FACHHOCHSCHULE TRIER
Hochschule für Technik, Wirtschaft und Gestaltung
University of Applied Sciences
Informatik

# VoIP – Internet Telephony

Rainer Oechsle
University of Applied Sciences, Trier
(FH Trier)

January 2006

## Outline (1)

- PSTN: Classical Circuit-Switched Telephony
  - POTS (Analog Telephony)
  - ISDN (Digital Telephony)
- VoIP: Packet-Switched Telephony – Basic Techniques
  - Audio Codecs
  - Data Transport (RTP, RTCP)
  - Addressing
  - Signaling (SIP, H.323, IAX2)

## Outline (2)

- VoIP: Advanced Techniques
  - QoS
  - Security (ICE)
  - Addressing (DNS SRV/NAPTR – ENUM)
- Gateways between ISDN and VoIP Networks
- VoIP Hardware / Software
- Business Aspects
- Summary and Outlook

## PSTN: Classical Circuit-Switched Telephony



PSTN: Public Switched Telephone Network

- POTS (Analog Telephony)
- ISDN (Digital Telephony)

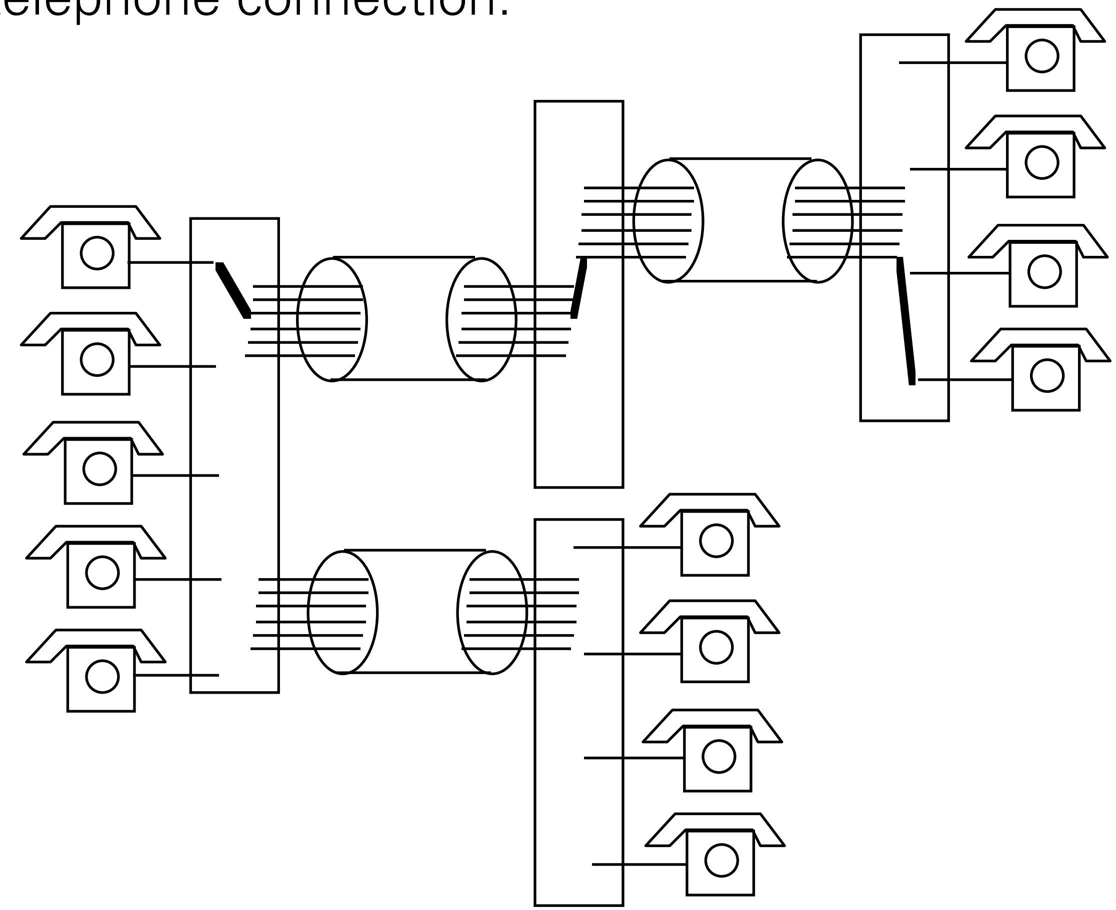
## POTS (Analog Telephony) (1)



- POTS: Plain Old Telephone Service
- inventors:
  - Philipp Reis, Frankfurt, 1861
  - Alexander Graham Bell, Boston, 1876, patent
  - Elisha Gray, 1876, 2 hours later
  - Antonio Santi Giuseppe Meucci, 1854, presented in 1860, filed patent in 1871, patent expired in 1874
- main usage (assumed): concert broadcasting
- voice coding and transmission: analog acoustic signal transformed to (similar) analog electrical signal

# POTS (Analog Telephony) (2)

- analog telephone connection:



## POTS (Analog Telephony) (3)

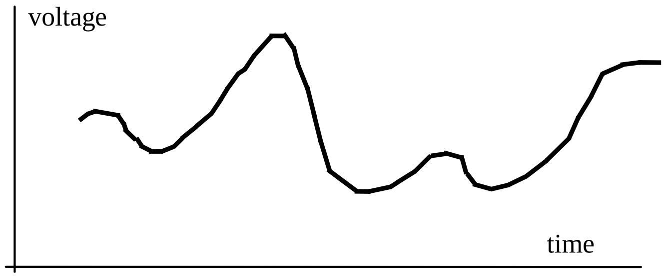
- analog telephone connection (cont.):
  - dedicated link between partners
  - reserved bandwidth that cannot be used by other users
    => billing based on connection time
- signaling (call setup and teardown):
  - in the beginning: point-to-point links
  - later: telephone wired to central office:
    - manual signaling
    - automatic signaling (invented by Strowger, Indiana, 1891)

## PSTN: Classical Circuit-Switched Telephony

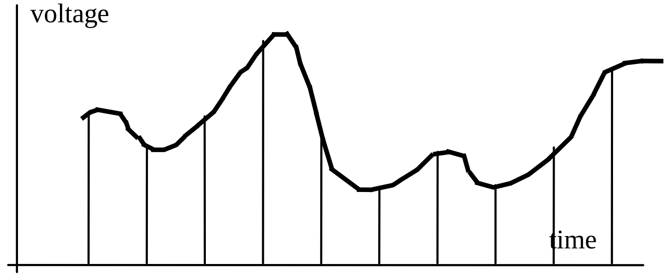
- POTS (Analog Telephony)
- ISDN (Digital Telephony)

# ISDN (Digital Telephony) (1)

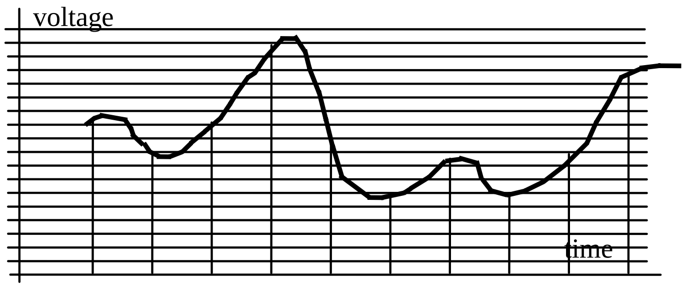
- transforming analog acoustic / electric signal to digital values: PCM (Pulse Code Modulation)




sampling:



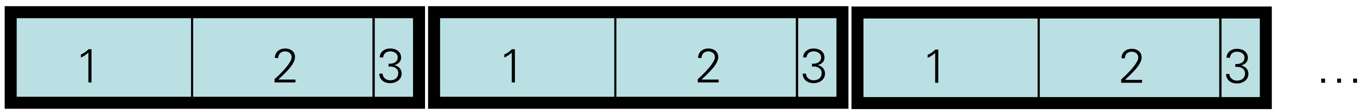

quantisation:




8000 samples/s, each sample coded by 8 bits => 64 kbit/s

## ISDN (Digital Telephony) (2)

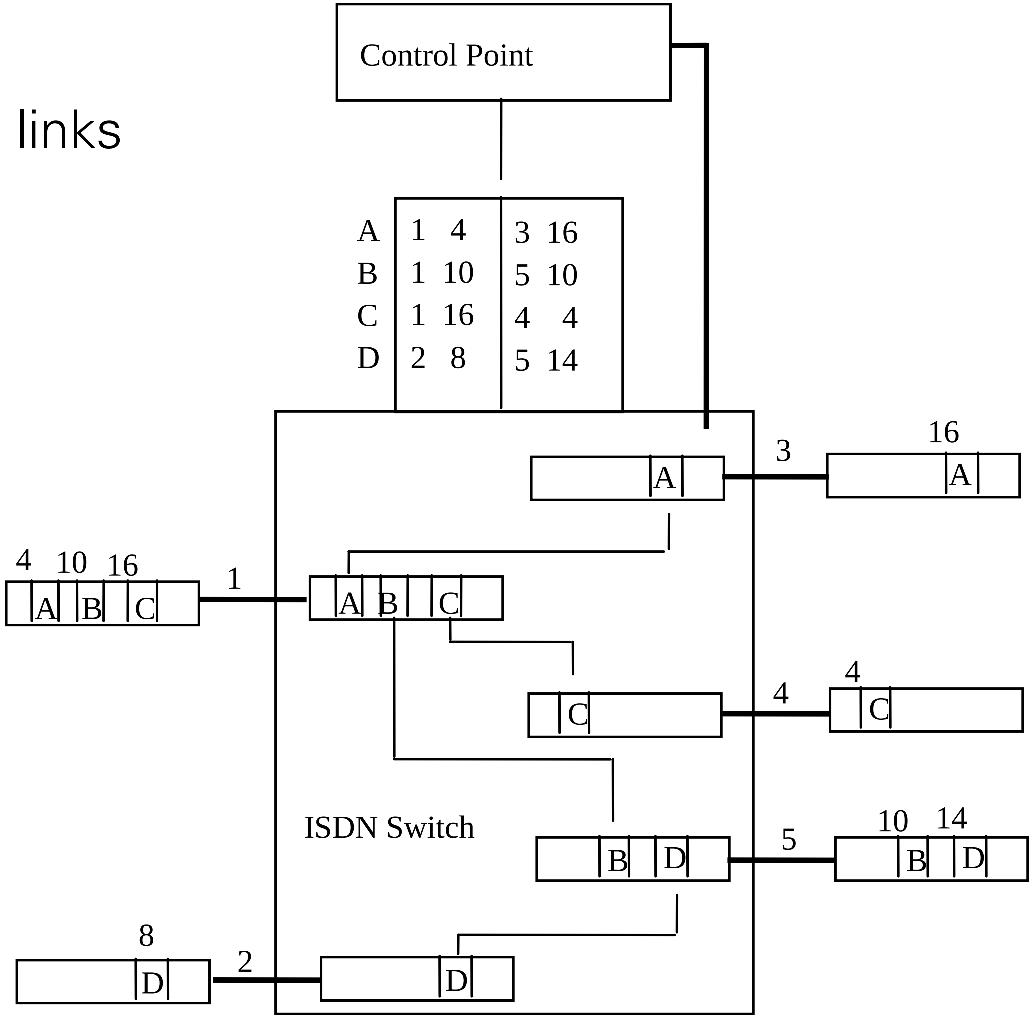
- multiplexing (transmitting serveral logical channels on one physical channel):
  - 8000 frames / s (every 125 µs)
  - example: 2 channels (64 kbit/s for each one) and 1 channel (16 kbit/s)




    - channels 1 and 2: 8 bits / frame
    - channel 3: 2 bits / frame
  - example: 32 channels (64 kbit/s for each one) => 2 Mbit/s link

# ISDN (Digital Telephony) (3)

- circuit switching:
  switching for TDM links



## ISDN (Digital Telephony) (4)

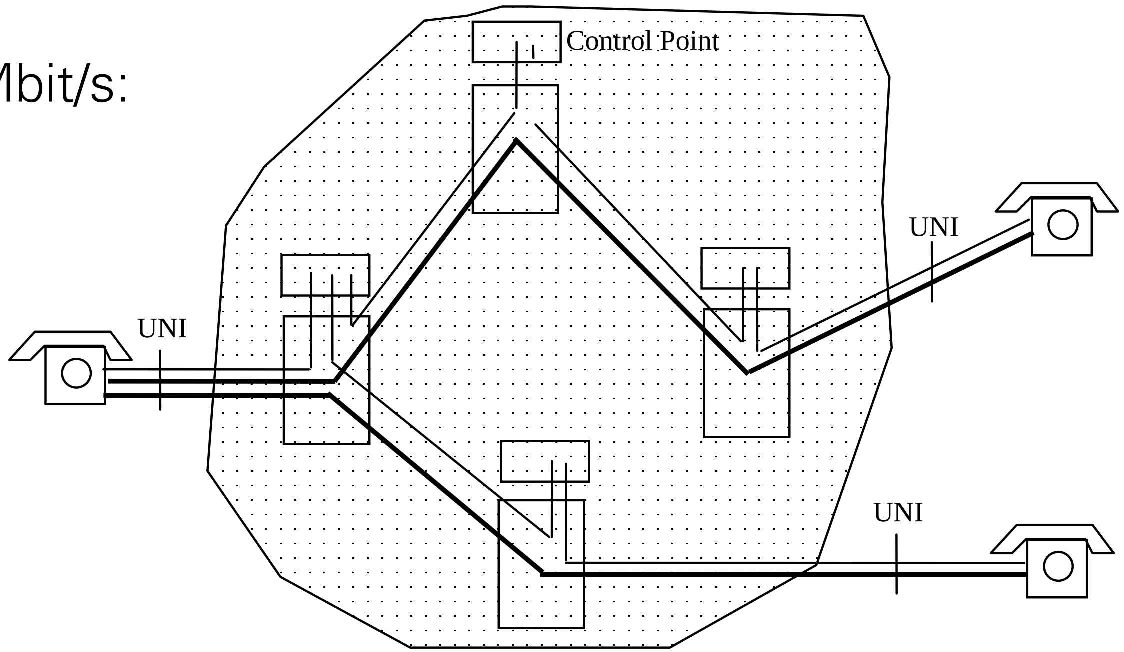


- digital telephone connection:
  - similar to analog telephone connection
  - on each link a channel is reserved
    => reserved resource (bandwidth) between partners
    => low delay, no jitter

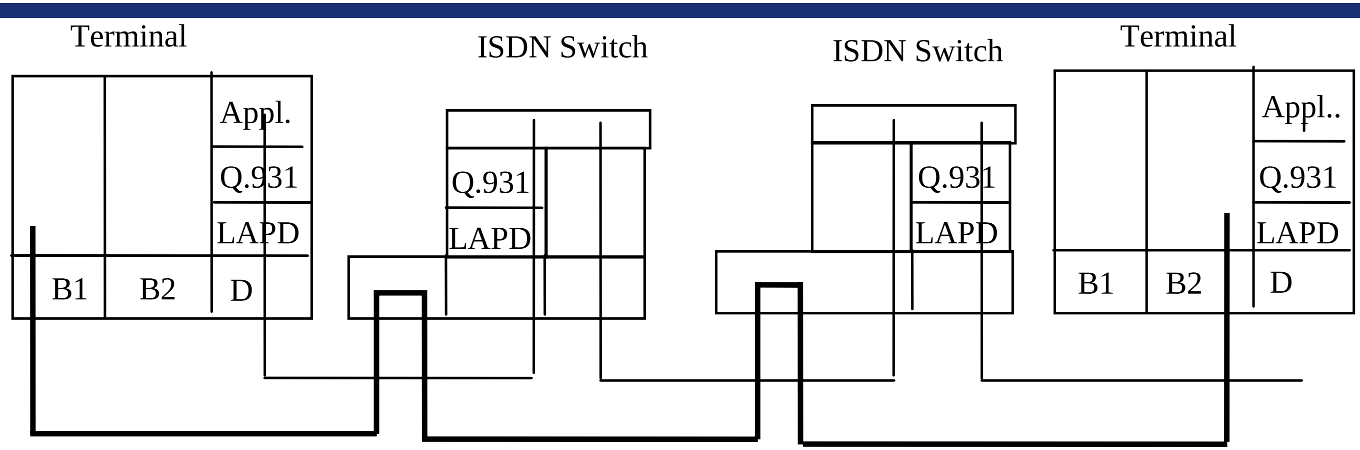
## ISDN (Digital Telephony) (5)



- basic rate interface: 144 kbit/s:
  2 B channels (64 kbit/s)
  + 1 D channel (16 kbit/s,
  for signaling)
- primary rate interface: 2 Mbit/s:
  30 B channels (64 kbit/s)
  + 1 D channel (64 kbit/s,
  for signaling)



# ISDN (Digital Telephony) (6)



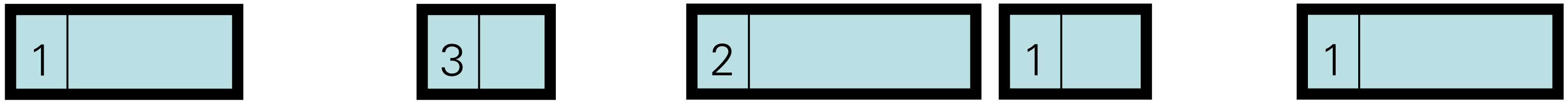

- D channel: permanent connection to control point of nearest ISDN switch
  - layer 1: TDM
  - layer 2 of D channel: LAPD
    - frame-based communication
    - protocol between terminal and ISDN switch
    - reliability by acknowledgements and retransmission
    - layer 3: Q.931
- permanent connection between control points of ISDN switches

# ISDN (Digital Telephony) (7)

- Q.931:
  layer 3 protocol
  on D channel

## ISDN (Digital Telephony) (8)

- contrast to circuit switching: packet switching (Internet):




  - packets have different lengths
  - different gaps between packets
  - header for marking logical channel or destination (IP)
  - switching in routers may lead to queueing
  - no qos quarantee

# ISDN (Digital Telephony) (9)

- summary:

| | ISDN | VoIP |
|---|---|---|
| coding / decoding | PCM | PCM and other codecs? |
| multiplexing / switching | TDM and circuit switching (qos guaranteed) | packet switching (qos guarantee?) |
| data transport | B channel | transport protocol? |
| addressing | telephone numbers (E.164) | VoIP addressing? |
| signaling | LAPD and Q.931 signaling protocol | signaling protocol? |

other topics:

- firewalls / NATs
- gateways ISDN / VoIP

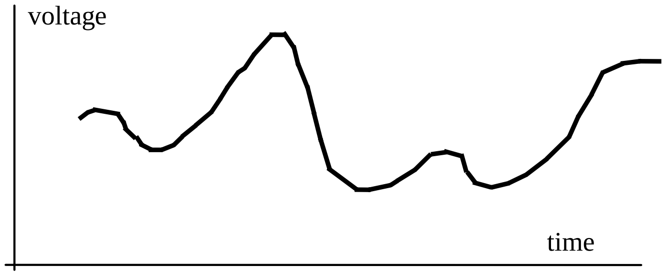
- Audio Codecs
- Data Transport (RTP, RTCP)
- Addressing
- Signaling (SIP, H.323, IAX2)

# Audio Signal

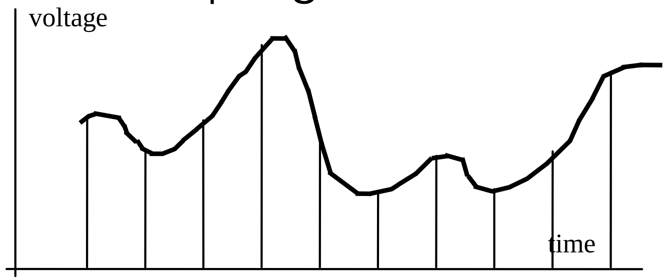
- audio signal: change of air pressure over time (analog signal)
  => transformed into a similar electrical signal: change of voltage over time
- coding: transformation of analog signals into a sequence of digital values
- decoding: inverse transformation
- codec: co̲d̲ing / d̲e̲c̲oding (software / hardware) component
- two kinds of codecs in VoIP:
  - sample-based
  - frame-based

## Sample-based Coding

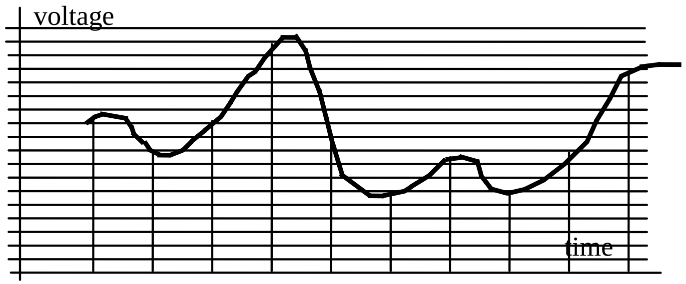
- basic procedure: PCM (Pulse Code Modulation), used in ISDN:
  - sampling: discrete time values
  - quantisation: discrete sampling values




sampling:




quantisation:

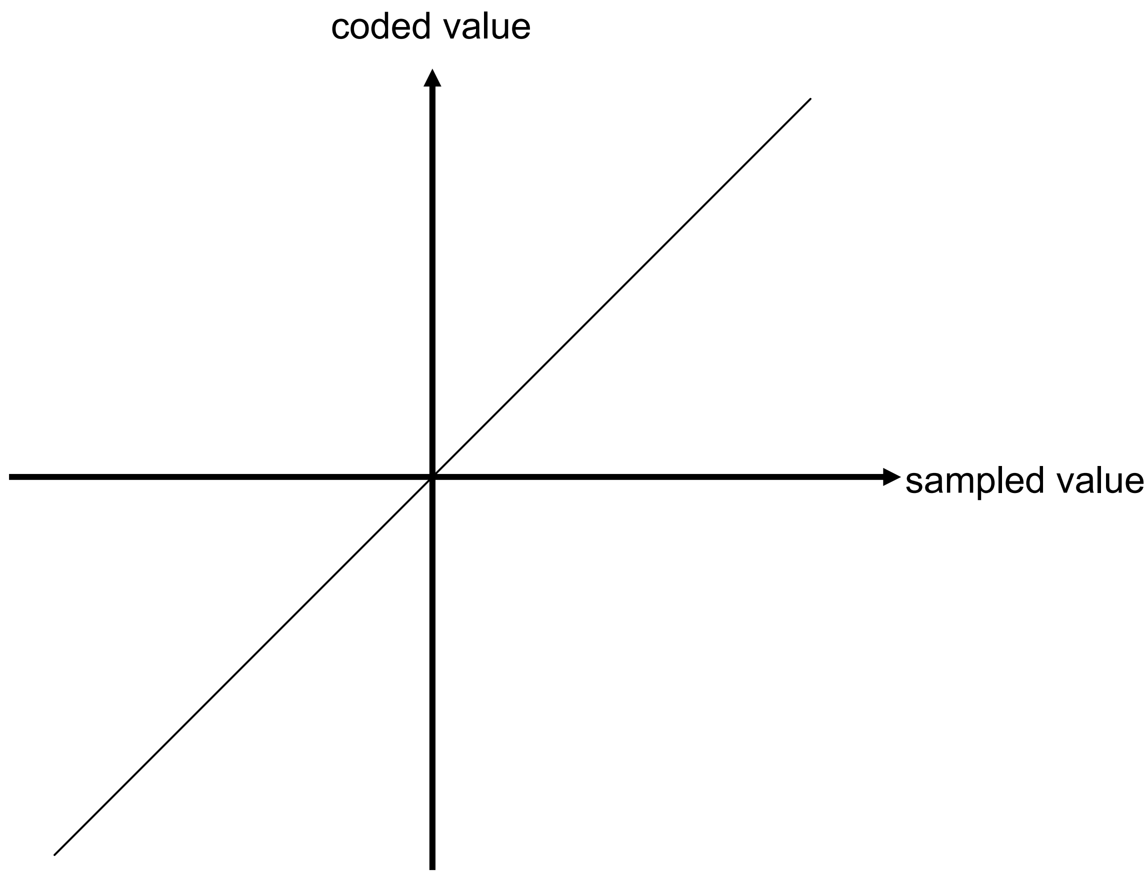


## Sampling

- Fourier transformation: separation of frequencies
- Shannon's theorem: $F_{sampling} \geq 2 * F_{max}$
- low pass filtering (cutting off higher frequencies)
  => reducing $F_{max}$ => reducing $F_{sampling}$
- examples:
  - ISDN: 300 – 3400 Hz => sampling rate: 8000 Hz
    8 bits / sample value => 64 kbit/s
  - CD: 22000 Hz => sampling rate: 44000 Hz
    16 bits / sample value => 700 kbit/s
    2 channels for stereo: 1.4 Mbit/s

