). It is also possible that there are a large number of hops from source to destination, resulting in high buffer delays (when traversing N nodes, the upper limit for the queuing delay is roughly N x b/r). By reducing again b or increasing r, a lower overall delay is obtainable.



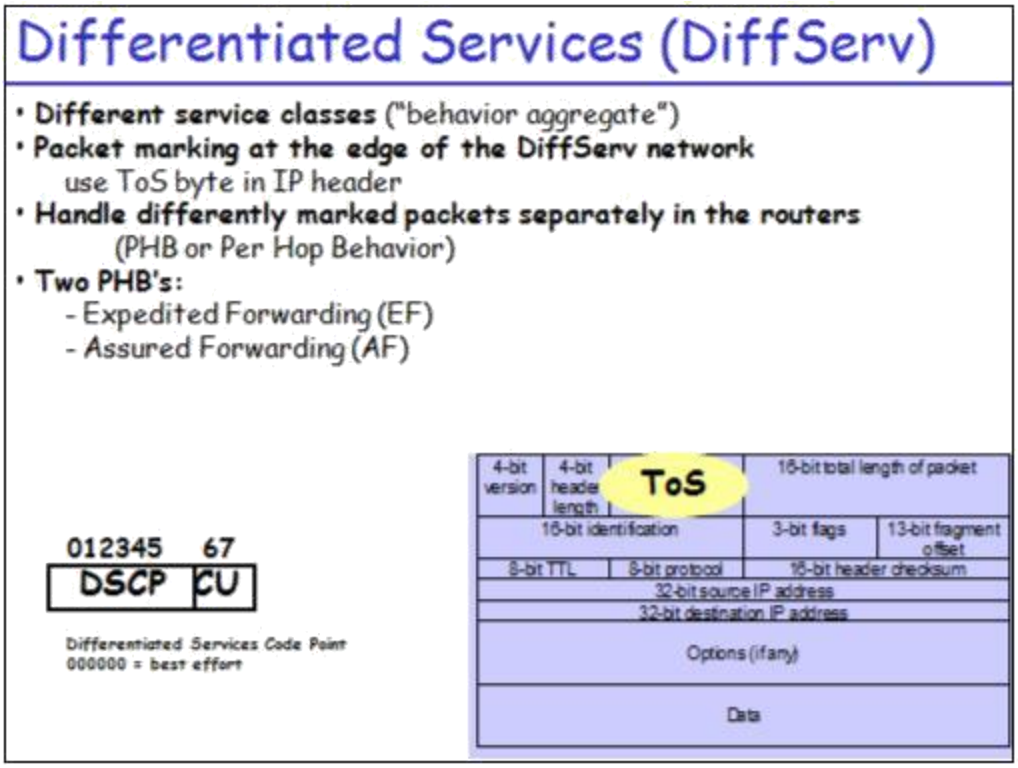
A simple example of admission control is shown here. Initially there is a guaranteed flow of 4 Mbit/s (Flow 1) and the rest is best effort (BE) and this corresponds to the remaining 6 Mbit/s (on a 10 Mbit/s outgoing line). A new flow request arrives for a flow of 2 Mbit/s (r = 250 kByte/s, b = 5 kByte) and this is acceptable by reducing the bandwidth available for the BE traffic (from 6 Mbit/s to 4 Mbit/s).

Content

1. Introduction: Multimedia Services
2. The current telephone network: PSTN
3. Voice over IP: VoIP
   - Architecture
   - Control plane
   - User plane (terminal and network)
     - Node and Overall network issues
       - Intserv
       - Diffserv
4. Multimedia over IP