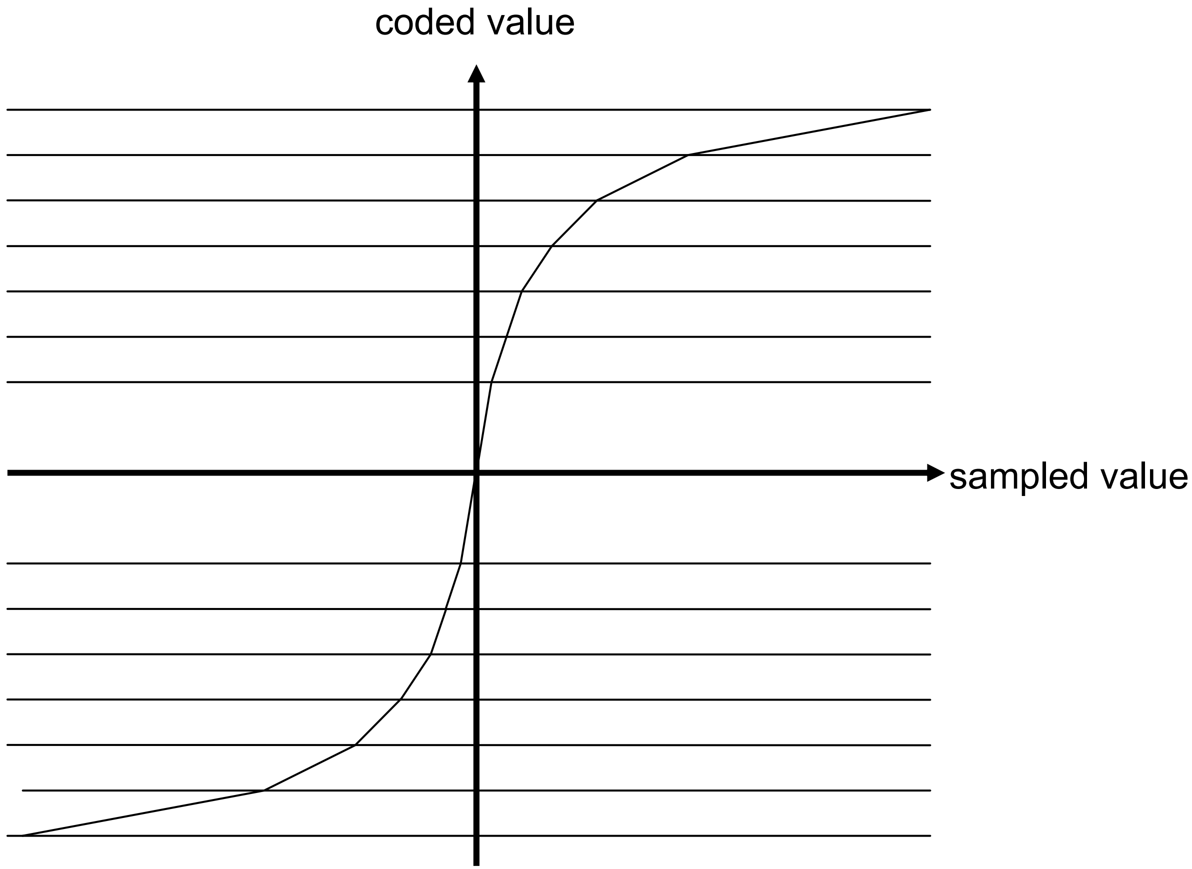
## Quantisation (1)

- linear
  - drawback: quantisation errors for smaller values lead to bigger problems



## Quantisation (2)

- non-linear:
  - higher resolution for smaller values, lower resolution for larger values
  - 13 segments => A law (Europe), 15 segments => µ law (USA, Japan)



## Sample-based Coding

- PCM (Pulse Code Modulation): G.711, used in ISDN
  8.000 * 8 bit/s = 64 kbit/s
- DPCM (Differential PCM):
  - prediction of next value based on n previous values
  - linear prediction (estimated value):
    $s_e(i) = a_1 * s(i-1) + a_2 * s(i-2) + \ldots + a_n * s(i-n)$
  - transmitting: $d(i) = s(i) - s_e(i)$
  - for "good" predictions: d(i) small values
    => can be coded with less bits
- ADPCM (Adaptive DPCM): G.726, G.727
  - adaptation of $a_1, a_2, \ldots, a_n$
  - code words of 2, 3, 4 or 5 bits
    => 16, 24, 32, 40 kbit/s

## Frame-based Coding

- human voice: constant sound within 10 – 30 ms time interval
- characterizing sound signal by parameter values (instead of sample values):
  - voiced / unvoiced
  - pitch period
  - gain
  - linear predictive coding parameters $a_1, a_2, \ldots, a_{10}$
- 13 parameters coded with 48 bits
- 20 ms time frame => 50 time frames / s
  => 50 * 48 bit/s = 2,4 kbit/s
- 10 ms time frame => 100 time frames / s
  => 100 * 48 bit/s = 4,8 kbit/s

## Coding / Decoding Assessments

- MOS: Mean Opinion Score
- assessment of decoded signal by test persons
- scale:

| | |
|---|---|
| 5 | excellent |
| 4 | good |
| 3 | fair |
| 2 | poor |
| 1 | bad |

# Assessment of VoIP Codecs

| | | |
|---|---|---|
| PCM | 64 kbit/s | 4.3 – 4.5 |
| ADPCM | 16 / 24 / 32 / 40 kbit/s | 3.4 / 3.6 / 3.9 / 4.2 |
| Frame-based | 4.8 – 16 kbit/s | 3.5 – 4.1 |

comparison: analog telephony: 3.5 – 4.0

## VoIP: Packet-Switched Telephony – Basic Techniques

- Audio Codecs
- <u>Data Transport (RTP, RTCP)</u>
- Addressing
- Signaling (SIP, H.323, IAX2)

## VoIP Data Transport

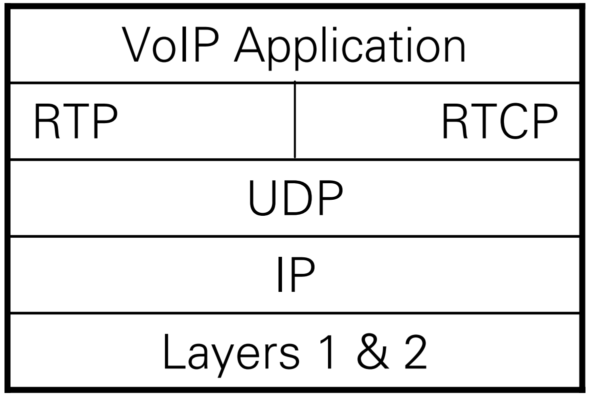
- candidate Internet transport protocols:
  - TCP:
    - connection-oriented,
      reliable (acknowledgements / resending)
    - problem: resending causes additional delay
  - UDP:
    - connectionless, unreliable
    - problem: no sequence numbers, no timestamps
- => both are not appropriate
- => RTP/RTCP (on top of UDP)

# RTP / RTCP (1)

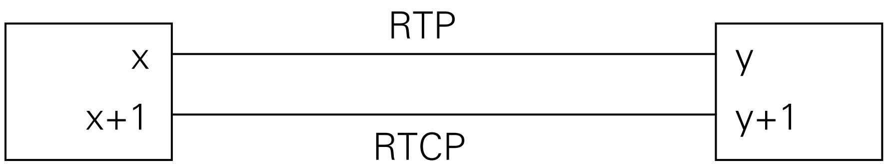
- RTP: Real-time Transport Protocol
  RTCP: RTP Control Protocol
  RFC 3550
- RTP: voice data transport
- RTCP: reports about current status of receiver and current quality of audio transport, e.g.

# RTP / RTCP (2)

- both protocols use UDP:




- port numbers:




x, y: even port numbers

# RTP (1)

- RTP functions:
  - provides end-to-end delivery services for data with real-time characteristics (audio and video)
  - includes payload type identification, sequence numbering, and timestamping
  - supports mixers and translators
- does not:
  - provide any mechanism to ensure timely delivery or provide other quality-of-service guarantees
  - guarantee delivery or prevent out-of-order delivery
  - assume that the underlying network is reliable and delivers packets in sequence

# RTP (2)

- header format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

V: Version, P: Padding, X: eXtension, CC: CSRC Count, M: Marker, PT: Payload Type

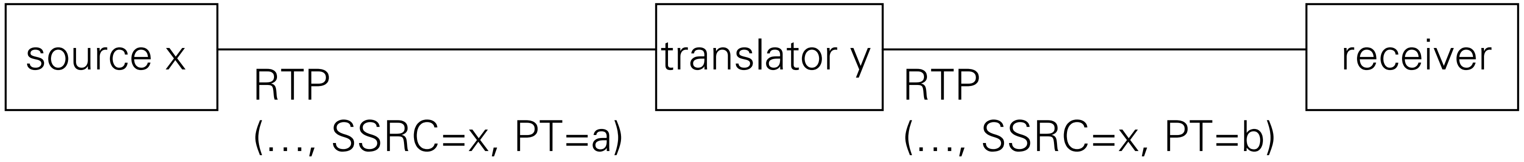
# RTP (3)

- payload types:
  - static payload types: 0 – 95
  - dynamic payload types: 96 – 127 (defined during signaling)
  - examples:

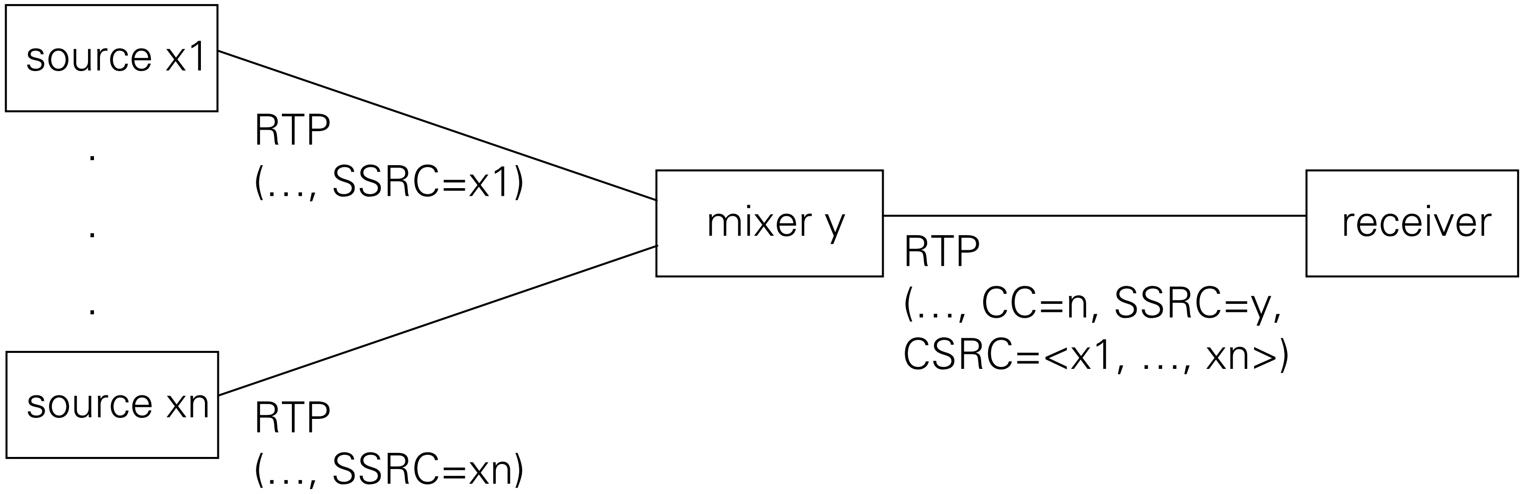
| PT Number | Coding Name | Coding Type | Bit Rate (kbit/s) |
|---|---|---|---|
| 0 | PCM µ Law | sample-based | 64 |
| 4 | G.723 | frame-based | 5.3 / 6.3 |
| 8 | PCM A Law | sample-based | 64 |
| 15 | G.728 | frame-based | 16 |
| 18 | G.729 | frame-based | 8 |
| dynamic | G.726-32 | sample-based | 32 |
| dynamic | G.726-16 | sample-based | 16 |

# RTP (4)

- translator: coding transformation




- mixer: mixing several audio streams into a single audio stream



# RTCP (1)

- functions:
  - monitors the quality of service
    => sender may adapt audio coding
    (low quality of service => reduce data rate)
    - messages of types SR, RR, XR
  - identification of sources (canonical name)
    - messages of type SDES
  - support for multi-point communication
    (information on leaving conference members)
    - messages of type BYE
  - application-specific information
    - messages of type APP

# RTCP (2)

- RTCP header format:
  - similar to RTP header format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|    RC   |      PT       |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         SSRC of sender                        |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

V: Version, P: Padding, RC Report Count, PT: Payload Type = Message Type, SSRC: Synchronization Source

## RTCP (3)

- message type RR (Receiver Report):
  - used if receiver has not sent any data packets during the interval since issuing the last report
  - includes zero or more reception report blocks, one for each of the synchronization sources from which this receiver has received RTP data packets since the last report
  - report block contains: fraction lost, cumulative number of packets lost, extended highest sequence number received, interarrival jitter, …

# RTCP (4)

- message type SR (Sender Report):
  - used if receiver acted also as a sender during the interval since issuing the last report
  - includes zero or more reception report blocks (like RR) and additionally timestamps, sender's packet count, sender's octet count, …
- SR / RR reports are transmitted periodically
- more than one RTCP message may be contained within a single UDP datagram

# SRTP

- Secure RTP, RFC 3711
  - privacy
  - integrity
  - authentication
  - no replay

## CRTP / ROHC (1)



- Compressed RTP, RFC 2508 /
  Robust Header Compression, RFC 3095
- header overhead:

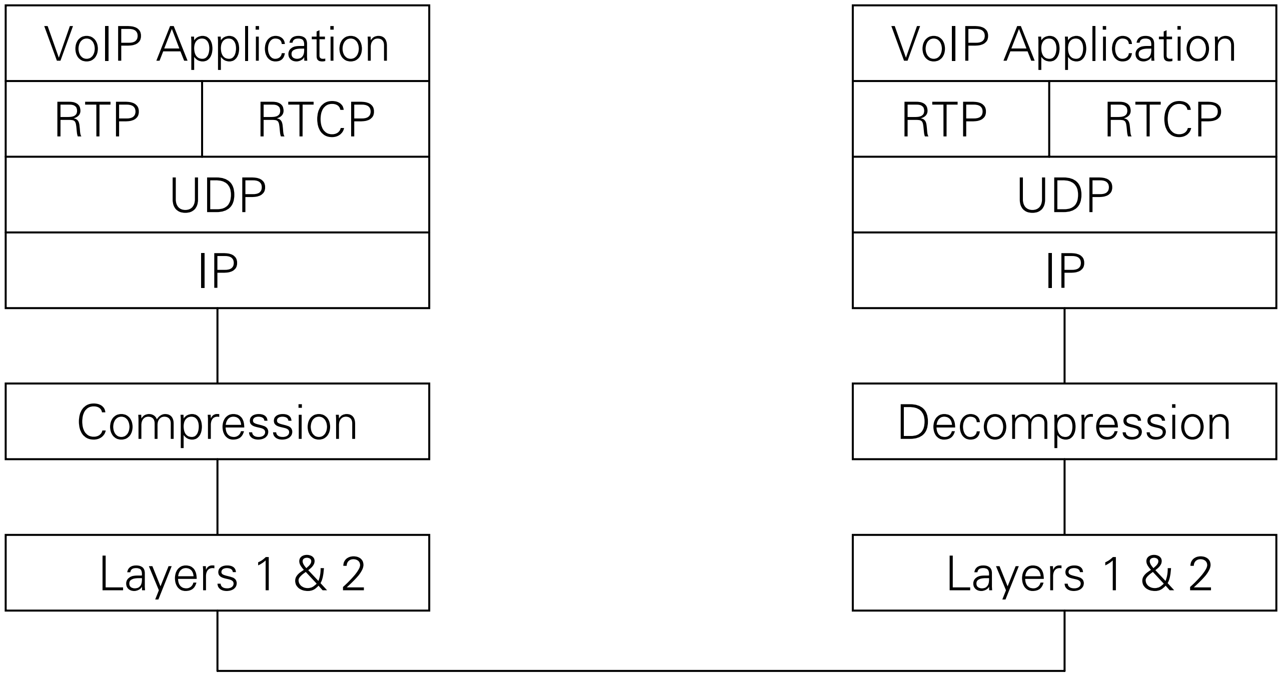
| | |
|---|---|
| RTP header: | ≥ 12 bytes |
| UDP header: | 8 bytes |
| IP header: | ≥ 20 bytes |
| total: | ≥ 40 bytes |

# CRTP / ROHC (2)

- problem (for low bandwidth links, and short payloads): header overhead, although nearly the same information is contained in every packet
- solution: compression
- approach: sending full header first, afterwards only deltas
  => reduction down to 2 bytes possible

# CRTP / ROHC (3)

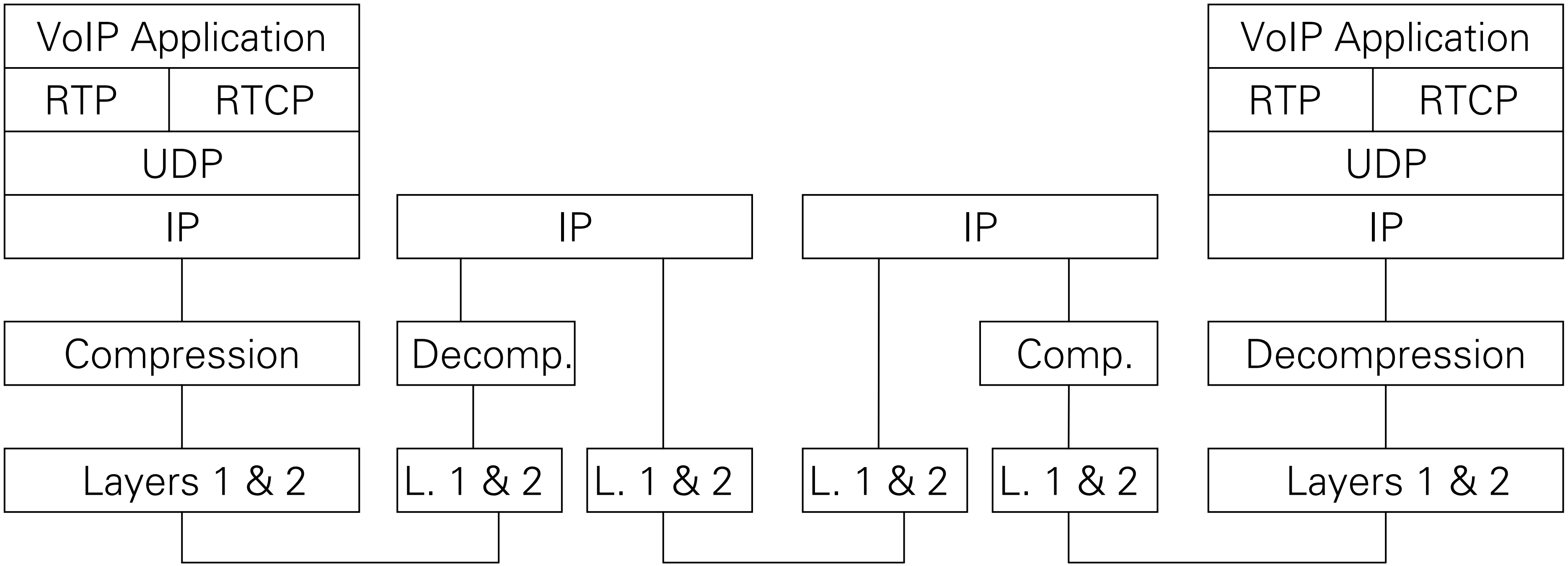
- end-to-end compression:



## CRTP / ROHC (4)



- compression on low bandwidth links only:



- Audio Codecs
- Data Transport (RTP, RTCP)
- <u>Addressing</u>
- Signaling (SIP, H.323, IAX2)

## Addressing

- format of SIP addresses:
  <user | phone number>@<domain | hostname | IP address> [;transport=udp|tcp]
- examples for SIP addresses:
  romeo@abc.de
  romeo@abc.de; transport=udp
  romeo@143.93.53.147
  06518103508@gateway.de
- simplest form:
  name after @ (abc, 143.93.53.147, gateway.de):
  name or address of partner or SIP server

- Audio Codecs
- Data Transport (RTP, RTCP)
- Addressing
- <u>Signaling (SIP, H.323, IAX2)</u>

# Signaling (1)

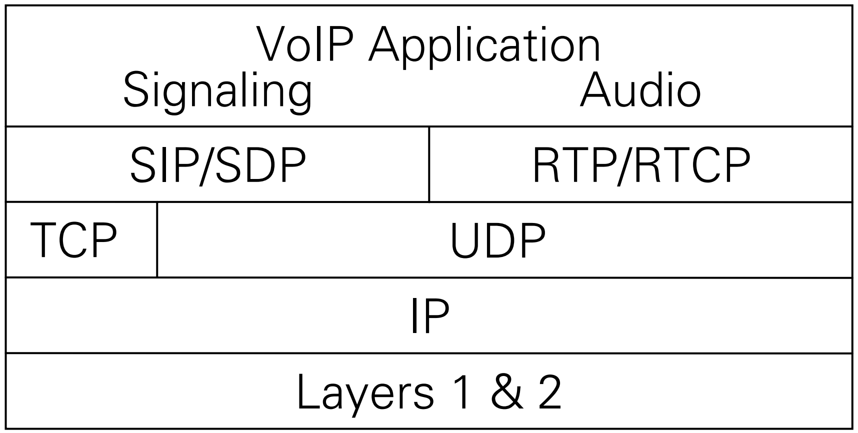
- main function of signaling in PSTN:
  connection setup and teardown (needed for circuit-switching)
- VoIP:
  - use of connectionless packet-switching
  - on top of it: use connectionless UDP transport protocol
  - so why is signaling needed?

## Signaling (2)

- signaling in VoIP:
  - locating partners
    (people may use different devices (PC, notebook, handheld)
    => setup should be done to a certain person, not to a
    certain device)
  - agreeing on port numbers for RTP / RTCP sessions
  - agreeing on coding / decoding procedures (codecs)
  - used media (audio / video)
- signaling protocols:
  - SIP
  - H.323
  - IAX2 (proprietary)

# SIP (1)

- Session Initiation Protocol
  RFCs 3261, 3262, 3263, 3264, 3265
- on top of UDP (TCP also possible)

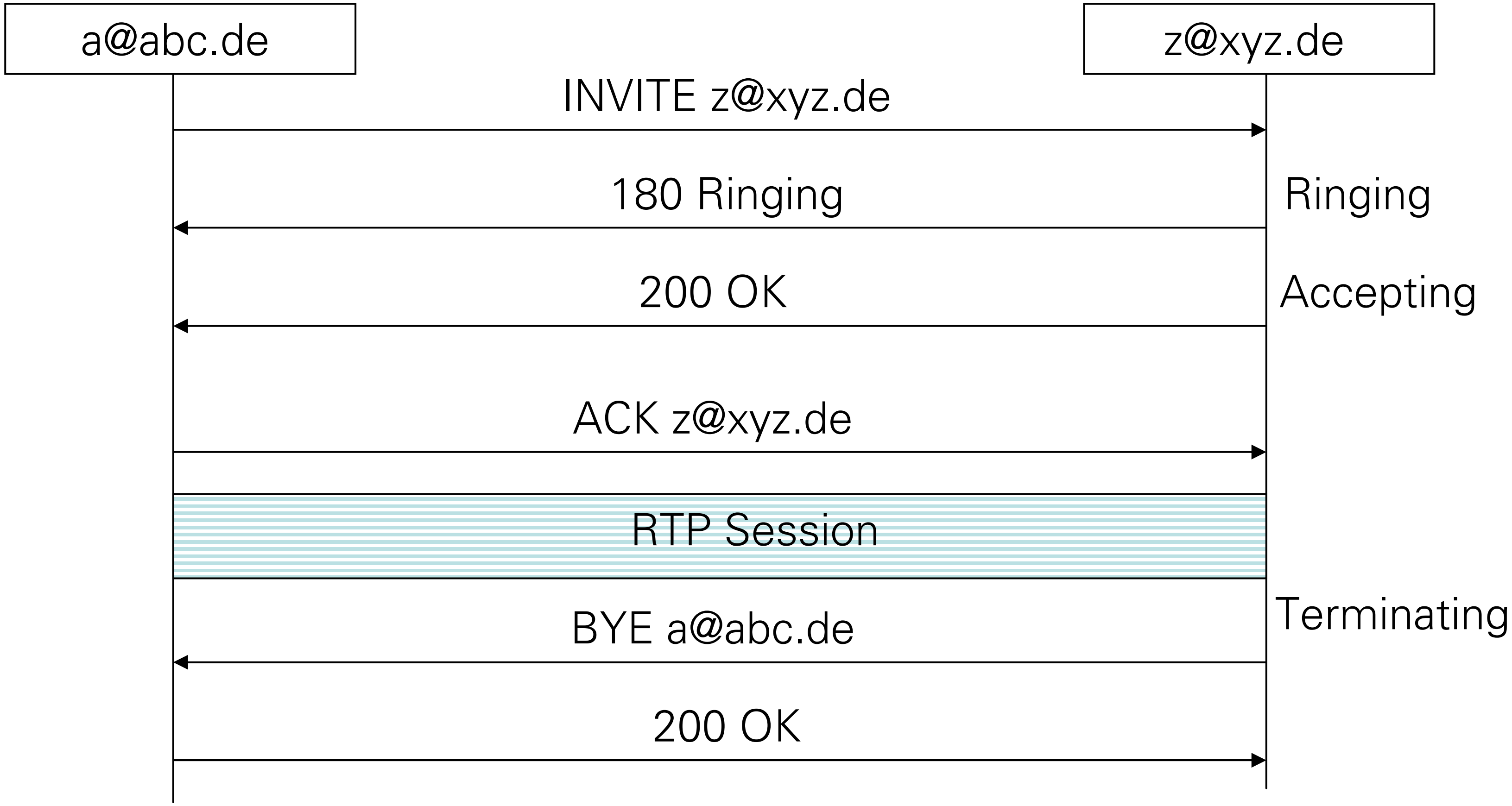


## SIP (2)

- SIP: not only for VoIP, but in general for multimedia communication
    - audio / video conferencing, server-based or p2p
    - special cases: VoIP, UMTS
- well-known port number: 5060
- support for instant messaging
- ASCII protocol like SMTP and HTTP
- format similar to HTTP

# SIP (3)

- protocol overview:

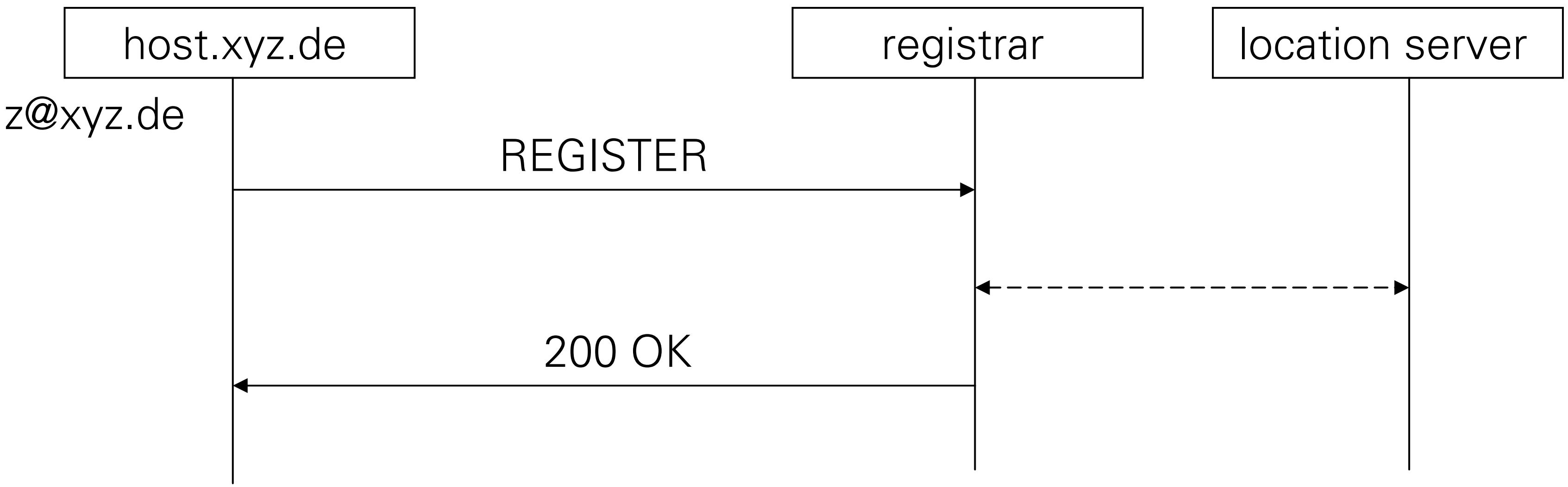


# SIP (4)

- protocol overview (cont.):
  - INVITE request contains session description (format: SDP [Session Description Protocol], not really a protocol, but a format description)

## SIP (5)

- users may use different devices
  => registering at registrar
- registrar may use a separate location server for storing information
- registrar and location server may be the same (SIP server)



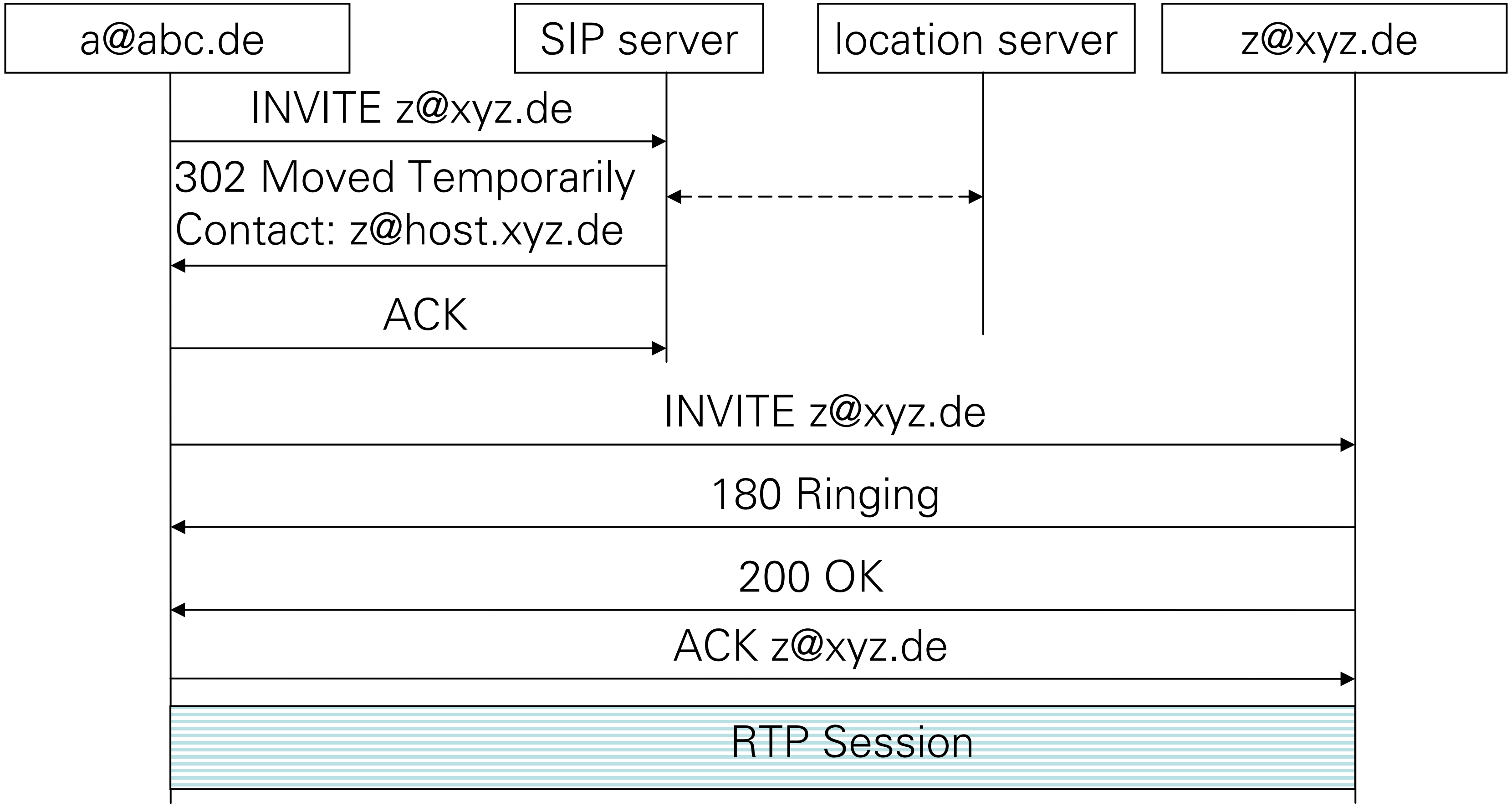
## SIP (6)



```
REGISTER sip:registrar@xyz.de SIP/2.0
Via: SIP/2.0/UDP host.xyz.de
From: sip:z@xyz.de
To: sip:z@xyz.de
Call-ID: 71710@host.xyz.de
CSeq: 1 REGISTER
Contact: <sip:z@host.xyz.de: 3890; transport=udp>
Expires: 1440
```

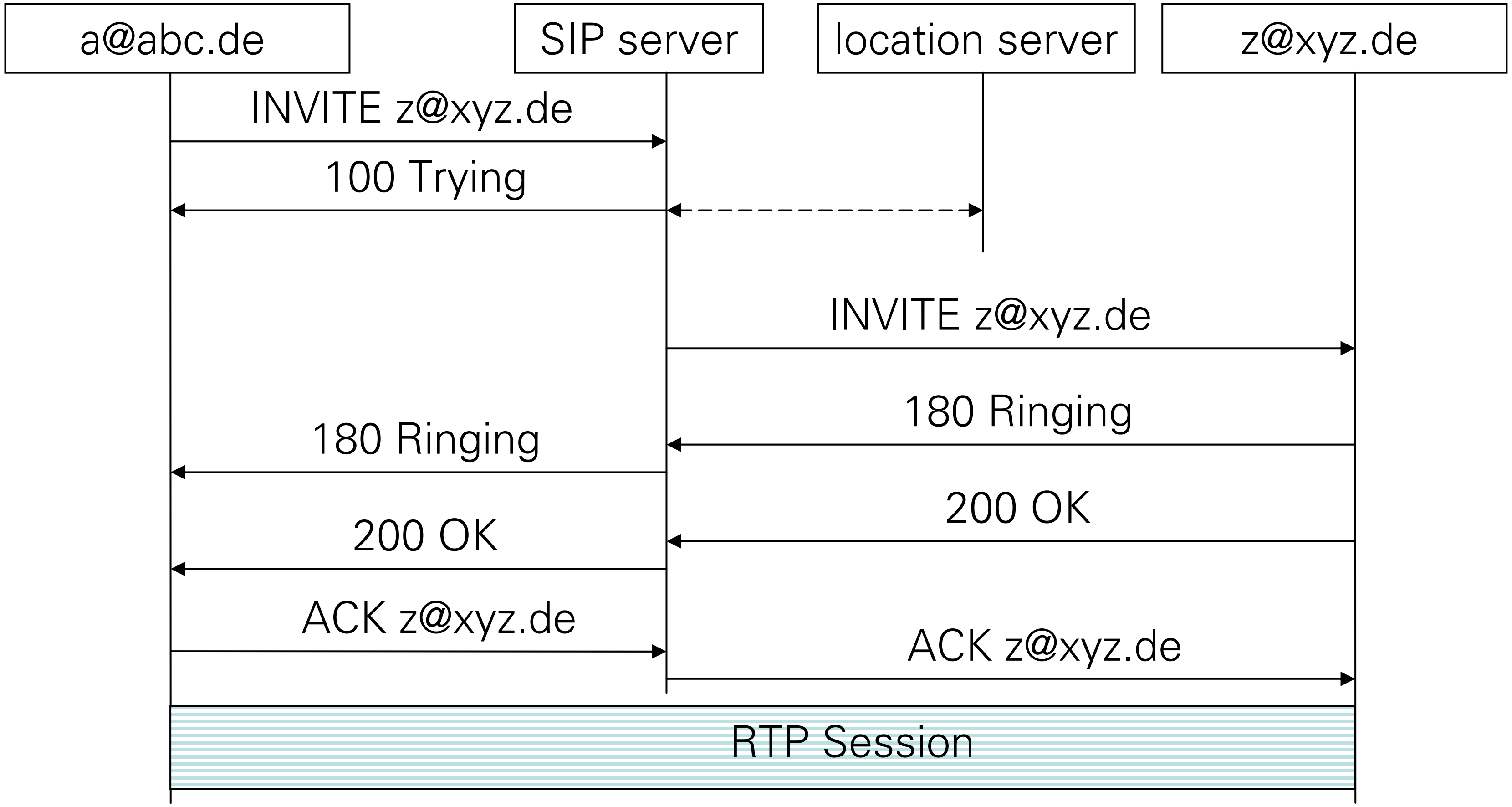
## SIP (7)

- looking up a user; SIP server may act as a redirect server:



## SIP (8)

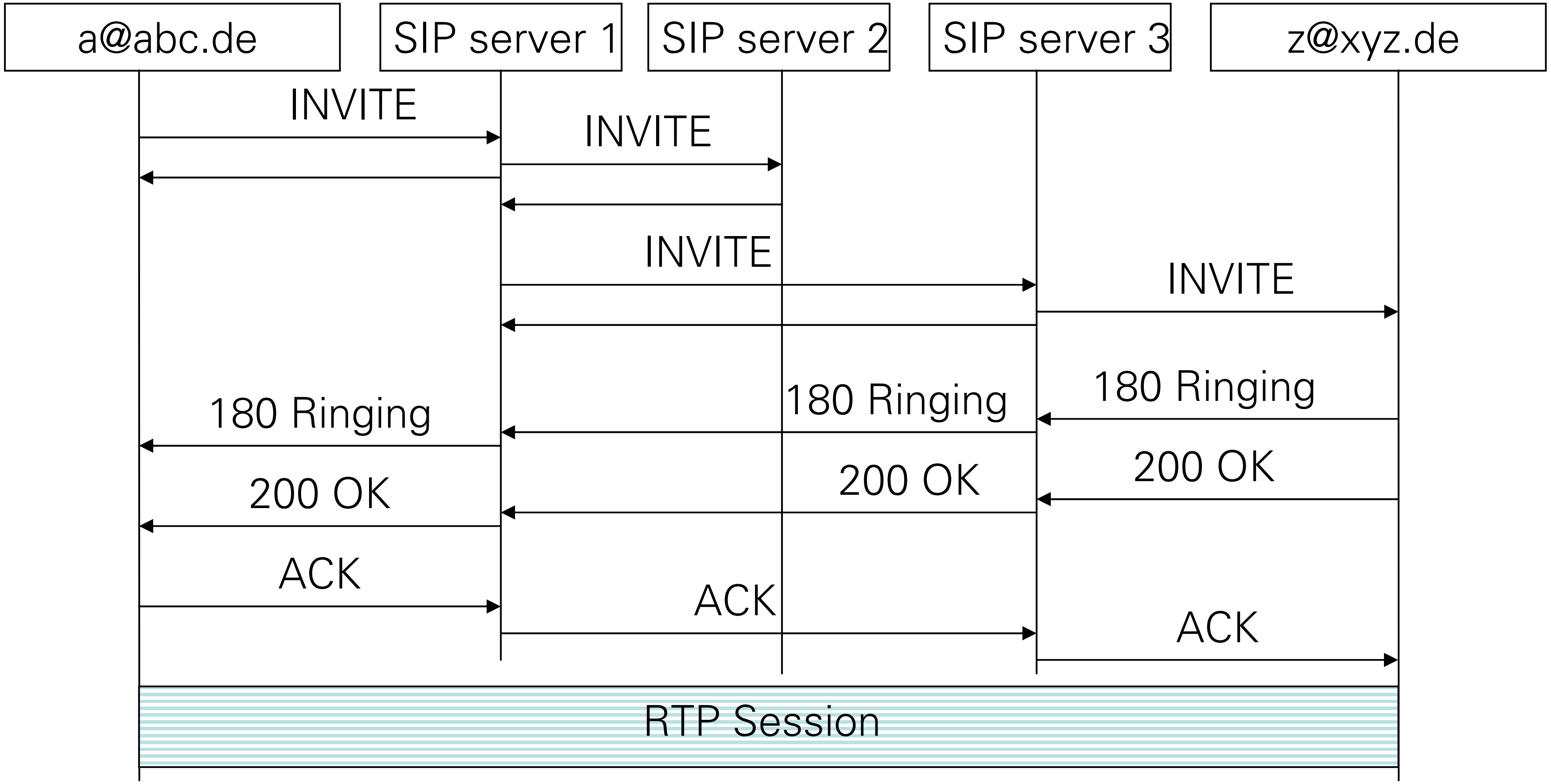
- looking up a user; SIP server may act as a proxy server:



## SIP (9)

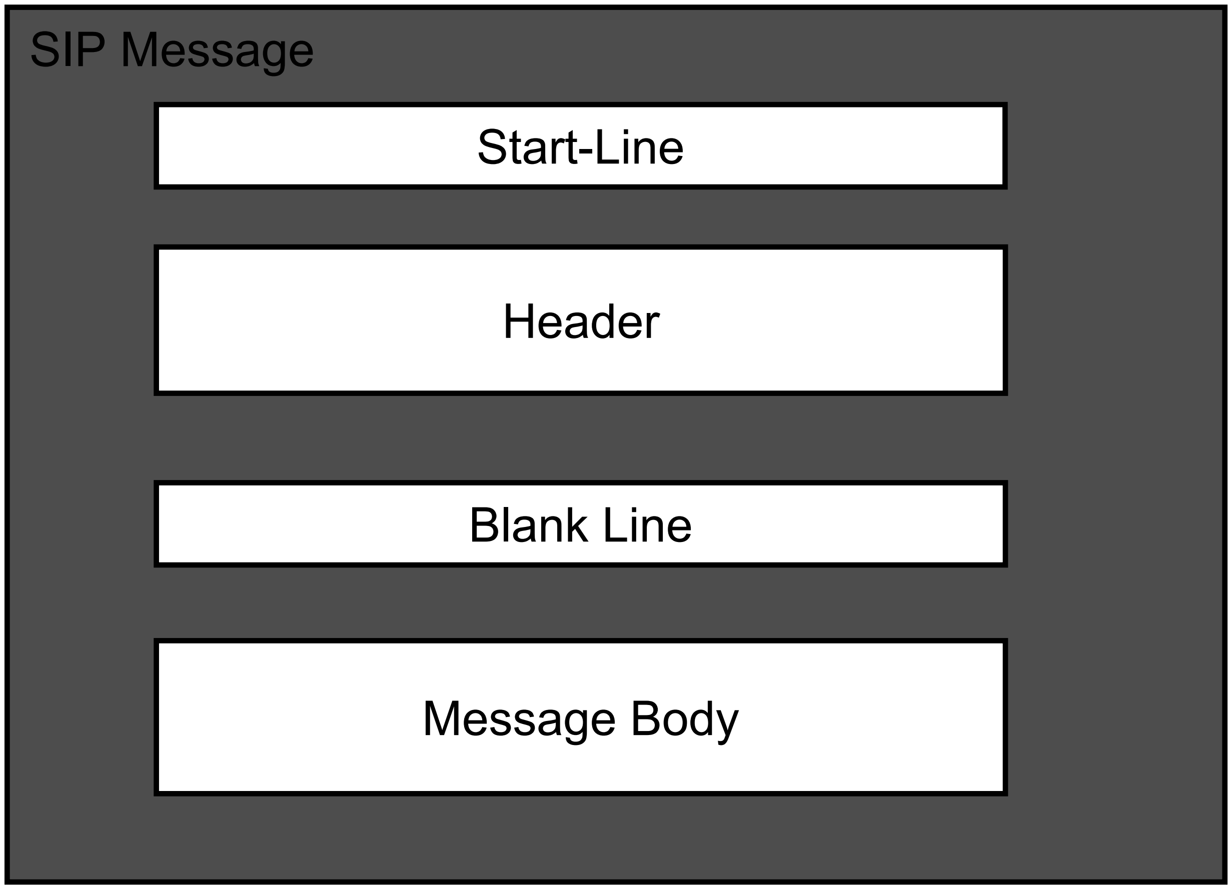


- there may be more than one SIP server in a call setup (both redirect and proxy servers):