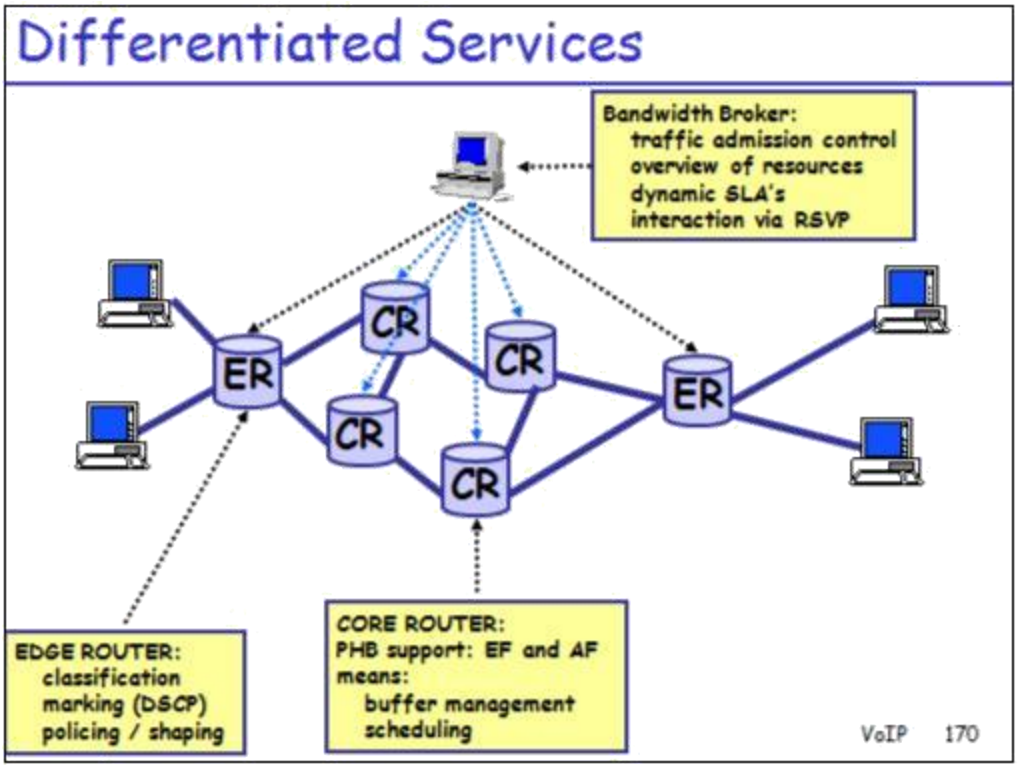
This figure illustrates the DiffServ approach: different classes are treated separately in the network (only state about the different classes has to be maintained in the routers). There is no need to separate individual flows, only classes (which may contain many flows) are separated. The figure illustrates two classes.



Differentiated Services (DiffServ) will make use of different service classes to provide a certain level of QoS to different flows. The term "behavior class" is used (see RFC 2475, "An Architecture for Differentiated Services"). Inside the network, the nodes (DiffServ routers) will support certain Per Hop Behaviors (PHB) to provide the required QoS for the different behavior classes. This PHB will influence the sharing in a router of buffers and link bandwidth between the different classes of traffic.

In practice, two service classes (behavior classes) are defined: expedited forwarding (EF) and assured forwarding (AF).

CU: currently unused

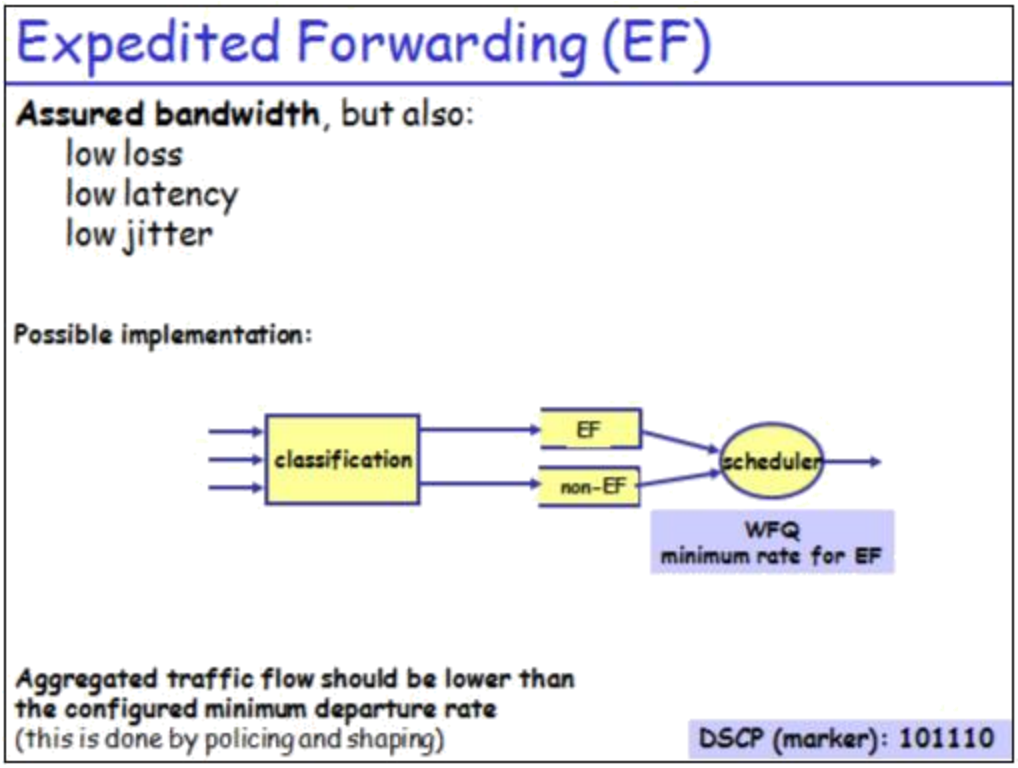


The functions required in a DiffServ enabled network are indicated on the figure.

At an edge router, the flow has to be classified based on e.g. source and destination IP addresses and port numbers (but also other means are possible, e.g. RTP versus non-RTP packets). Based on this classification one will mark the packets using the DSCP (DiffServ Code Point). Policing and shaping may be required because flows have to be conforming a certain traffic profile (otherwise it would not be possible to provide any QoS in the network because a certain uncontrolled flow could overwhelm the network using all resources of lower quality classes).

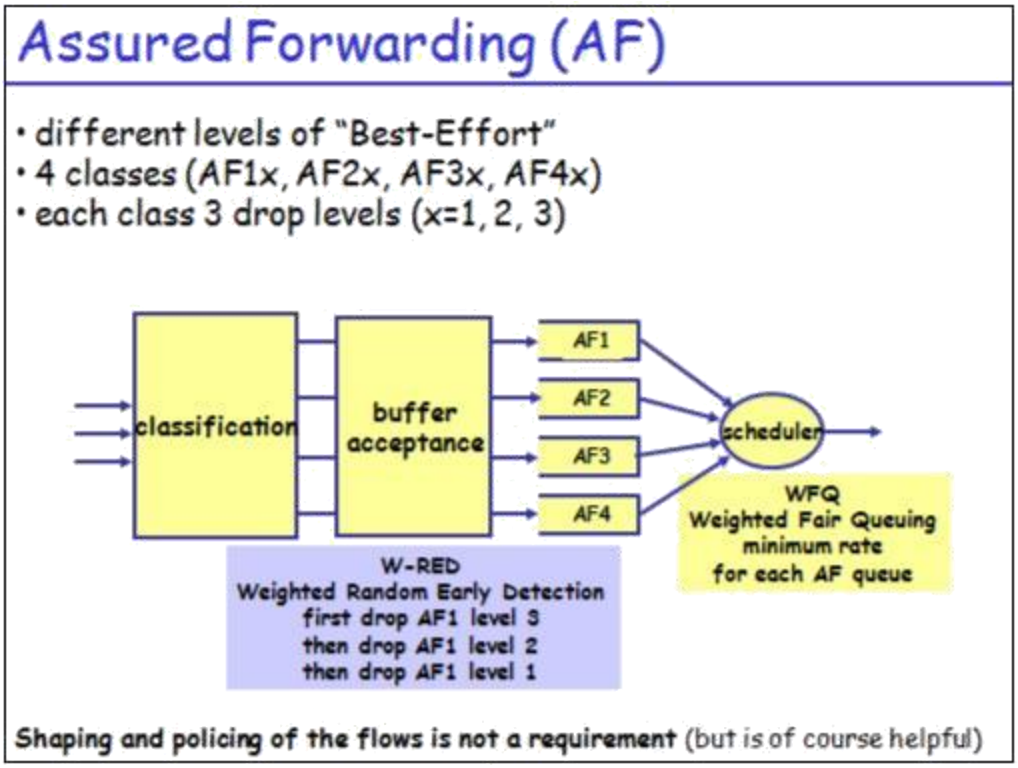
In a core router one has to support certain Per Hop Behaviors (PHB). These PHB's will define differences in the performance behavior amongst different classes. Note that a PHB is only a description of a behavior at a node (node is considered as a "black box") and does not specify how this behavior should be achieved. The available means to support a certain PHB are buffering, buffer management and scheduling. There are two important PHB's: Expedited Forwarding (EF) and Assured Forwarding (AS).