# SIP (10)

- SIP protocol format:



# SIP (11)

- SIP protocol format (cont.):
  - request format:

```
<method> <uri> <sip version>
<header1>: <value1>
...
<headerN>: <valueN>
                                                //empty line
<message body (optional)>
```

## SIP (12)



- SIP protocol format (cont.):
  - request (method) types:

    INVITE
    BYE
    ACK
    OPTIONS
    CANCEL
    REGISTER
    INFO
    PRACK
    UPDATE
    MESSAGE
    REFER
    SUBSCRIBE
    NOTIFY

# SIP (13)

- SIP protocol format (cont.):
  - response format:

```
<sip version> <status code> <reason phrase>
<header1>: <value1>
...
<headerN>: <valueN>
                         //empty line
<message body (optional)>
```

## SIP (14)



- SIP protocol format (cont.):
  - status codes:

1xx: informational
  100 Trying
  180 Ringing

2xx: success
  200 OK

3xx: redirection
  301 Moved Permanently
  302 Moved Temporarily

4xx: client error
  400 Bad Request
  401 Unauthorized
  404 Not Found

5xx: server error
  500 Internal Server Error
  501 Not Implemented

6xx: global failure
  600 Busy Everywhere

# SIP (15)

- SIP protocol format (cont.):
  - headers:
    Via
    From
    To
    Call-ID
    CSeq
    Contact
    Content-Type
    Content-Length
    Accept
    Subject

# SDP (1)

- format of message body in SIP INVITE messages:
  SDP (Session Description Protocol), RFC 2327
- ASCII format
- describes multimedia session:
  - media types
  - media formats
  - port numbers
  - …

## SDP (2)



- structure:

session description
    v=  (protocol version)
    o=  (owner/creator, session identifier, timestamp, IN, IP address type [IPv4], IP address)
    s=  (session name)
    i=* (session information)
    u=* (URI of description)
    e=* (email address)
    p=* (phone number)
    c=* (connection information – not required if included in all media)
    b=* (bandwidth information)
    One or more time descriptions
    z=* (time zone adjustments)
    k=* (encryption key)
    a=* (zero or more session attribute lines)
    Zero ore more media descriptions
time description
    t=  (time the session is active [start time / stop time])
    r=* (zero or more repeat times)
media description
    m=  (media type, port, transport address, list of payload types)
    i=* (media title)
    c=* (connection information – optional if included at session-level)
    b=* (bandwidth information)
    k=* (encryption key)
    a=* (zero or more media attribute lines [codecs for payload types in m line])
*optional

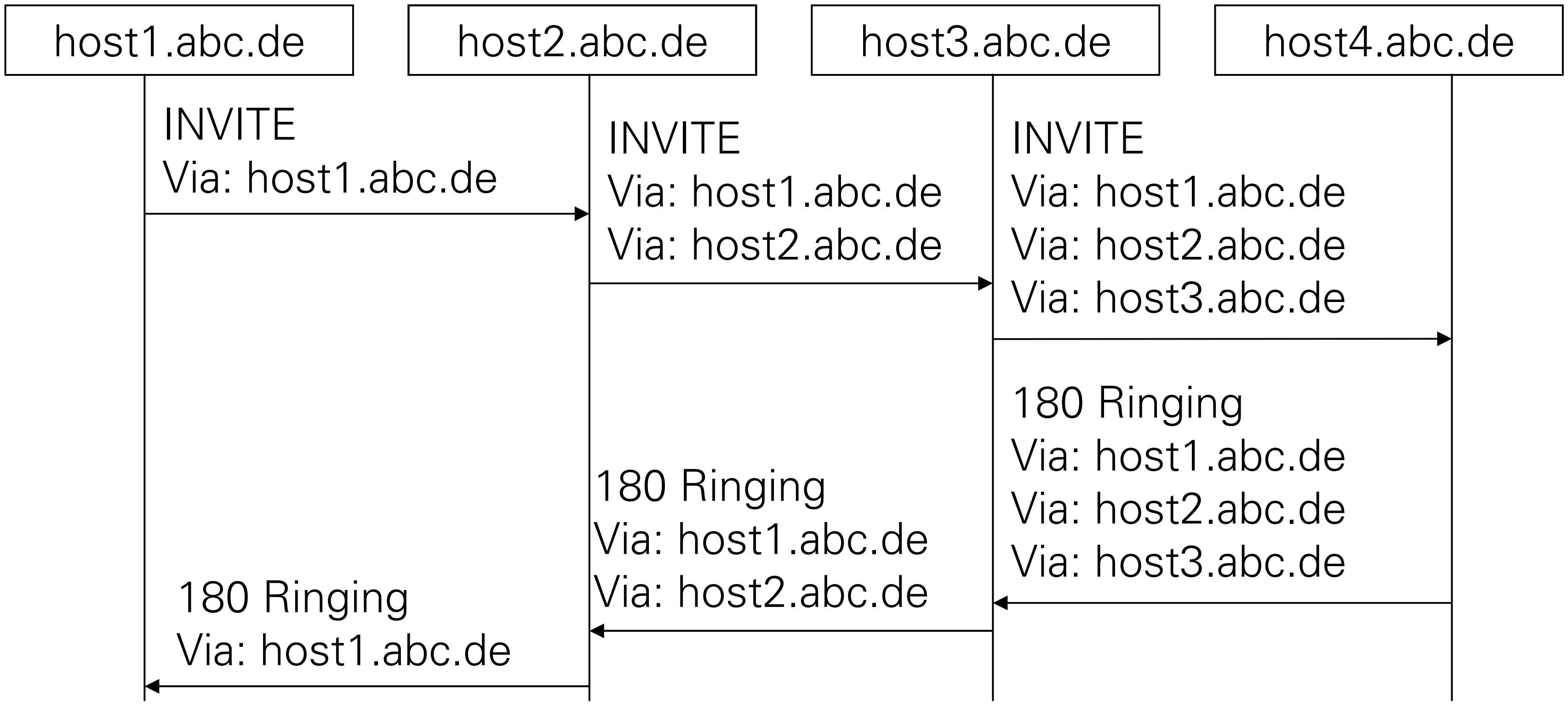
## SDP (3)

- example:

  v=0
  o=oechsle 12345678 1234567 IN IP4 147.168.20.30
  s=SDP Seminarbeispiel
  i=http://www.fh-trier.de/sdp-example.html
  e=oechsle@fh-trier.de
  c=IN IP4 136.199.168.20
  t= 1234567890 1234570000
  m=audio 4711 RTP/AVP 0 98

## SIP (16)

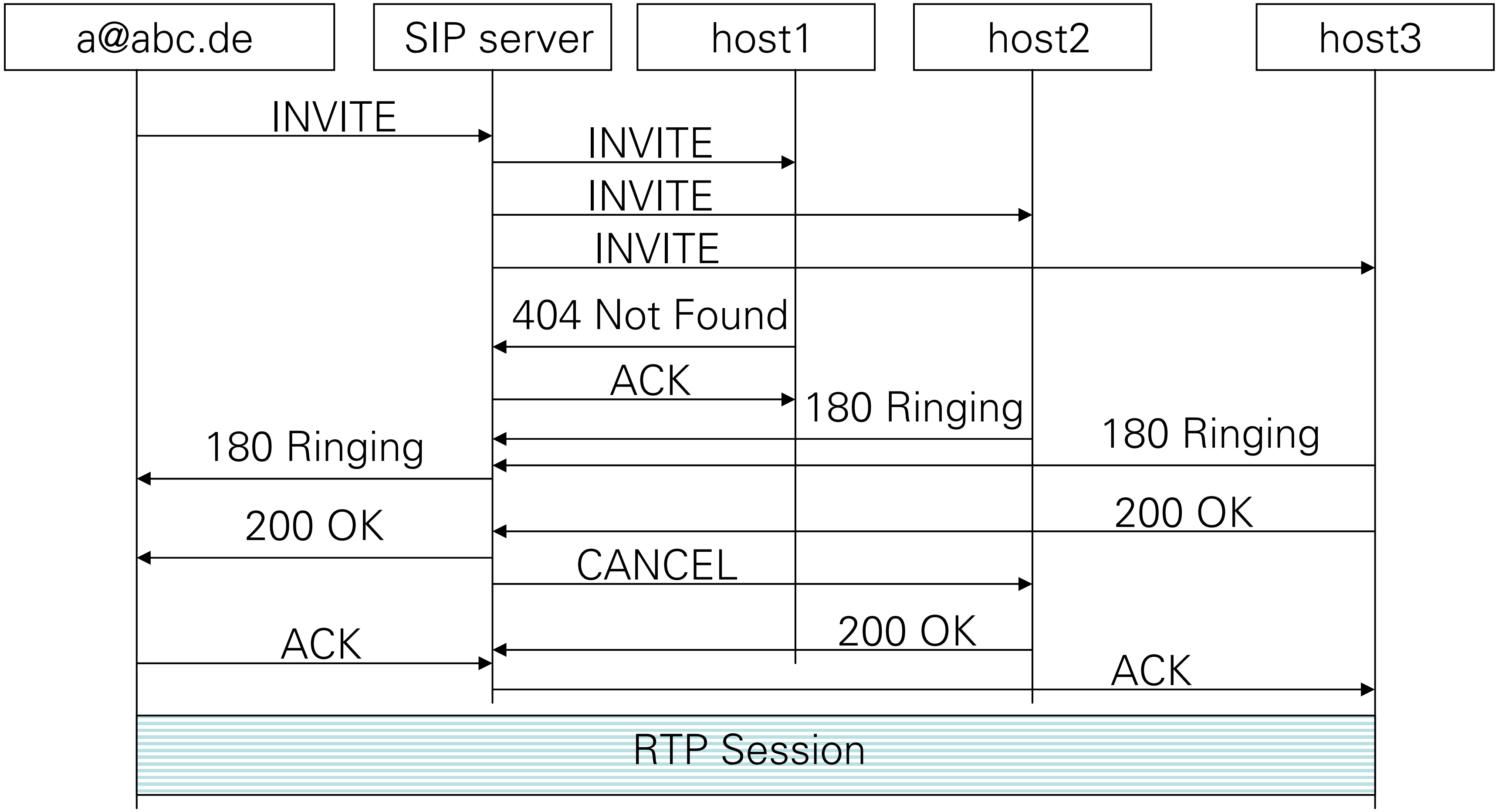
- response routing:



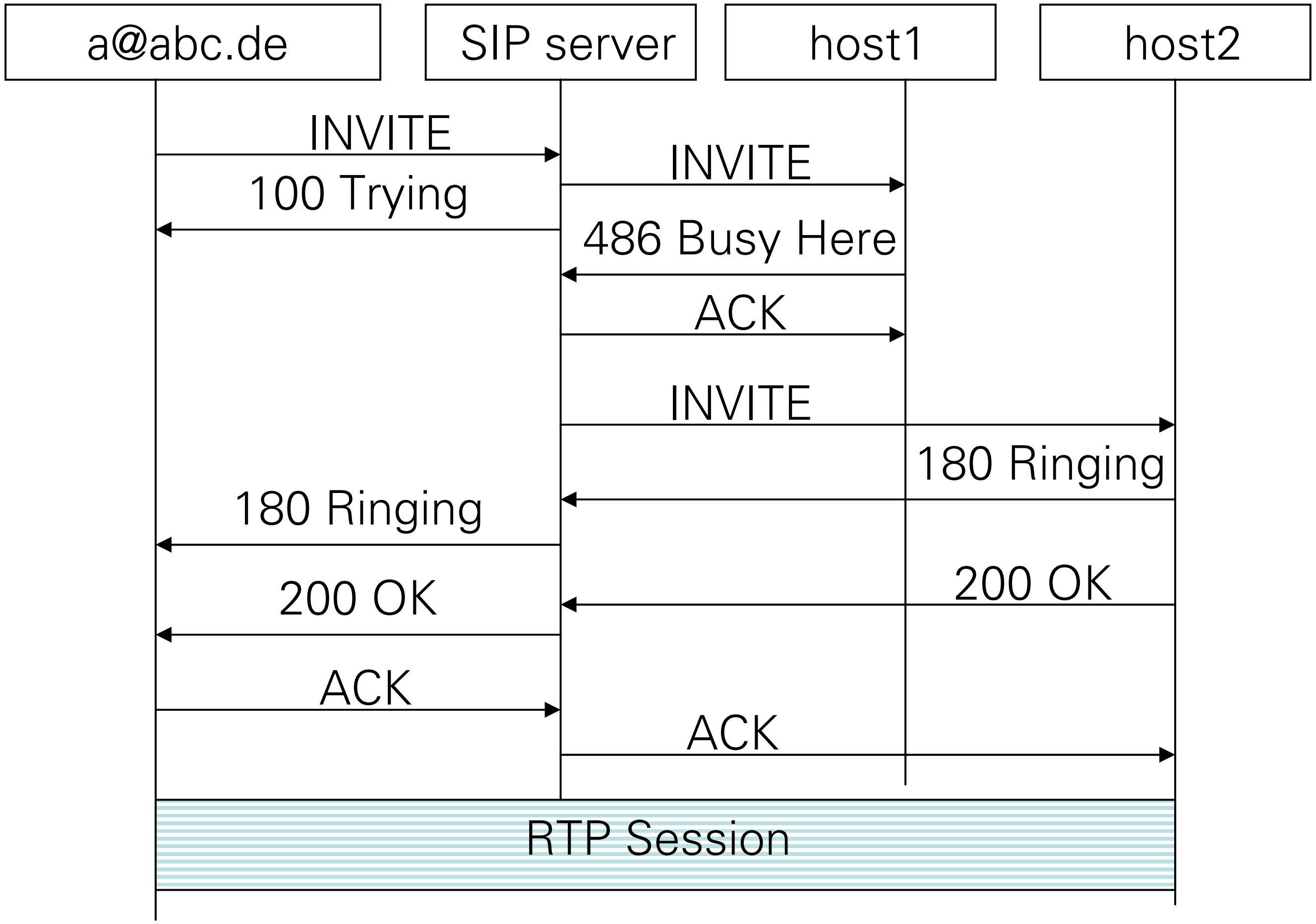

- request routing

## SIP (17)

- additional service of SIP servers: forking proxy

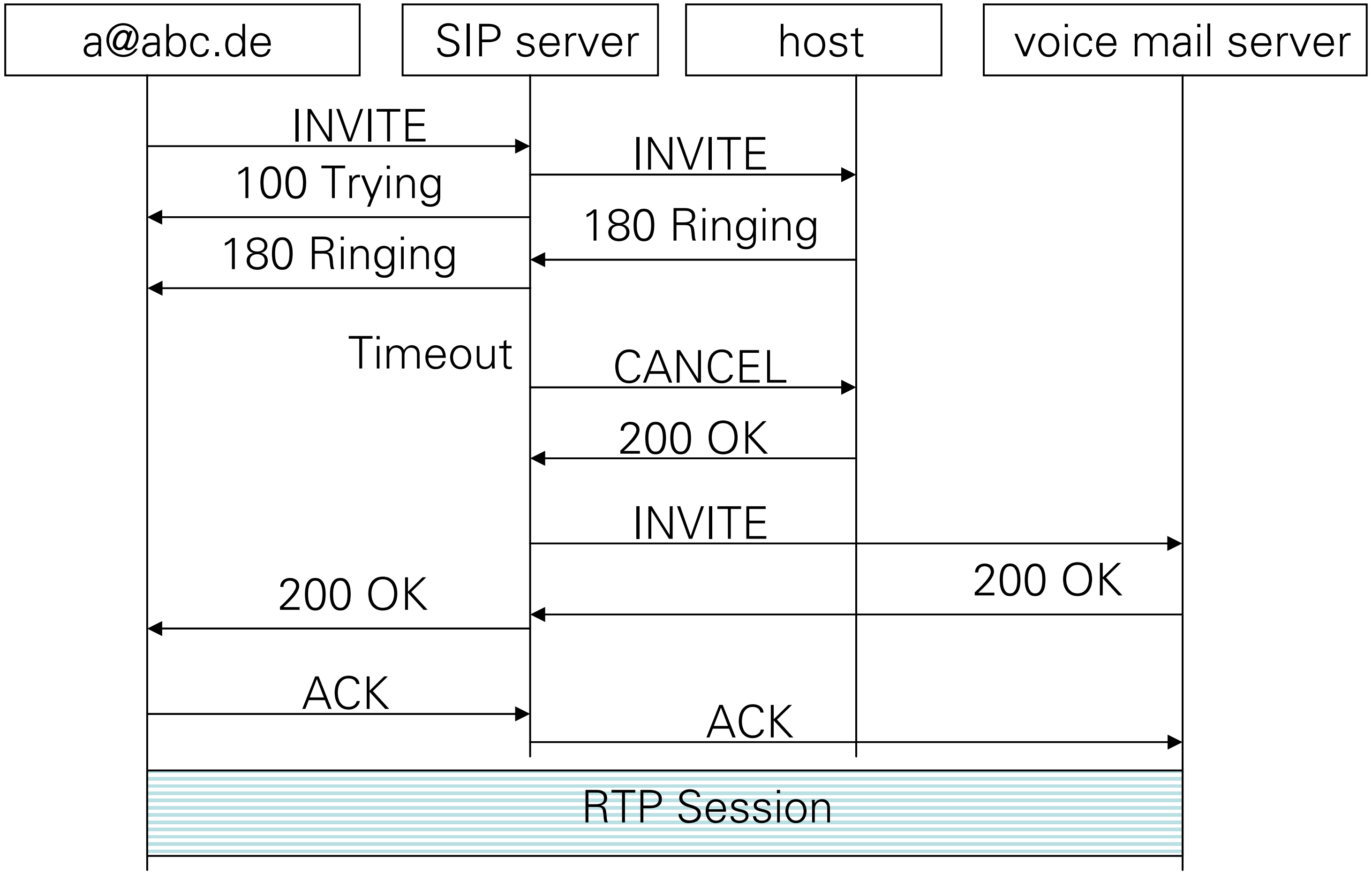


- additional service of SIP servers: call forwarding when busy



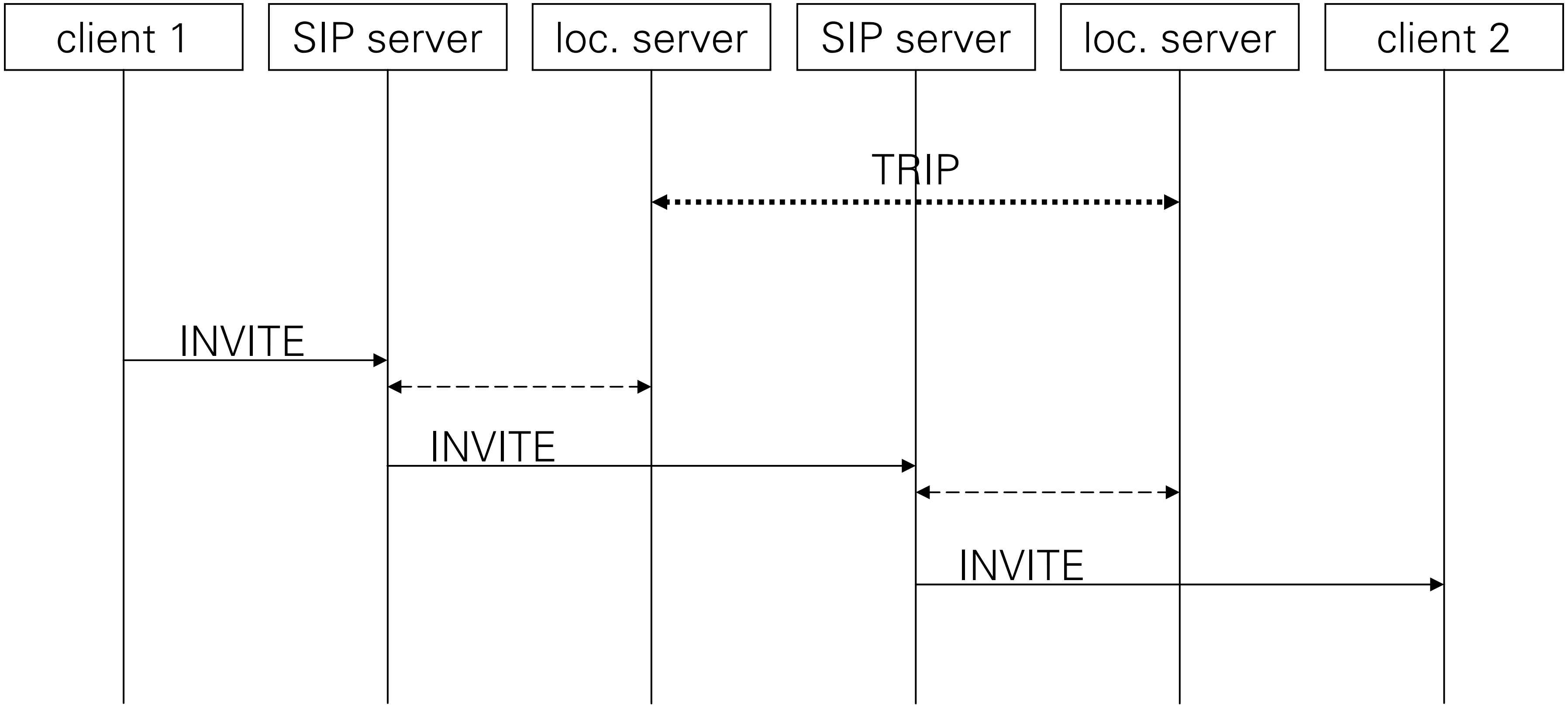
## SIP (19)

- additional service of SIP servers: call forwarding when busy



# SIP (20)

- location servers exchange information about SIP servers using the TRIP protocol (Telephony Routing over IP):