In general, a traffic flow has to be allowed to a certain traffic class ("behavior aggregate"). This could be done in a static way, meaning that the network operator is manually configuring the edge router in order to allow a certain flow to a certain traffic class (static SLA or Service Level Agreement). The operator could keep track of all the current traffic already allowed on the network, and based on this information a decision is made whether to accept the additional flow. This could also be done in a more dynamic way using a dynamic SLA and a centralised computer (Bandwidth Broker or BB) which will interact with the customers (to negotiate and agree on a SLA), with the edge routers for configuration and (eventually) with the core routers to obtain information about the current traffic and to set certain parameters (e.g. weight factors in WFQ). Note that the latter (traffic measurement) is not necessary because the Bandwidth Broker may keep track of the current flows already allowed to a certain traffic class. The interaction of BB with customer and edge router could be based on the use of RSVP (which was originally developed in conjunction with IntServ).



The expedited forwarding PHB specifies that the departure rate of a class of traffic from a router must equal or exceed a configured rate. This means that during any interval of time, the class of traffic can be guaranteed to receive enough bandwidth so that the output rate of the traffic equals or exceeds this minimum configured rate. Note that the EF per- hop behavior implies some form of isolation among traffic classes, as this guarantee is made independently of the traffic intensity of any other classes that are arriving to a router. Thus, even if the other classes of traffic are overwhelming router and link resources, enough of those resources must still be made available to the class to ensure that it receives its minimum rate guarantee. EF thus provides a class with the simple abstraction of a link with a minimum guaranteed link bandwidth.

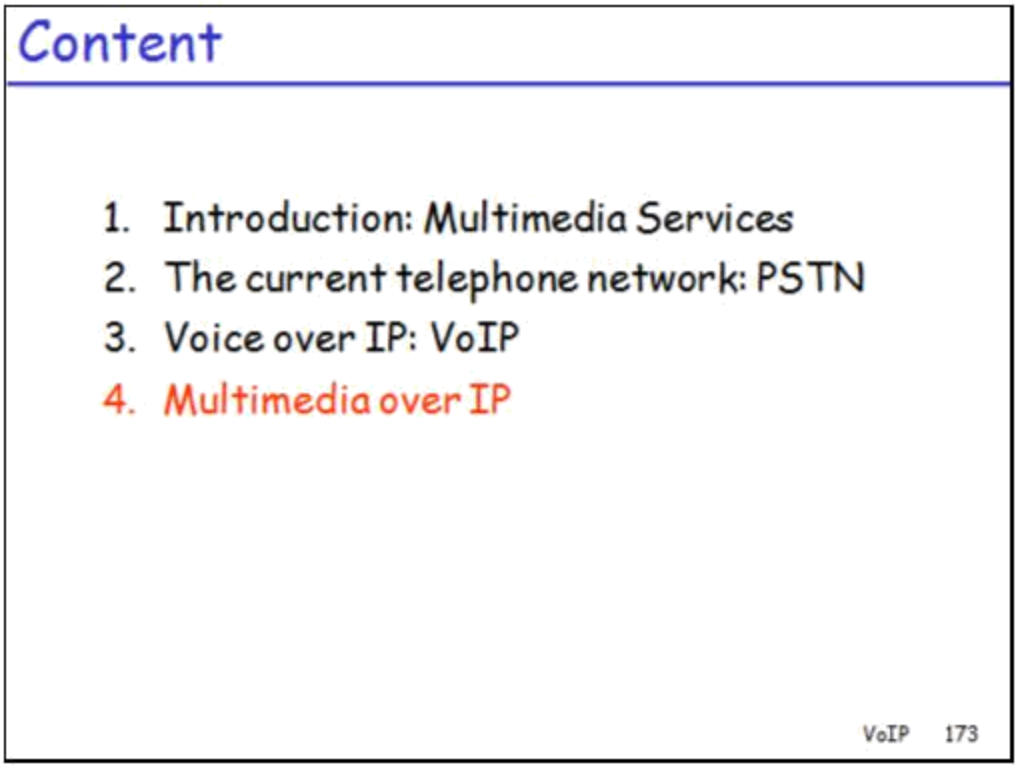
Note that it will only be possible to guarantee the PHB when the EF traffic is conforming its traffic profile. This should be checked at the edge of the network using policing and shaping.



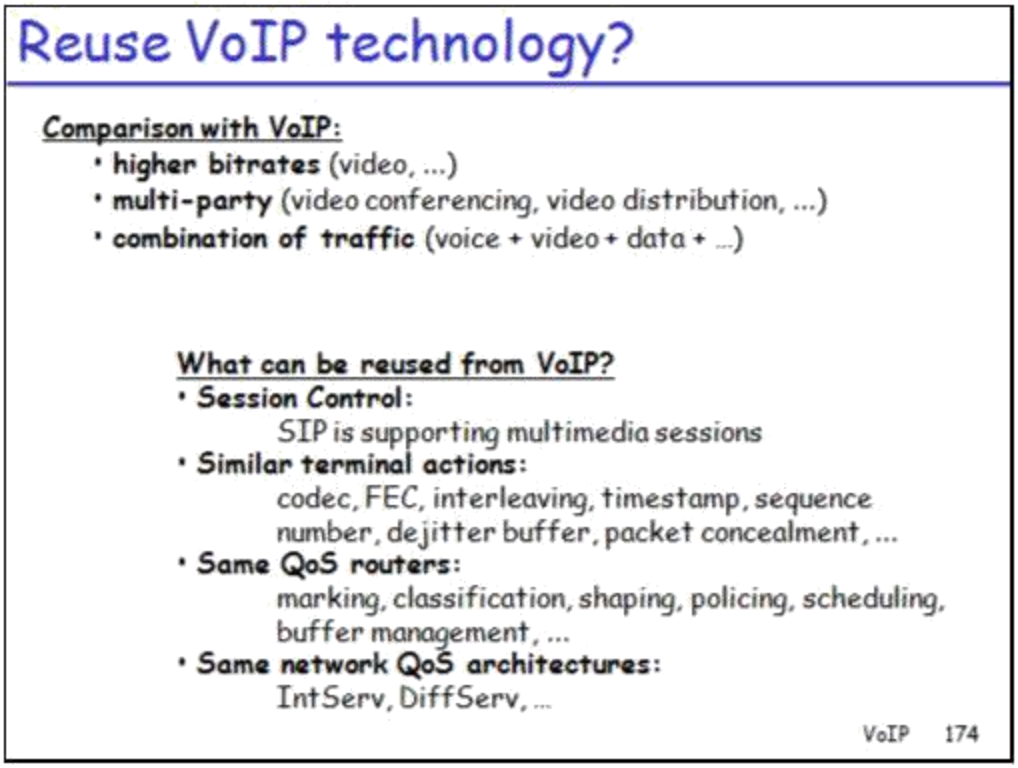
The assured forwarding (AF) Per Hop Behavior (PHB) is more complex. AF divides traffic into four classes, where each AF class is guaranteed to be provided with some minimum amount of bandwidth and buffering. Within each class, packets are further partitioned into one of three "drop preference" categories. When congestion occurs within an AF class, a router can then discard (drop) packets based on their drop preference values. By varying the amount of resources allocated to each class, an ISP can provide different levels of performance to the different AF traffic classes.