## H.323 (1)



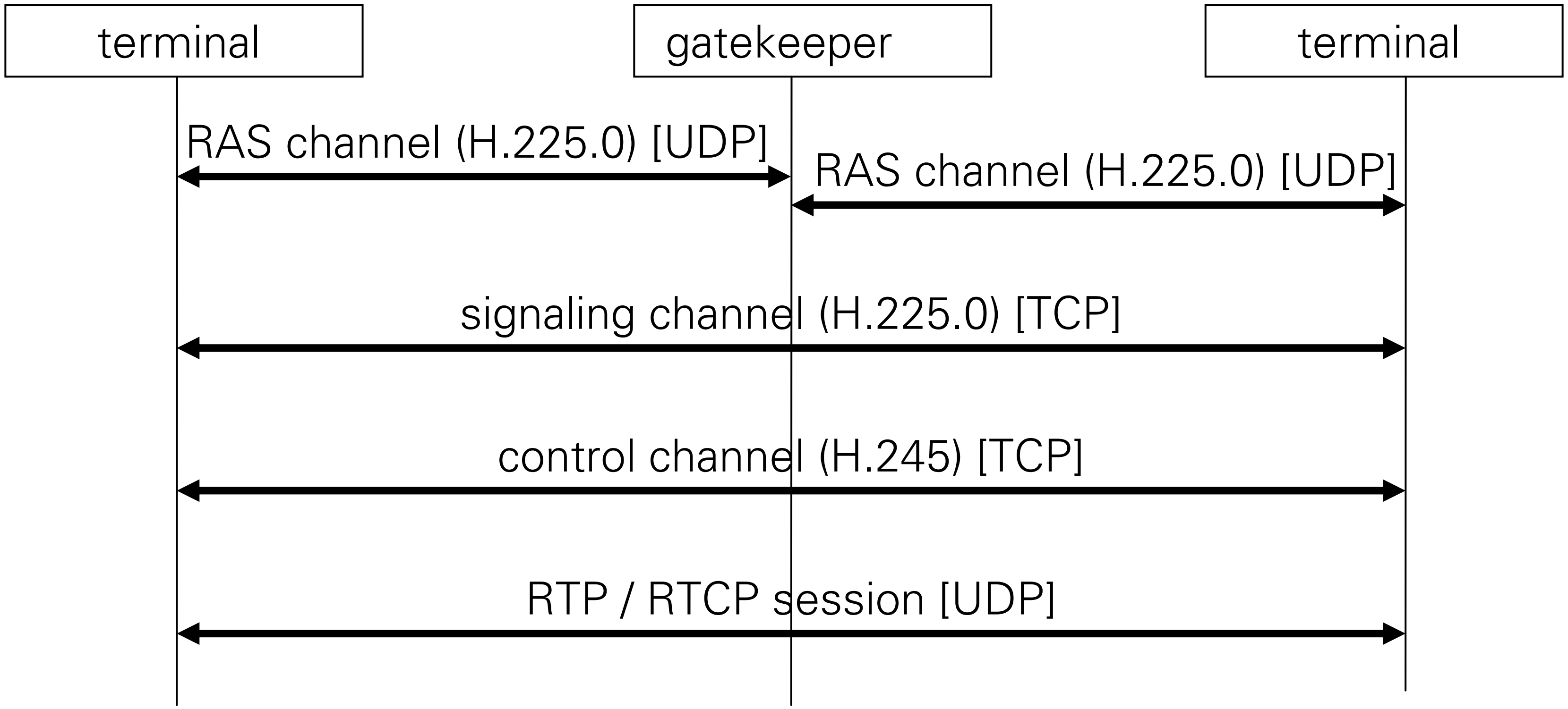
- ITU-T standard (International Telecommunications Union)
- framework for multimedia communication
- based on:
  - H.225.0 (signaling)
  - H.245 (signaling)
  - RTP / RTCP
  - H.450.x (supplementary services)
  - H.510 (roaming)

## H.323 (2)

- gatekeeper (server):
  - controls a group of H.323 terminals (H.323 zone)
  - call admission and resource reservation
  - location server: maps telephone numbers / email addresses to IP addresses
  - in small networks there may be no gatekeeper (=> each terminal needs its own mapping table)

# H.323 (3)

- H.323 channels within an H.323 zone:



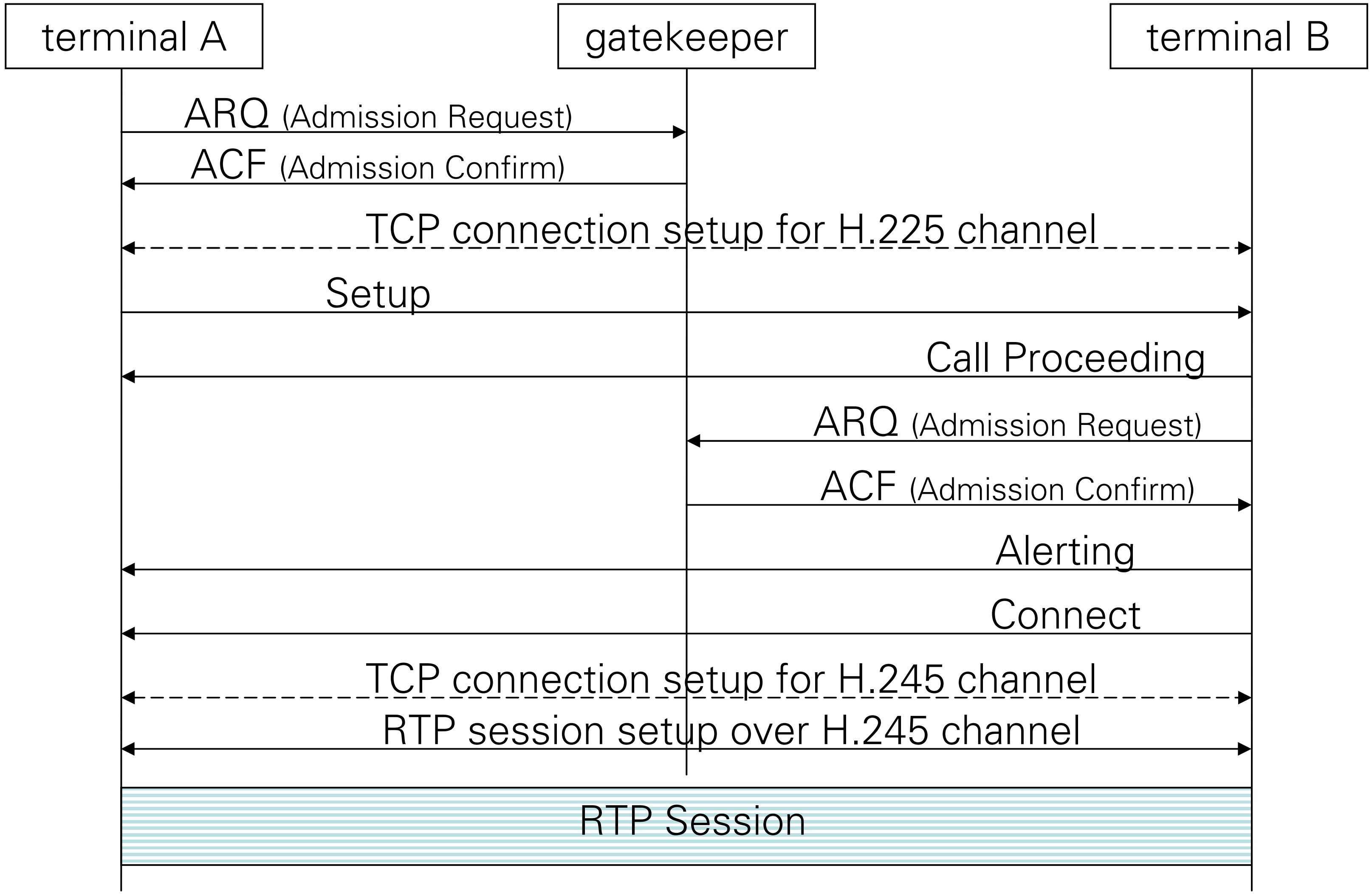
## H.323 (4)



- H.323. channels (cont.):
  - RAS channel: terminal – gatekeeper
    RAS: registration, admission, status
  - H.225.0 channel: terminal – terminal:
    meta-signaling
    => setup of a H.245 control (signaling) channel
  - H.245 channel: terminal – terminal:
    control (signaling) channel
    => setup of an RTP session
  - RTP channel: terminal – terminal:
    data channel, unidirectional
    => two channels needed

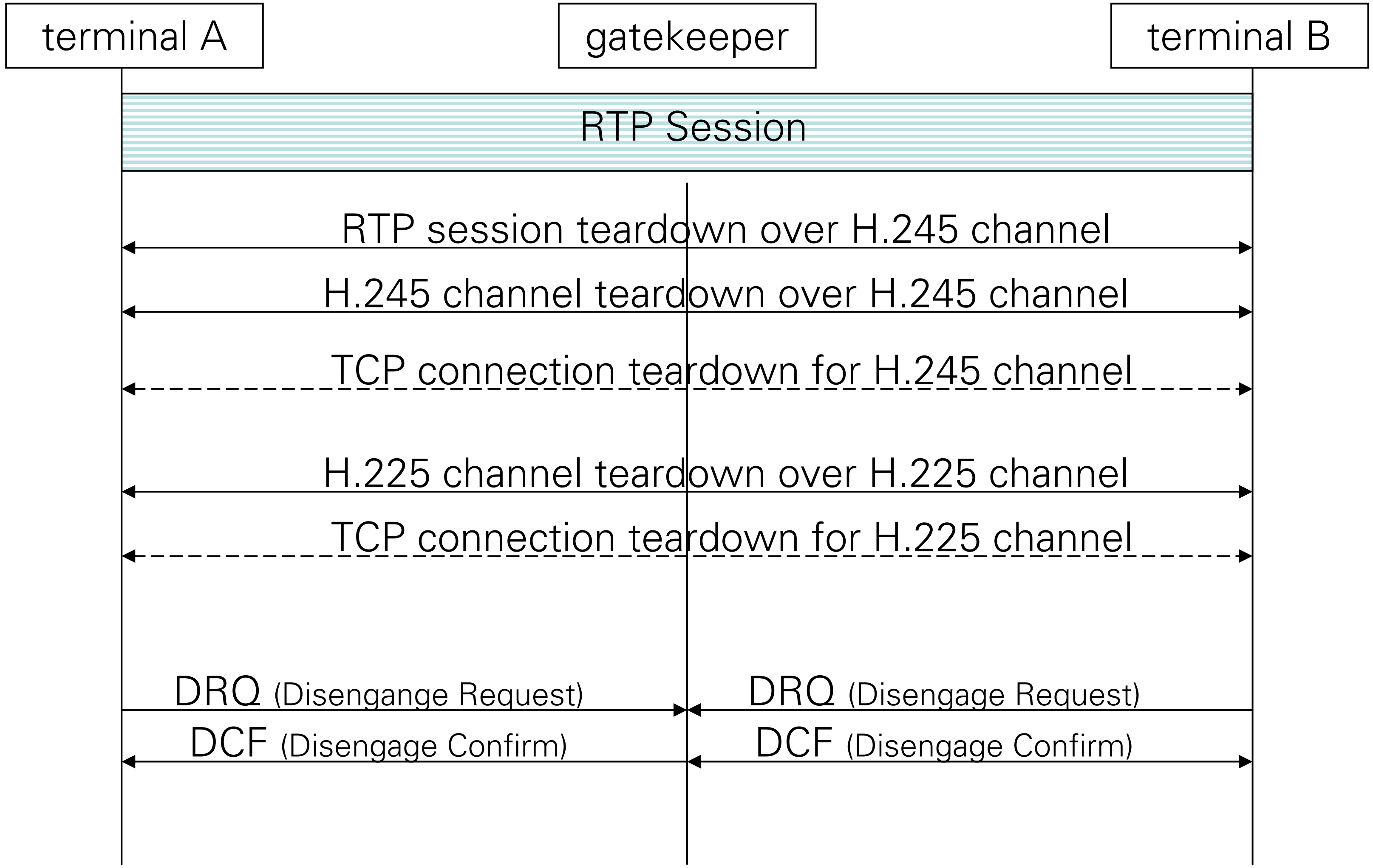
## H.323 (5)

- signaling procedure: call setup



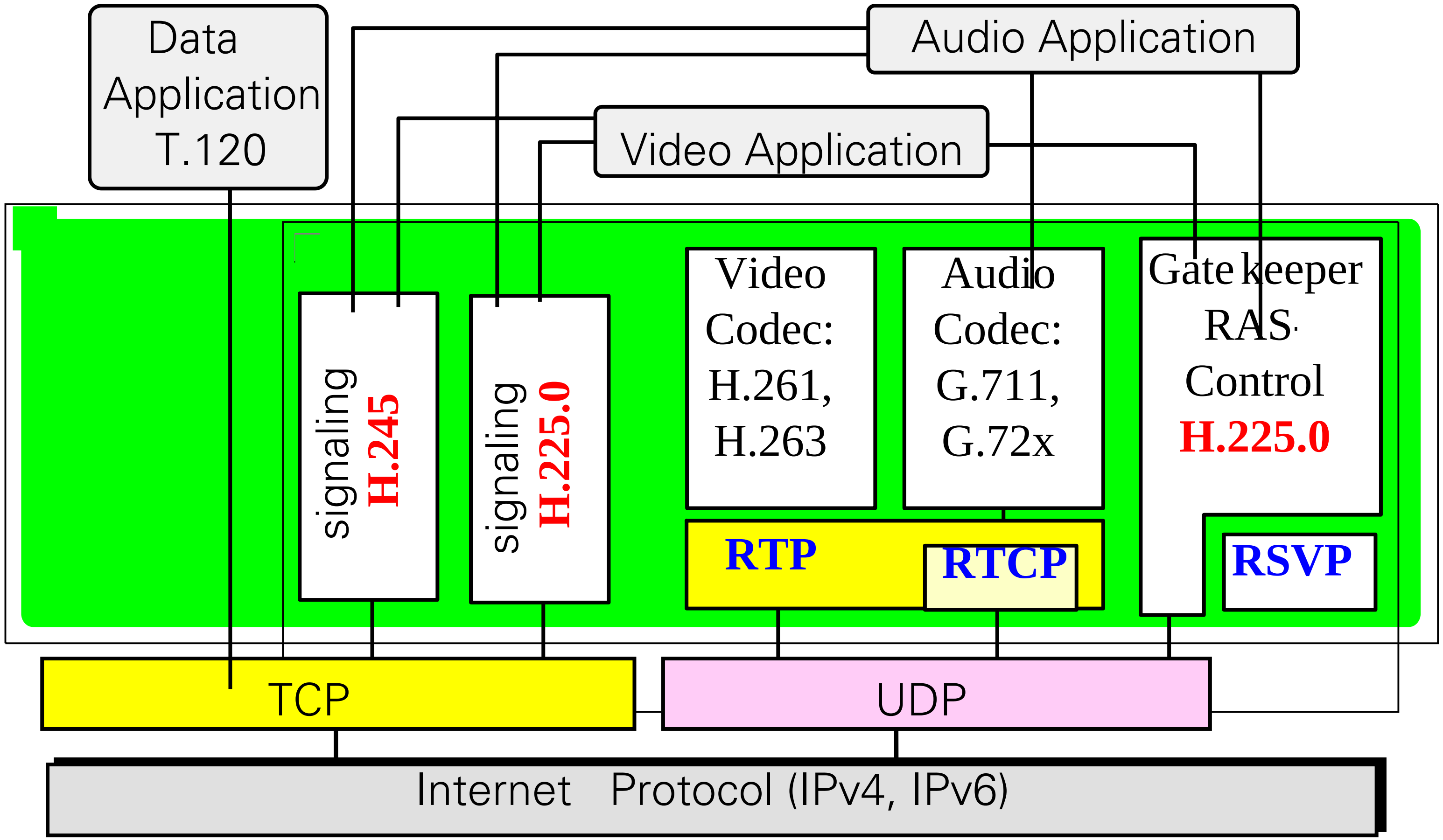
# H.323 (6)

- signaling procedure: call teardown



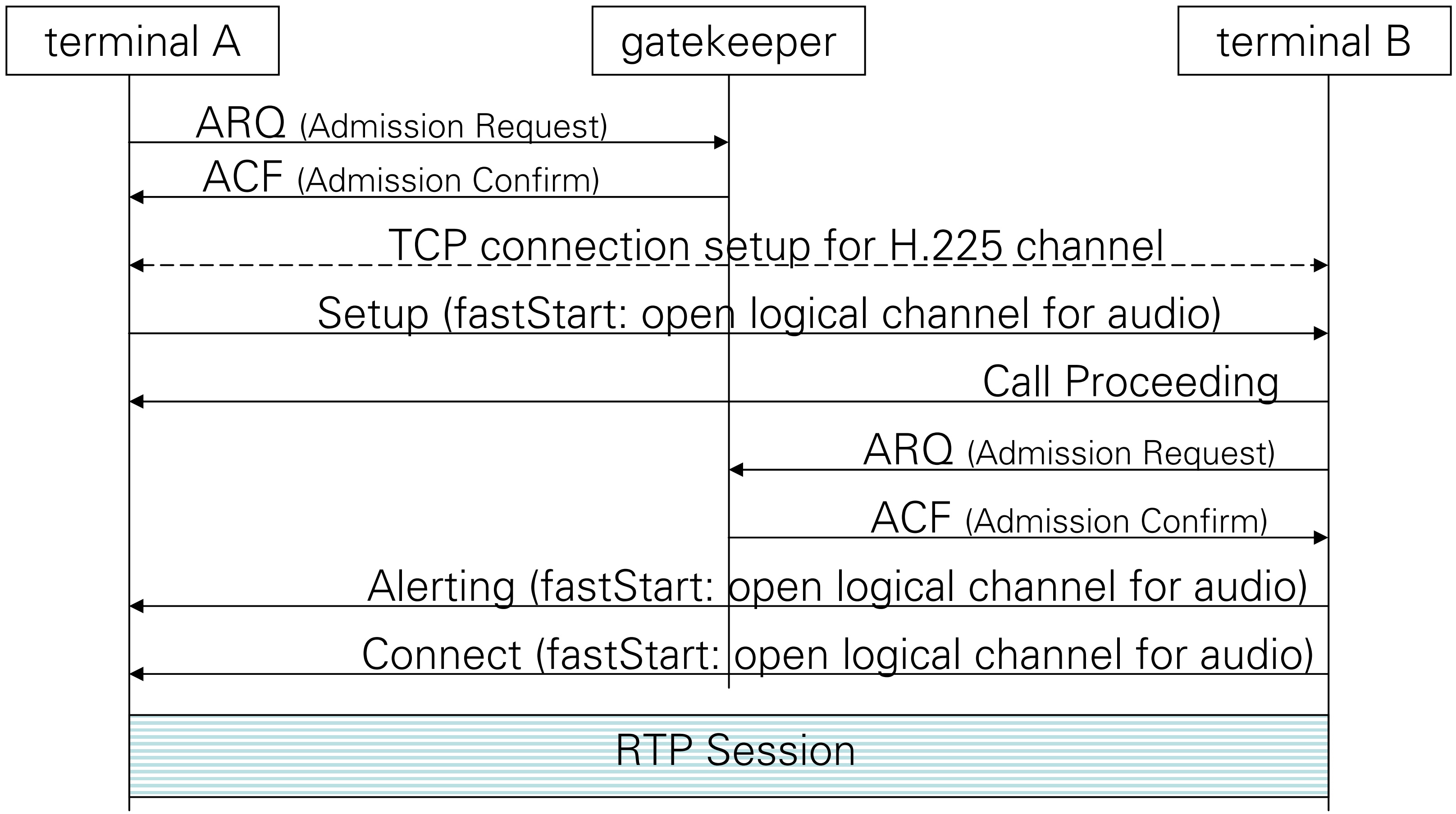
# H.323 (7)

- H.323 protocol family (figure taken from Anatol Badach):



## H.323 (8)

- fast connect procedure: H.245 messages can be transmitted within the H.225.0 channel => no H.245 channel needed

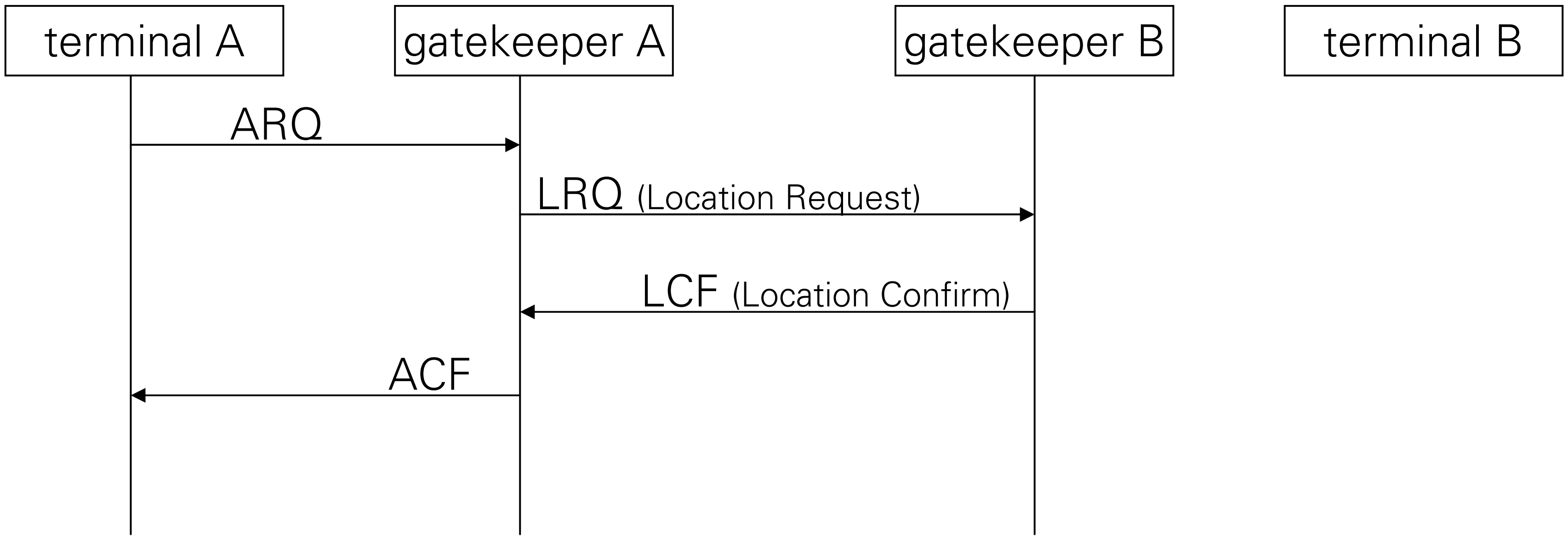


## H.323 (9)

- RAS messages (H.225.0):
  - gateway discovery (using multicast)
  - terminal registration at gatekeeper
    (telephone numer => IP address)
  - unregister
  - admission request (bandwidth reservation, also includes resolution of telephone number to IP address)
  - disengage
  - bandwidth (change admission request)
  - location request (between gatekeepers)

# H.323 (10)

- RAS message location request:



## H.323 (11)

- H.225.0 for signaling:
  - similar to Q.931 for ISDN
    => “D channel protocol over TCP“
  - direct (end-to-end) signaling between terminals (as shown)
  - or gatekeeper may act as a proxy (like in SIP)

## H.323 (12)

- H.245:
  - capability exchange (available audio/video codecs, encryption capabilities)
  - master / slave determination
  - setup / teardown of RTP sessions for audio/video transmission (codecs, port numbers)
  - modification of parameters for RTP sessions (codecs)