The AF PHB could be used as a building block to provide different levels of service to the end systems, for example, O lympic- like gold, silver, and bronze classes of service. But what would be required to do so? If gold service is indeed going to be "better" (and presumably more expensive!) than silver service, then the network provider must ensure that gold packets receive lower delay and/or loss than silver packets. Recall, however, that a minimum amount of bandwidth and buffering are to be allocated to each class. What would happen if gold service was allocated x% of a link's bandwidth and silver service was allocated x/2 % of the link's bandwidth, but the traffic intensity of gold packets was 100 times higher than that of silver packets? In this case, it is likely that silver packets would receive better performance than the gold packets! (This is an outcome that leaves the silver service buyers happy, but the high-spending gold service buyers extremely unhappy!) C learly, when creating a service out of a PHB, more than just the PHB itself will come into play. In this example, the dimensioning of resources (determining how much resources will be allocated to each class of service) must be done hand-in- hand with knowledge about the traffic demands of the various classes of traffic (e.g. use of a bandwidth broker).

In contrast to the EF PHB, traffic conditioning at the edge of the network is a recommendation but not a must. (For details, see RFC 2597 “Assured Forwarding PHB Group”.)



This last paragraph will discuss some issues related to multimedia networking. Beside some general issues, some details will be given about IPv6 and multicast.

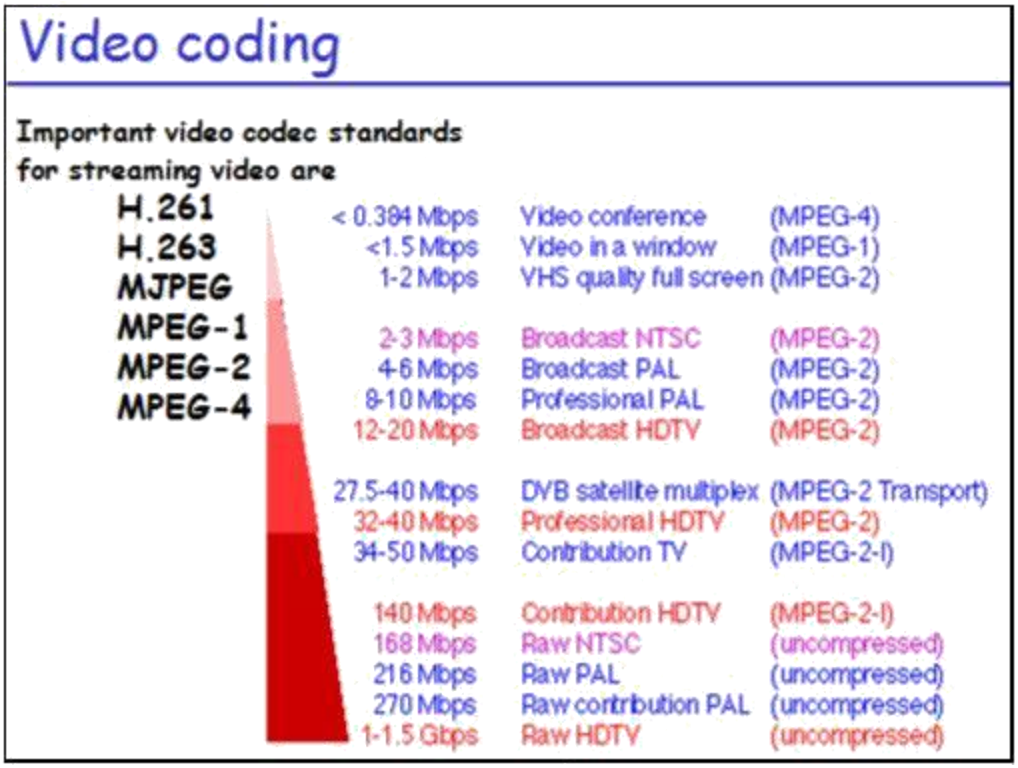


The last part of this chapter will discuss some issues related to multimedia networking over IP. The major difference compared to the previous part, where the focus was on VoIP, are:

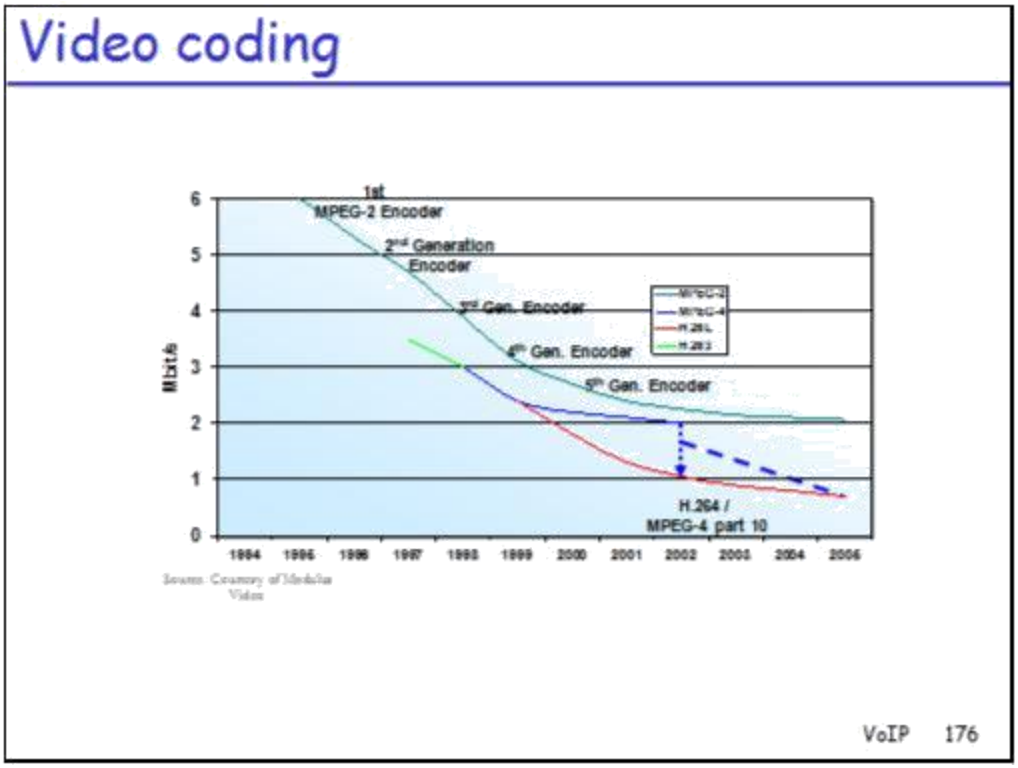
- Much higher bitrates are encountered (e.g. MPEG-2 video at 2 Mbit/s)
- Multi-party is much more common (e.g. video distribution), although radio broadcast of voice conferences is an example of VoIP with multiple parties (not considered previously).
- Multimedia will combine different traffic types such as voice, video, data, … This was not the case in VoIP. Problems will e.g. be encountered related to the synchronization of the different streams (e.g., voice and video signals of a movie).

It is interesting to note that most of the concepts introduced in the VoIP part are reusable for multimedia:

- The same standards are used for session control. Although in the VoIP case only point-to-point connections were considered, SIP and H.323 support also multi-party sessions.
- Similar terminal actions are required in order to reduce the required bandwidth, improve the sensitivity to packet loss or reduce the influence of delay and jitter.
- In the network, the same building blocks are used in the routers to support QoS. These building blocks will separate different traffic flows and give them specific treatments.
- The configuration of the QoS building blocks in the network will use the same architectures: IntServ (only for small networks) and Diffserv (currently the preferred option due to its scalability). MPLS, constraint based routing and load balancing will play an equally important role in multimedia support.



A number of video codecs are listed above. From the bitrates, it is clear that video will result in a much higher load on the network. It is also clear that uncompressed video (bitrates > 100 Mbit/s) could not be supported in the Internet.



This figure illustrates the evolution in video codecs.