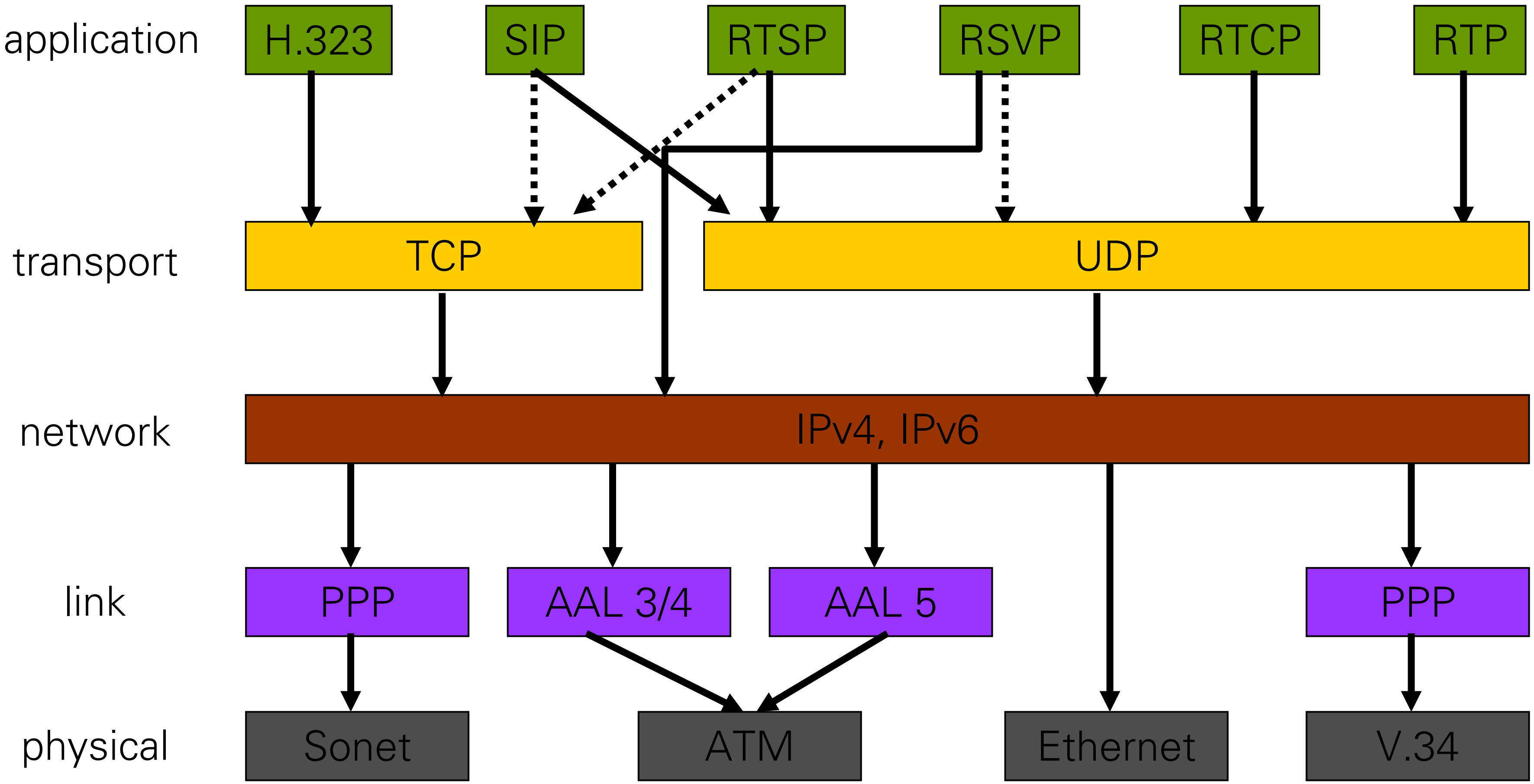
## H.323 (13)



- H.450.x:
  - call transfer: connection between A and B may be transferred to a connection between B and C by A
  - call forwarding (like SIP redirect)
  - call park and pickup at a different terminal
  - …

# Protocol Family

FACHHOCHSCHULE TRIER
Hochschule für Technik, Wirtschaft und Gestaltung
University of Applied Sciences
Informatik



# IAX2 (1)

- Inter-Asterisk eXchange (version 2)
- Asterisk signaling protocol
- collaborative, community-based effort
- Internet draft exists
- IAX2 is also used for data transmission (not RTP / RTCP)
- => IAX2 multiplexes signaling and multiple media streams over a single UDP association

# IAX2 (2)

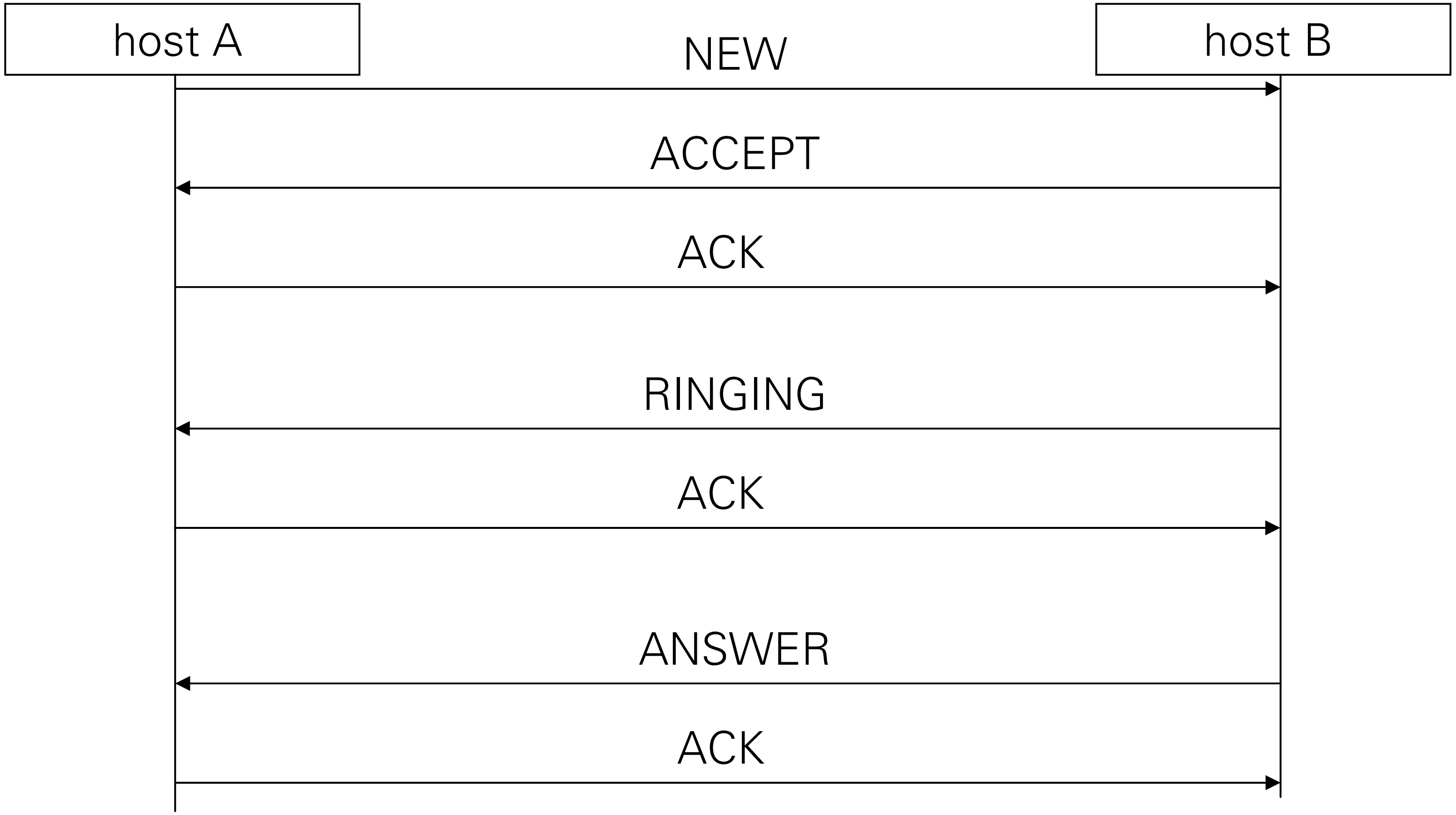


- UDP association uses well-known port number (4569) on both sides => no problems by using firewalls
- p2p protocol
- binary protocol (no ASCII protocol like SIP)
- 15-bit number (Call Number) used for multiplexing
  - 0: (yet) unknown
  - different call numbers for both directions

## IAX2 (3)

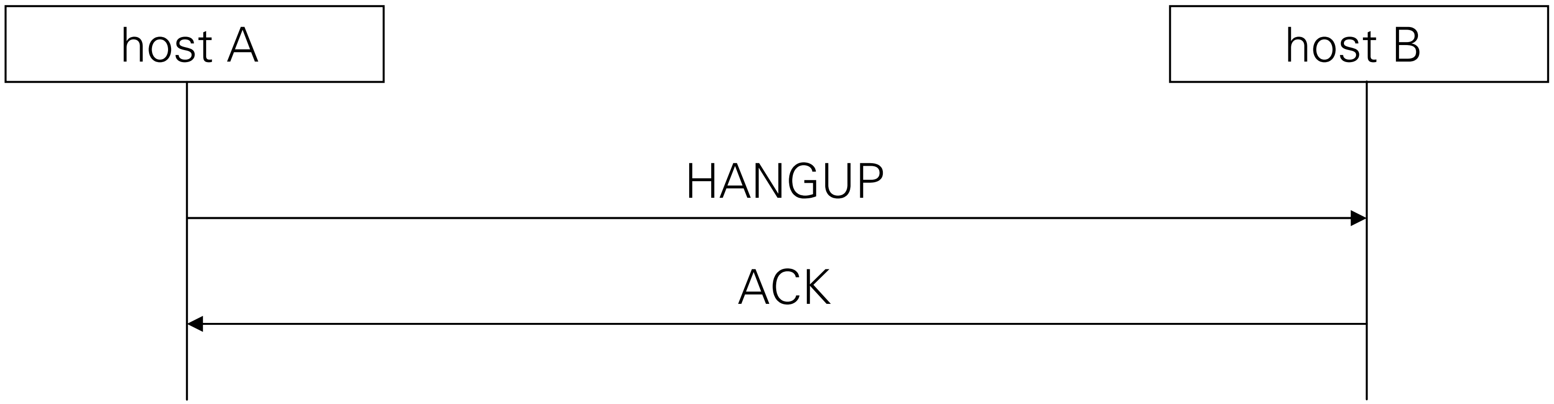


- basic message flow for call setup:



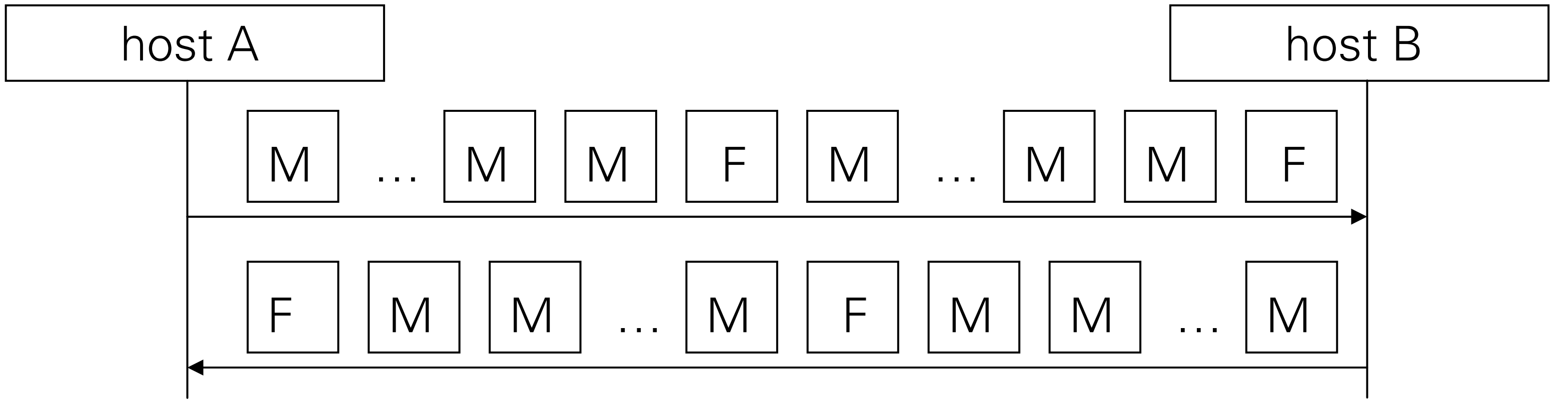
# IAX2 (4)

- basic message flow for call teardown:



## IAX2 (5)

- media flow (in both directions):
  - full frames (acknowledged), contains synchronization information
  - mini frames (not acknowledged), simple 4-byte header



# IAX2 (6)

- full frame format:

```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F| Source Call Number          |R| Destination Call Number     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| OSeqno        | ISeqno        |   Frame Type  |C|   Subclass    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                                 |
|                              Data                               |
|                                                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

## IAX2 (7)

- full frame format (cont.):

```
+-------------+-------------------------------------------+
| FIELD       |                               DESCRIPTION |
+-------------+-------------------------------------------+
| F           | Set to the value 1 indicating that this is a |
|             | full frame                                |
+-------------+-------------------------------------------+
| Source      | Call number of the transmitting side of the |
| Call Number | full frame                                |
+-------------+-------------------------------------------+
| R           | Set to the value 1 if this frame is being |
|             | retransmitted and the value 0 for the initial |
|             | transmission                              |
+-------------+-------------------------------------------+
| Destination | Call number of the receiving side of the full |
| Call Number | frame                                     |
+-------------+-------------------------------------------+
| Timestamp   | Full 32-bit timestamp                     |
+-------------+-------------------------------------------+
| OSeqno      | Outbound stream sequence number           |
+-------------+-------------------------------------------+
| ISeqno      | Inbound stream sequence number (next expected)|
+-------------+-------------------------------------------+
| C           | Subclass value format (1: subclass interpreted|
|             | as a power of 2, 0: subclass interpreted as |
|             | as 7-bit integer)                         |
+-------------+-------------------------------------------+
```

# IAX2 (8)

- full frame format (cont.):

| TYPE | DESCRIPTION | SUBCLASS DESCRIPTION | DATA DESCRIPTION |
|---|---|---|---|
| 0x01 | DTMF | 0-9, A-D, *, # | |
| 0x02 | Voice Data | Audio Compression Format | Raw Voice Data |
| 0x03 | Video | Video Compression Format | Raw Video Data |
| 0x04 | Control | Control Frame Type | |
| 0x05 | Null | | |
| 0x06 | IAX Control | IAX Protocol Message Type | Information Elements |
| 0x07 | Text | | Raw Text |
| 0x08 | Image | Image Compression Format | Raw Image Data |
| 0x09 | HTML | HTML Frame Type | Message Specific |

# IAX2 (9)

- full frame format (cont.):
  - IAX protocol message types:

| | |
|---|---|
| NEW | Initiate new call |
| ACK | Acknowledgement |
| HANGUP | Initiate call teardown |
| REJECT | Reject |
| ACCEPT | Accepted |
| AUTHREQ | Authentication request |
| AUTHREP | Authentication reply |
| PING | Ping request |

| | |
|---|---|
| REGREQ | Registration request |
| REGACK | Registration ack. |
| REGREJ | Registration reject |
| REGREL | Registration release |
| DIAL | Dial |
| QUELCH | Halt av transmission |
| UNQUELCH | Resume av transm. |
| VNAK | Av retransmit req. |

# IAX2 (10)

- mini frame format:

```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F| Source Call Number          |        Timestamp              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                            Data                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

low-order 16 bits of the timestamp: when the 16-bit timestamp wraps around, a full frame is sent for synchronization of its full 32-bit timestamp counter

- Summary:
  - Audio Codecs
  - Data Transport (RTP, RTCP)
  - Addressing
  - Signaling (SIP, H.323, IAX2)
- Conclusions:
  - in a friendly environment: okay
  - but if there is not a sufficient quality of service, not a sufficient degree of security, the addressing scheme ist not flexible enough, ...?

- <u>QoS</u>
- Security (ICE)
- Addressing (DNS SRV/NAPTR – ENUM)

## QoS (1)



- QoS: Quality of Service
- forces against good qos (= IP characteristics):
  - delay (latency)
  - jitter
  - packet loss
  - out-of-order delivery

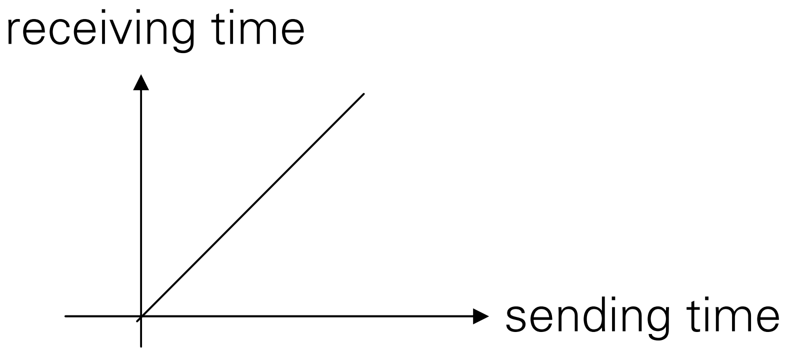
## QoS (2)



- delay (latency):
  - < 150 ms: good qos
    150 – 300 ms: fair
    > 300 ms: poor/bad
  - reasons for delay:
    packetization delay
    + coding delay
    + sending delay (n times)
    + propagation delay (n times)
    + queueing delay (n times)
    + playout buffering delay
    + decoding delay
  - tradeoff:
    packetization delay ↔ header overhead (> 40 bytes)

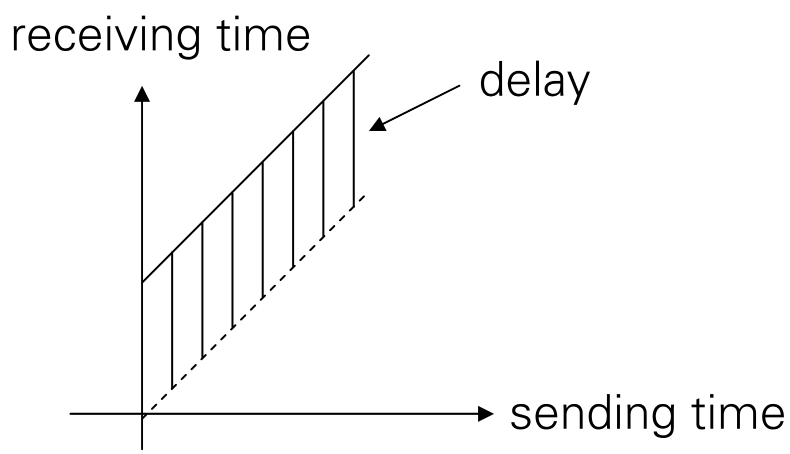
## QoS (3)

- jitter:
  - delay variation
  - optimal situation: no delay (=> no jitter):



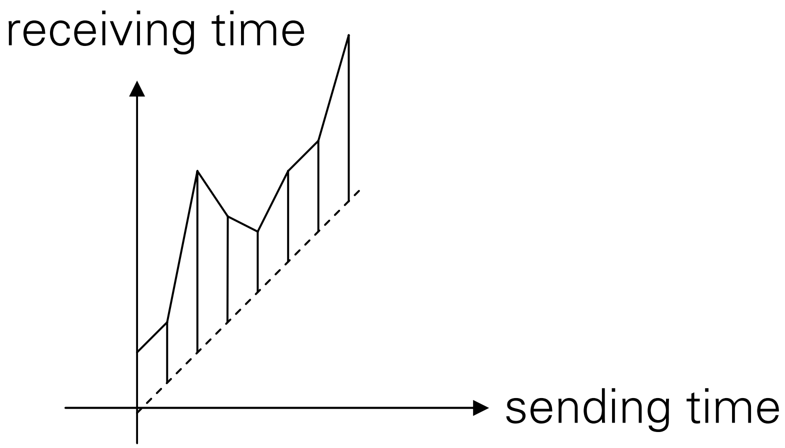
# QoS (4)

- jitter (cont.):
  - constant delay (=> no jitter):



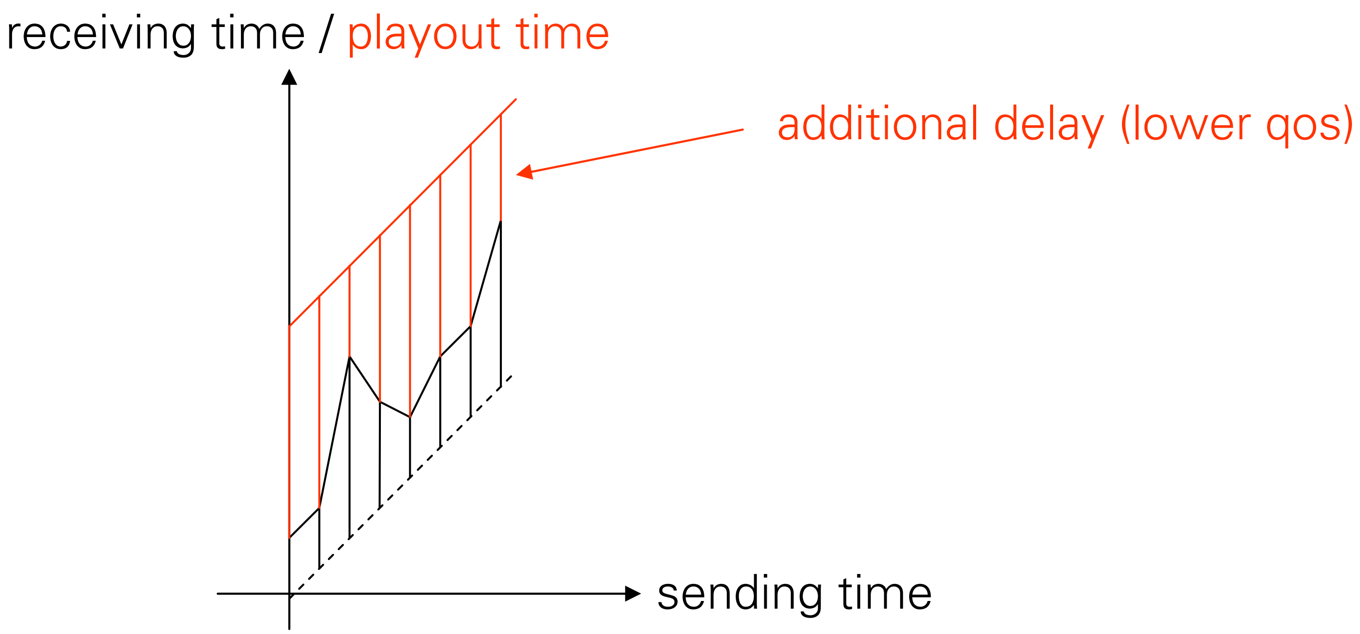
## QoS (5)

- jitter (cont.):
  - variable delay (=> jitter):



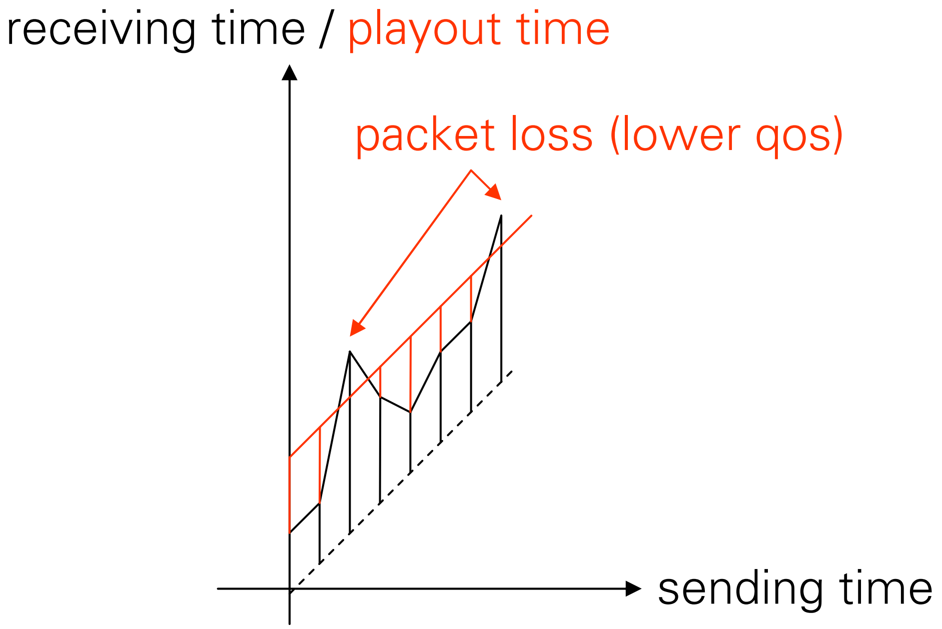
## QoS (6)

- jitter (cont.):
  - jitter can be compensated by playout buffering
  - example: buffering time long



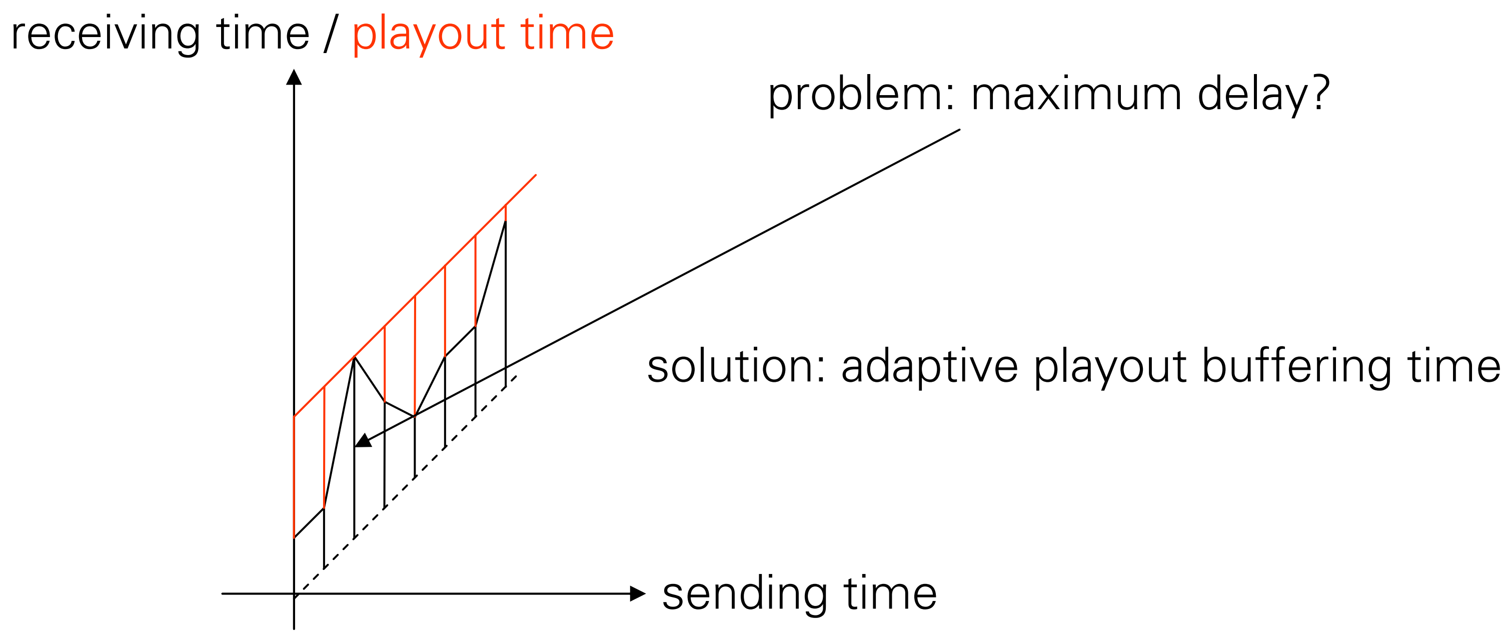
# QoS (7)

- jitter (cont.):
  - jitter can be compensated by playout buffering
  - example: buffering time too short



# QoS (8)

- jitter (cont.):
  - jitter can be compensated by playout buffering
  - example: optimal buffering time



## QoS (9)



- packet loss:
  - rare packet loss can be tolerated (no lower qos)
  - can be compensated by TCP, e.g.
  - but retransmission adds significant delay
    => TCP is not used, but UDP
- out-of-order delivery:
  - compensated by sequence numbering (RTP)

## QoS (10)



- conclusions:
  - moderate delay: no lower qos
  - moderate jitter: compensated by playout buffering
  - moderate packet loss: no lower qos
  - out-of-order delivery: compensated by sequence numbering
- => if the network offers reasonable qos, no problems with VoIP
- otherwise: resource reservation
- approaches:
  - Differentiated Services
  - Integrated Services

## Differentiated Services (1)



- basic idea: SLA (Service Level Agreement) for different traffic types
  - SLA: contract between customer and ISP (Internet Service Provider)
  - SLA contains qos parameters (delay, jitter, loss rate, bandwidth) for different traffic types
  - => long-term resource reservation

## Differentiated Services (2)

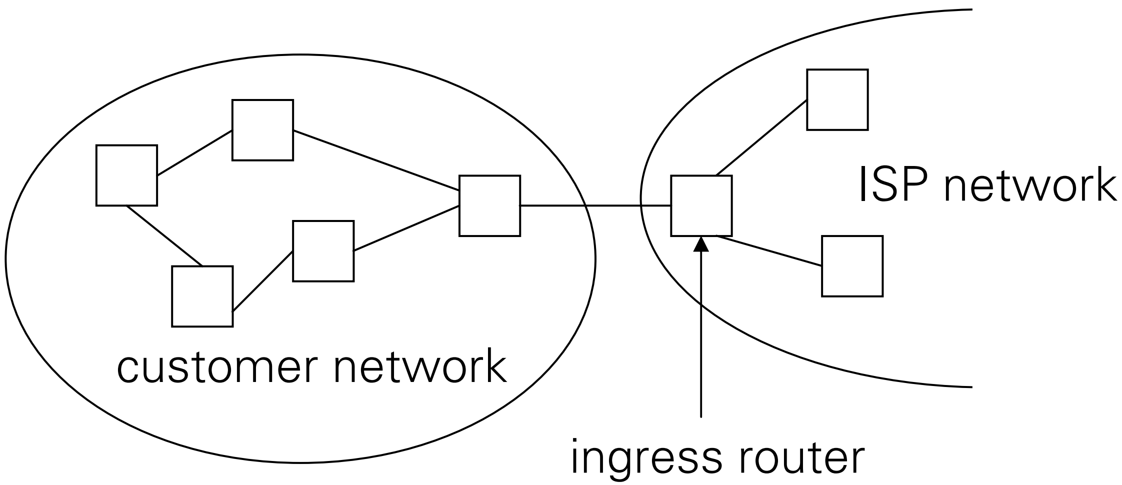
- traffic type marked by special fields in IP packet
  - IPv4: field "Service Type"

<table>
<tr><th>0</th><th>4</th><th>8</th><th>16</th><th>31</th></tr>
<tr><td>Version</td><td>HLen</td><td>Service Type</td><td colspan="2">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td>Fragment Offset</td></tr>
<tr><td colspan="2">Time To Live</td><td>Protocol</td><td colspan="2">Header Checksum</td></tr>
<tr><td colspan="5">Source IP Address</td></tr>
<tr><td colspan="5">Destination IP Address</td></tr>
<tr><td colspan="4">IP Options (if any)</td><td>Padding</td></tr>
<tr><td colspan="5">Data<br>.........</td></tr>
</table>

  - IPv6: field: "Traffic Class"

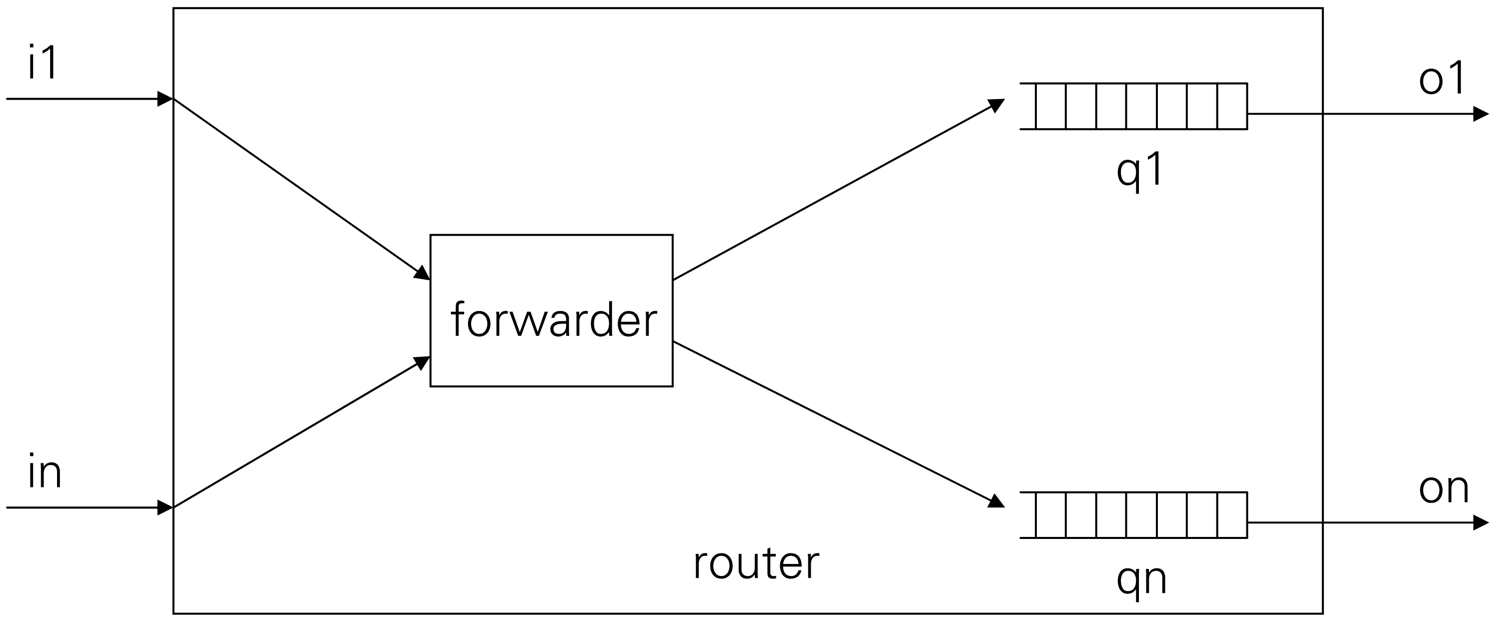
# Differentiated Services (3)

- marking done by sending host or router (customer router or ISP router)
- marking based on IP address / port numbers / transport protocol
- access control at ingress router



## Differentiated Services (4)

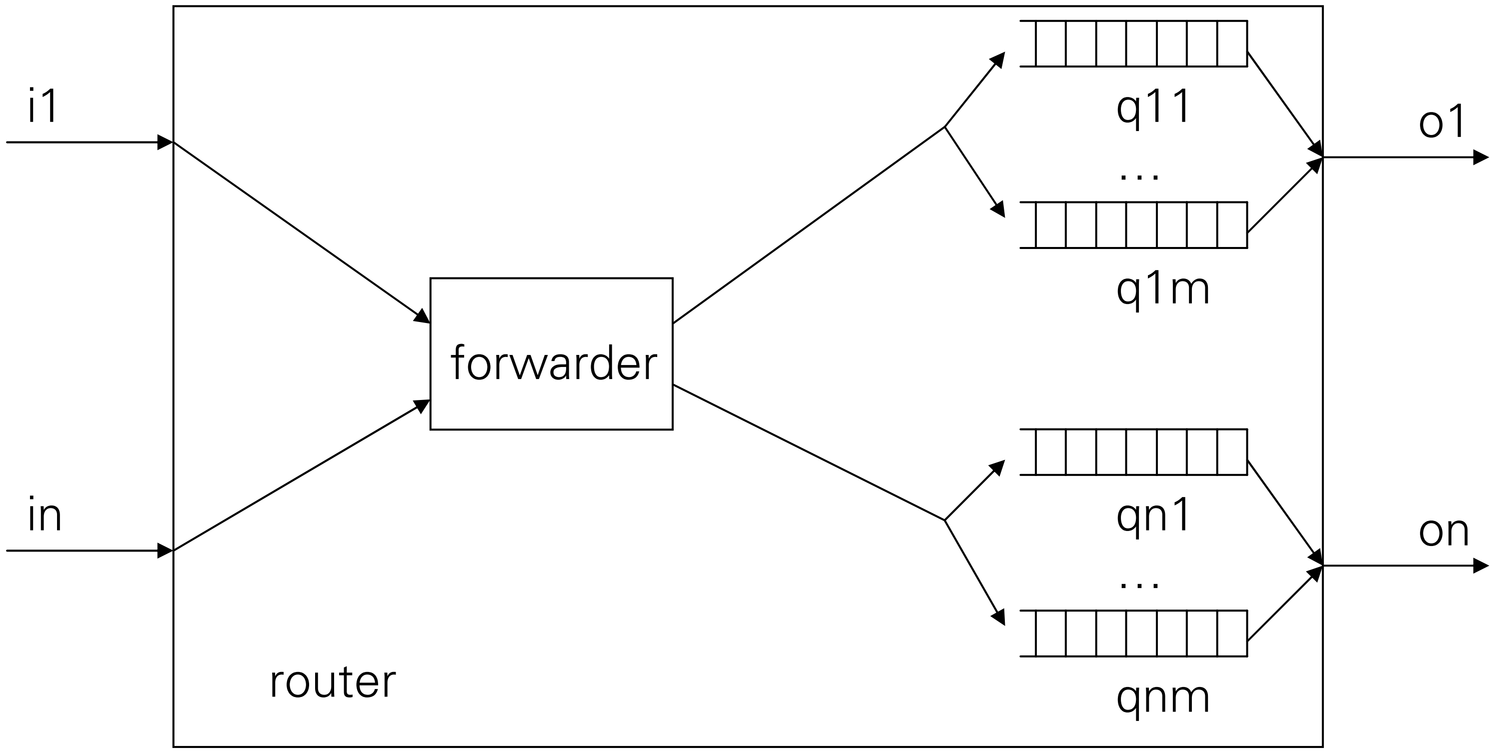
- implementation: "intelligent" queue management
- classical router:




ij – input j
oj – output j
qj – queue j

## Differentiated Services (5)

- now: different queues for different traffic types




ij – input
oj – output
qj - queue

## Differentiated Services (6)

- different scheduling procedures:
  - fair queueing: round robin scheduling
  - priority queueing:
    - qij is only served if all qik (k < j) are empty
    - problem: starvation of lower priority queues

## Differentiated Services (7)

- different scheduling procedures (cont.):
  - custom queueing:
    - also called class-based queueing or weighted round robin
    - for each qij: weight wj
    - may be transformed to a number of bytes or to a number of packets that are taken from each queue in each turn
    - example: wi1 = 0.5<br>wi2 = 0.25<br>wi3 = 0.25

      => two packets from qi1, one packet from qi2, and one packet from qi3 in each turn
  - weighted fair queueing

## Integrated Services (1)

- short-term reservation (call-by-call reservation)
- unidirectional reservation for p2p and multi-point sessions
- reservation is sent from receivers to sender
    - beacuse each receiver knows how much traffic it wants to receive
- => problem: path receiver – sender may be different from path sender – receiver

## Integrated Services (2)



- solution: RSVP (Resource reSerVation Protocol):
  - PATH messages sent from sender to receiver
  - state is maintained in routers
  - RESV messages are forwarded along the route that the PATH messages took before
- each router has to decide how much resouces are still available
- router may reduce the amount of reserved resources
  => the minimum available resources are reserved

## VoIP – Advanced Techniques

- QoS
- <u>Security (ICE)</u>
- Addressing (DNS SRV/NAPTR – ENUM)

## Security (1)

- two aspects:
  - missing security:
    - SPIT (Spam over Internet Telephony)
    - privacy / integrity problems

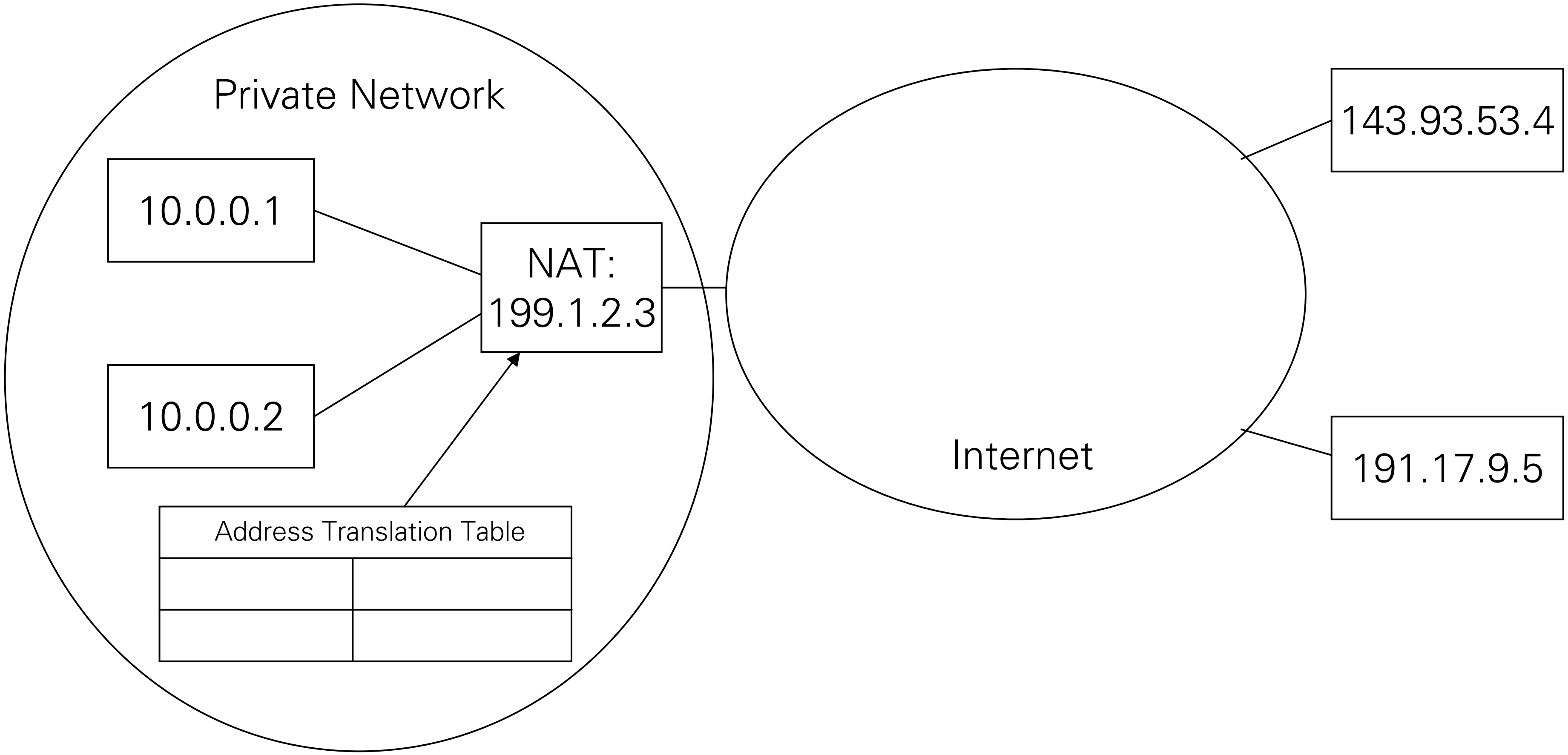
    => same problems as for other Internet applications (like e-mail, e.g.)
    => similar solutions: encryption (SRTP, IPSEC)
  - too much security:
    - no VoIP communication through firewalls and NATs

## Security (2)



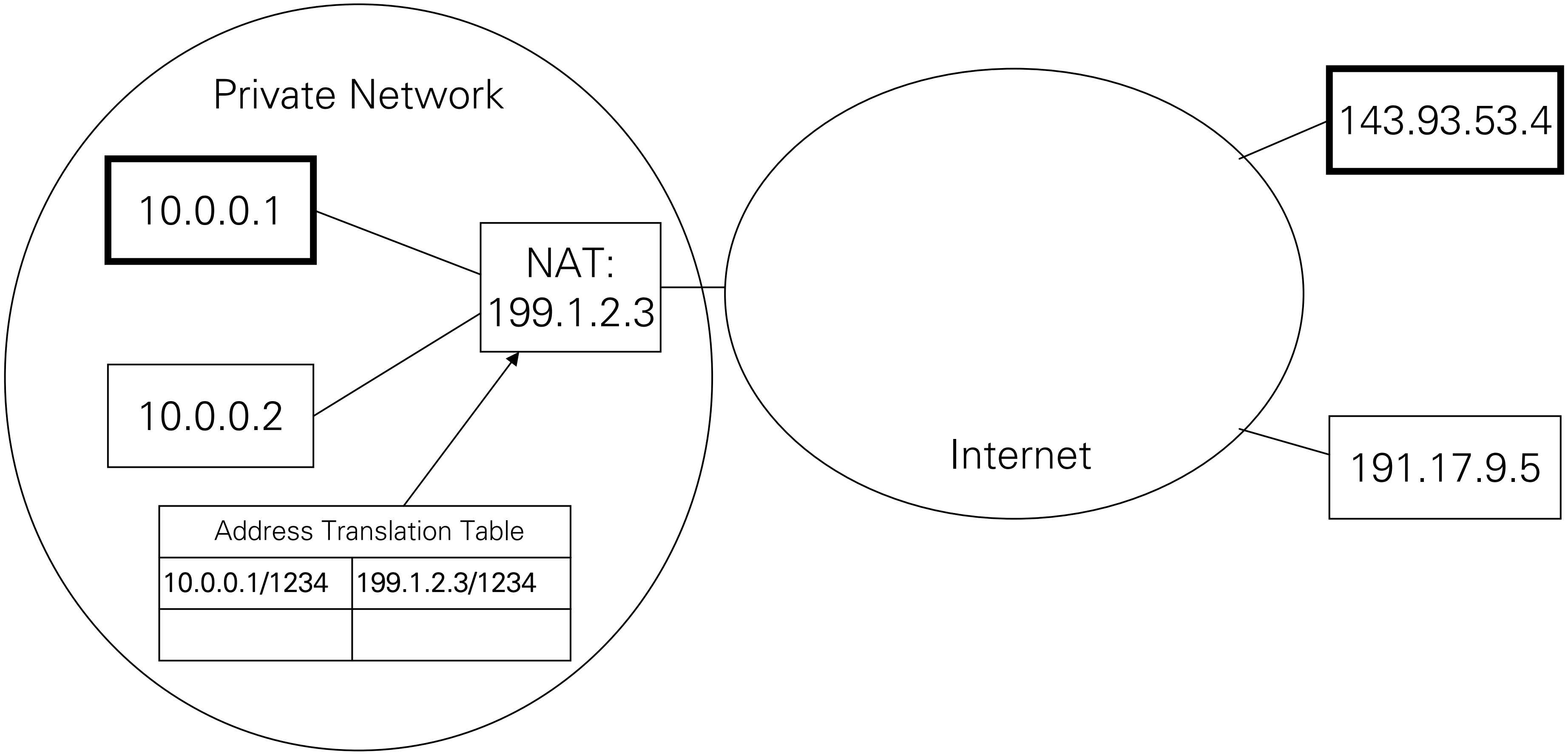
- too much security:
  - IAX2: a single well-known port is used for signaling and data transport
    => port has to be opened in firewall
    => no problems with firewalls and less problems with NATs
  - SIP / H.323: well-known ports for signaling, but ports for data transport are dynamically allocated and exchanged during signaling
    => ICE (uses STUN)

# NAT (1)


Private Network
10.0.0.1
10.0.0.2
NAT:
199.1.2.3
Address Translation Table
Internet
143.93.53.4
191.17.9.5

# NAT (2)

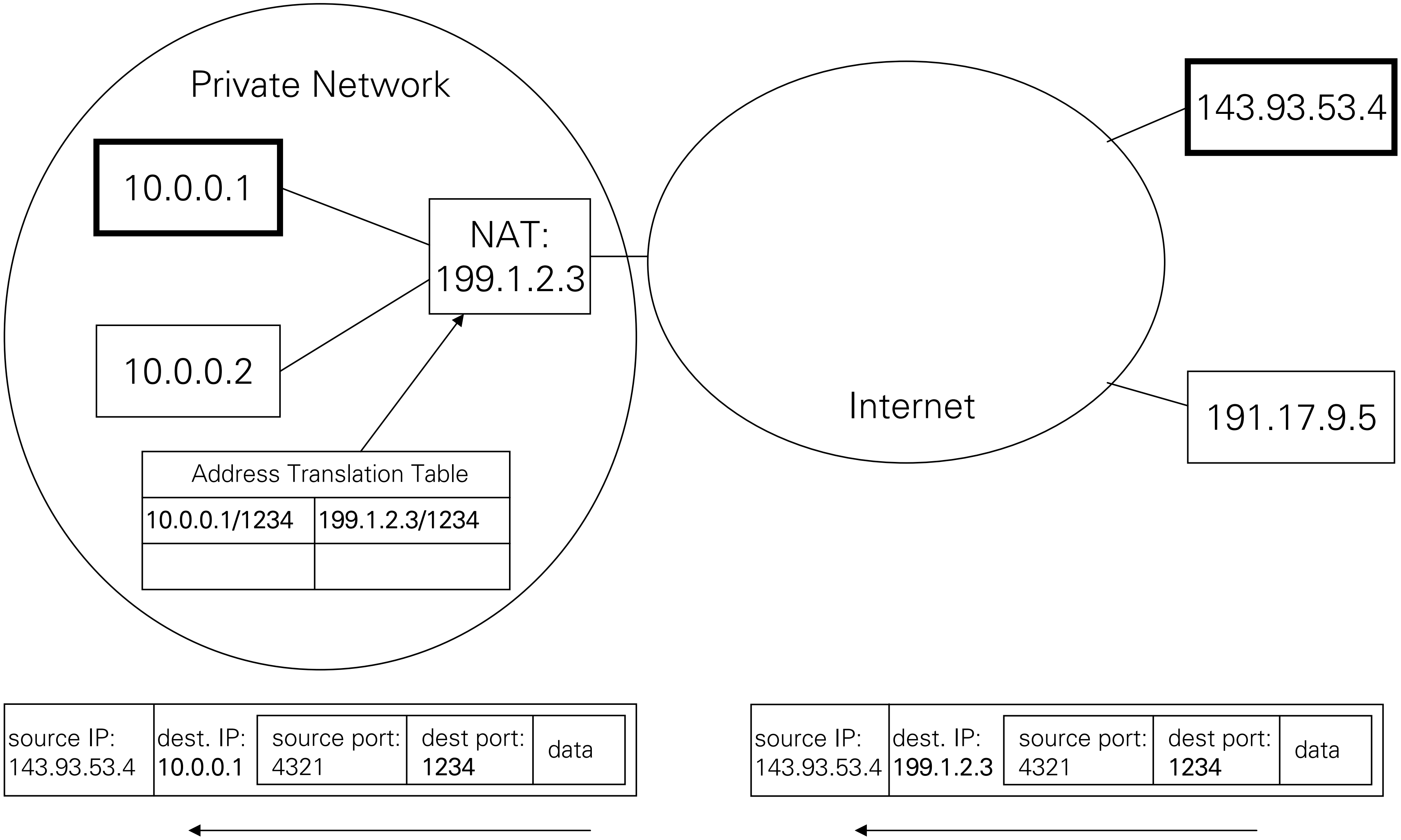




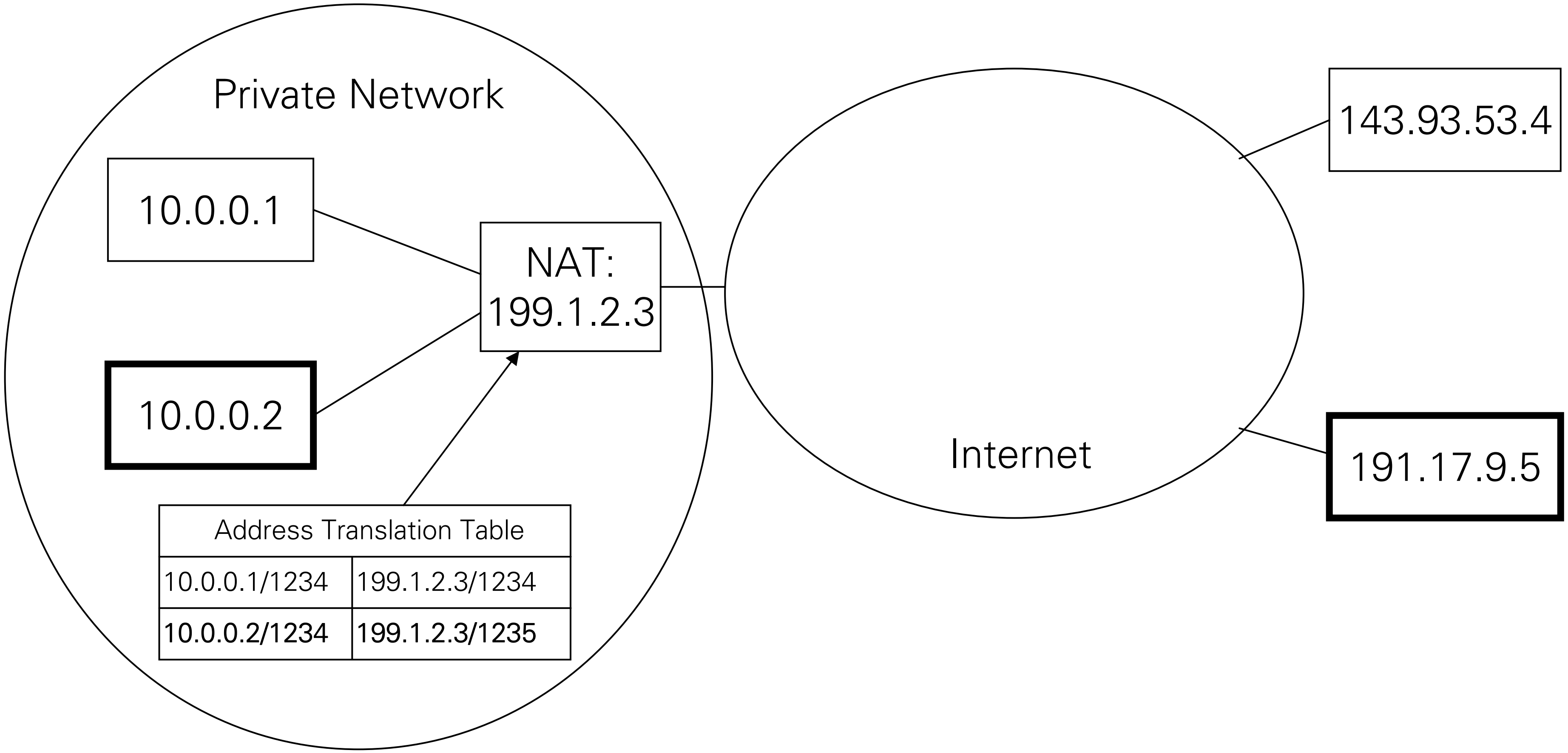

| source IP: | dest. IP: | source port: | dest port: | data |
| --- | --- | --- | --- | --- |
| 10.0.0.1 | 143.93.53.4 | 1234 | 4321 | |

| source IP: | dest. IP: | source port: | dest port: | data |
| --- | --- | --- | --- | --- |
| 199.1.2.3 | 143.93.53.4 | 1234 | 4321 | |

# NAT (3)


Private Network
10.0.0.1
NAT:
199.1.2.3
10.0.0.2
Address Translation Table
10.0.0.1/1234
199.1.2.3/1234
143.93.53.4
Internet
191.17.9.5
source IP:
143.93.53.4
dest. IP:
10.0.0.1
source port:
4321
dest port:
1234
data
source IP:
143.93.53.4
dest. IP:
199.1.2.3
source port:
4321
dest port:
1234
data

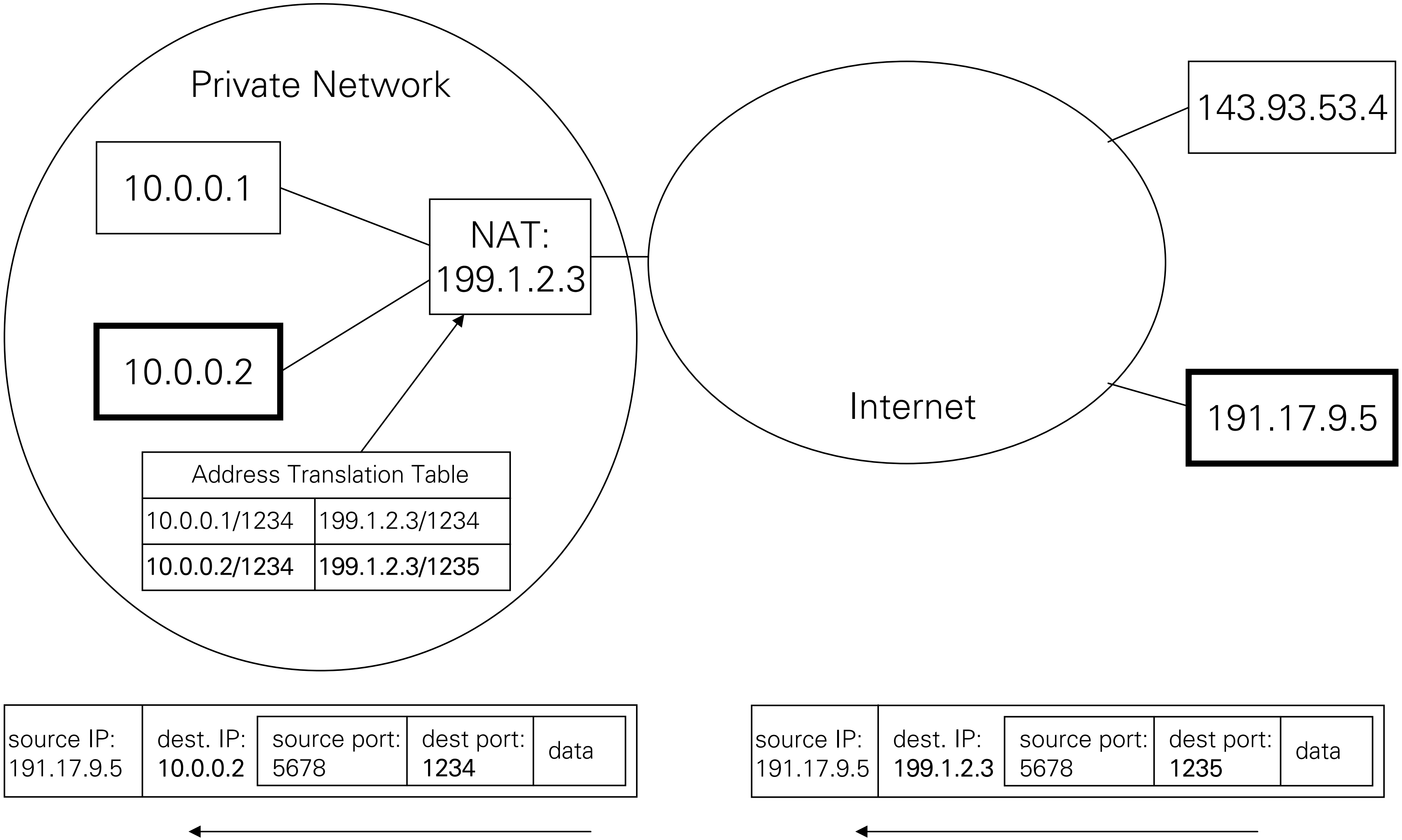
# NAT (4)




| source IP: | dest. IP: | source port: | dest port: | data |
|---|---|---|---|---|
| 10.0.0.2 | 191.17.9.5 | 1234 | 5678 | |

| source IP: | dest. IP: | source port: | dest port: | data |
|---|---|---|---|---|
| 199.1.2.3 | 191.17.9.5 | 1235 | 5678 | |

# NAT (5)


Private Network
10.0.0.1
NAT:
199.1.2.3
10.0.0.2
Address Translation Table
10.0.0.1/1234
199.1.2.3/1234
10.0.0.2/1234
199.1.2.3/1235
Internet
143.93.53.4
191.17.9.5
source IP:
191.17.9.5
dest. IP:
10.0.0.2
source port:
5678
dest port:
1234
data
source IP:
191.17.9.5
dest. IP:
199.1.2.3
source port:
5678
dest port:
1235
data

## NAT (6)

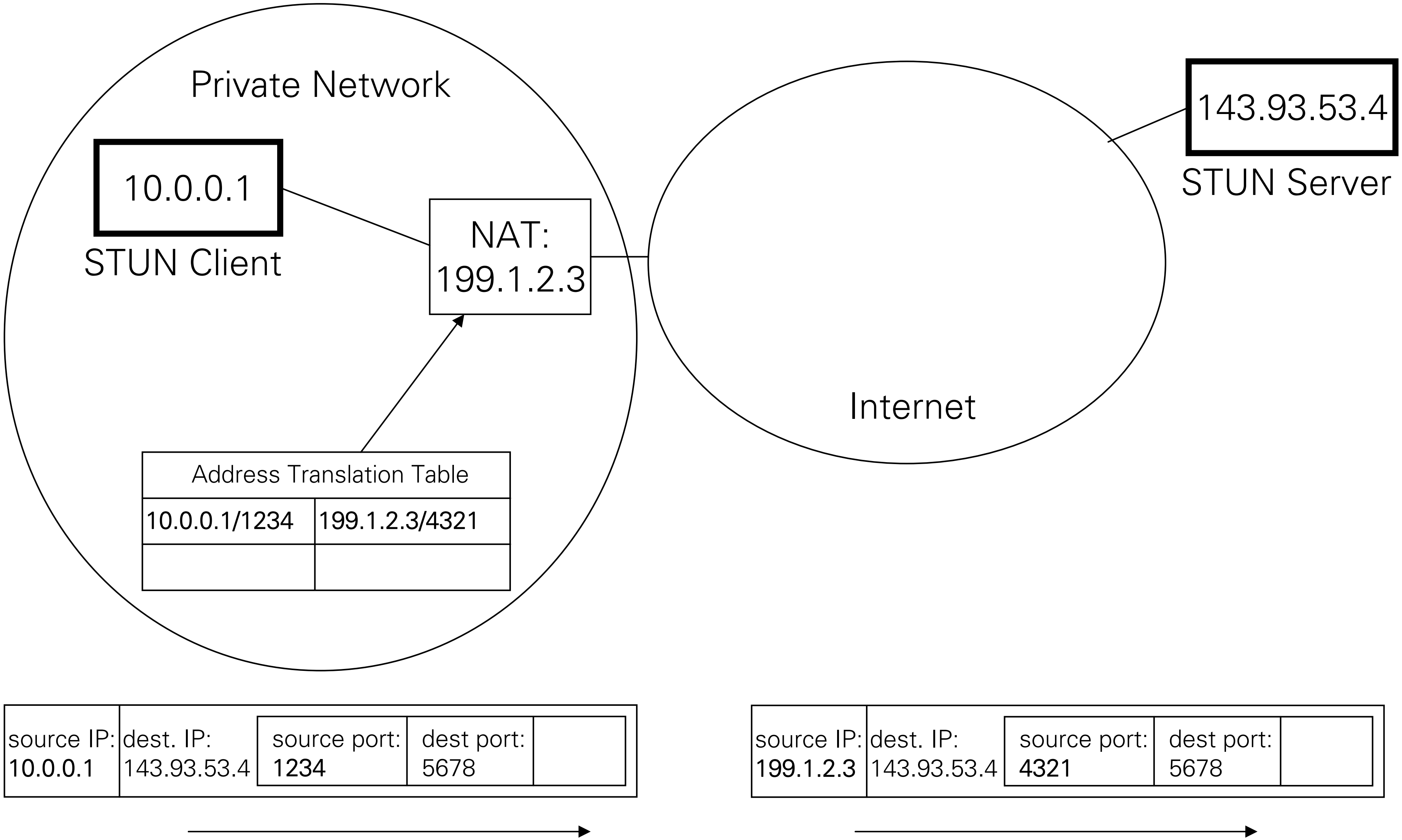


- conclusion:
  NAT means: address **and port** translation

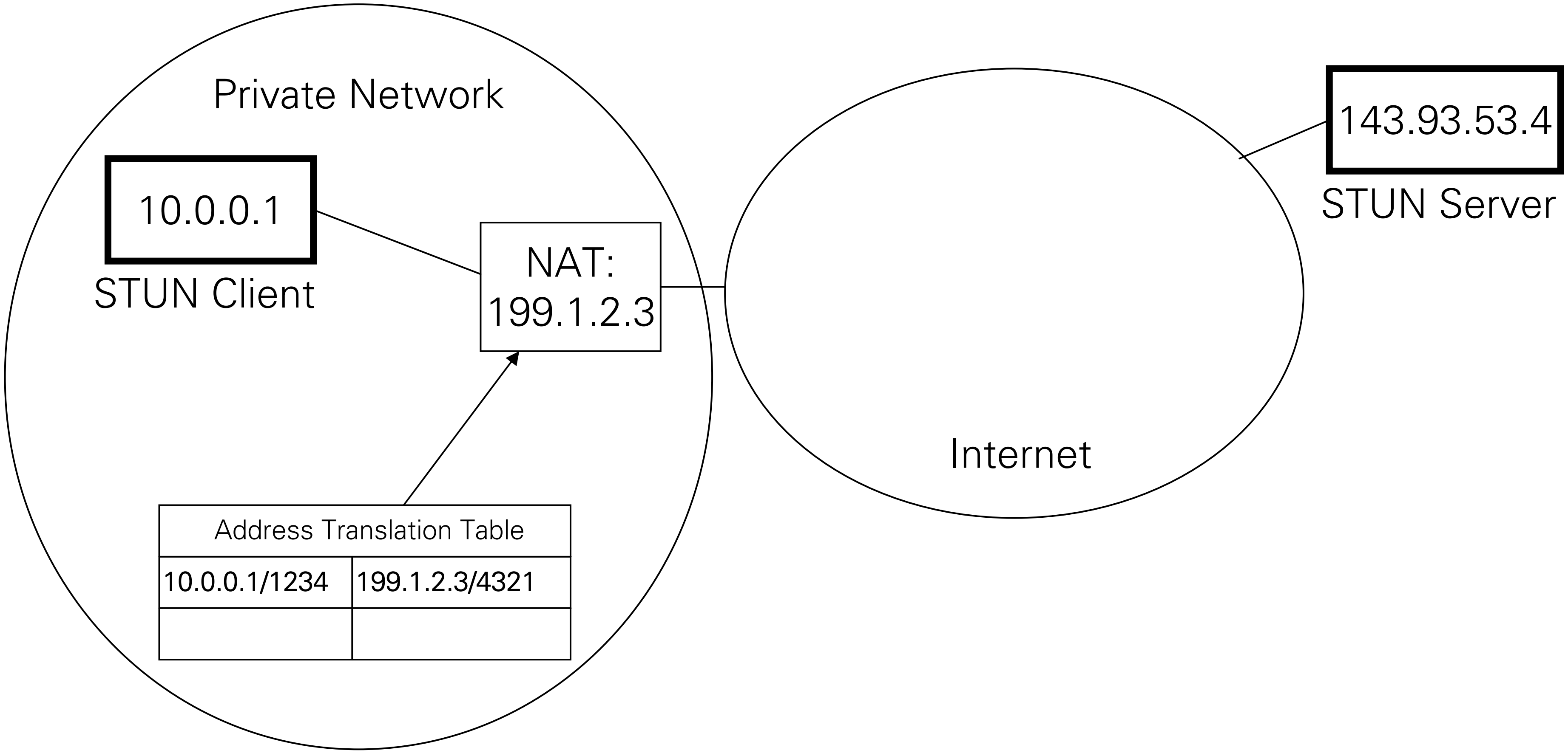
## STUN (1)

- Simple Traversal of UDP through Network Address Translators (NATS), RFC 3489
- basic idea (simplified):
  - STUN client sends a request to a STUN server (like PING)
  - STUN server sends back a response (PONG) with the source address and source port number that were seen in the STUN request

# STUN (2)


Private Network
10.0.0.1
STUN Client
NAT:
199.1.2.3
Address Translation Table
10.0.0.1/1234
199.1.2.3/4321
Internet
143.93.53.4
STUN Server
source IP: 10.0.0.1
dest. IP: 143.93.53.4
source port: 1234
dest port: 5678
source IP: 199.1.2.3
dest. IP: 143.93.53.4
source port: 4321
dest port: 5678

# STUN (3)




| source IP: | dest. IP: | source port: | dest port: | 199.1.2.3 |
|---|---|---|---|---|
| 143.93.53.4 | **10.0.0.1** | 5678 | **1234** | 4321 |

| source IP: | dest. IP: | source port: | dest port: | 199.1.2.3 |
|---|---|---|---|---|
| 143.93.53.4 | **199.1.2.3** | 5678 | **4321** | 4321 |

# ICE (1)



- Interactive Connectivity Establishment:
  methodology for NAT traversal for SIP, Internet Draft
- makes use of existing protocols, especially:
  - STUN
- requires no changes to NATs, but some additional SDP attributes

## ICE (2)

- overview:
  - SIP client 1 obtains as many contact IP address / port combinations as it can
    - SIP client 1 may use STUN, TURN, RSIP for this purpose
    - possible: just use the local address / port combinations
  - a STUN server has to be active for each IP address / port combination on the SIP client's computer
  - each IP address / port combination is placed into a an m-line of the SDP message (as alternatives) sent to SIP client 2
    - with priority values

## ICE (3)



- overview (cont.):
  - SIP client 2 sends STUN requests to each alternative IP address / port combination
    - to check for connectivity
    - to gather its own contact IP address / port combinations
  - SIP client 2 sends an answer to the SIP request with its own IP address / port combinations
  - a STUN server has to be active for each IP address / port combination on this SIP client‘s computer, too

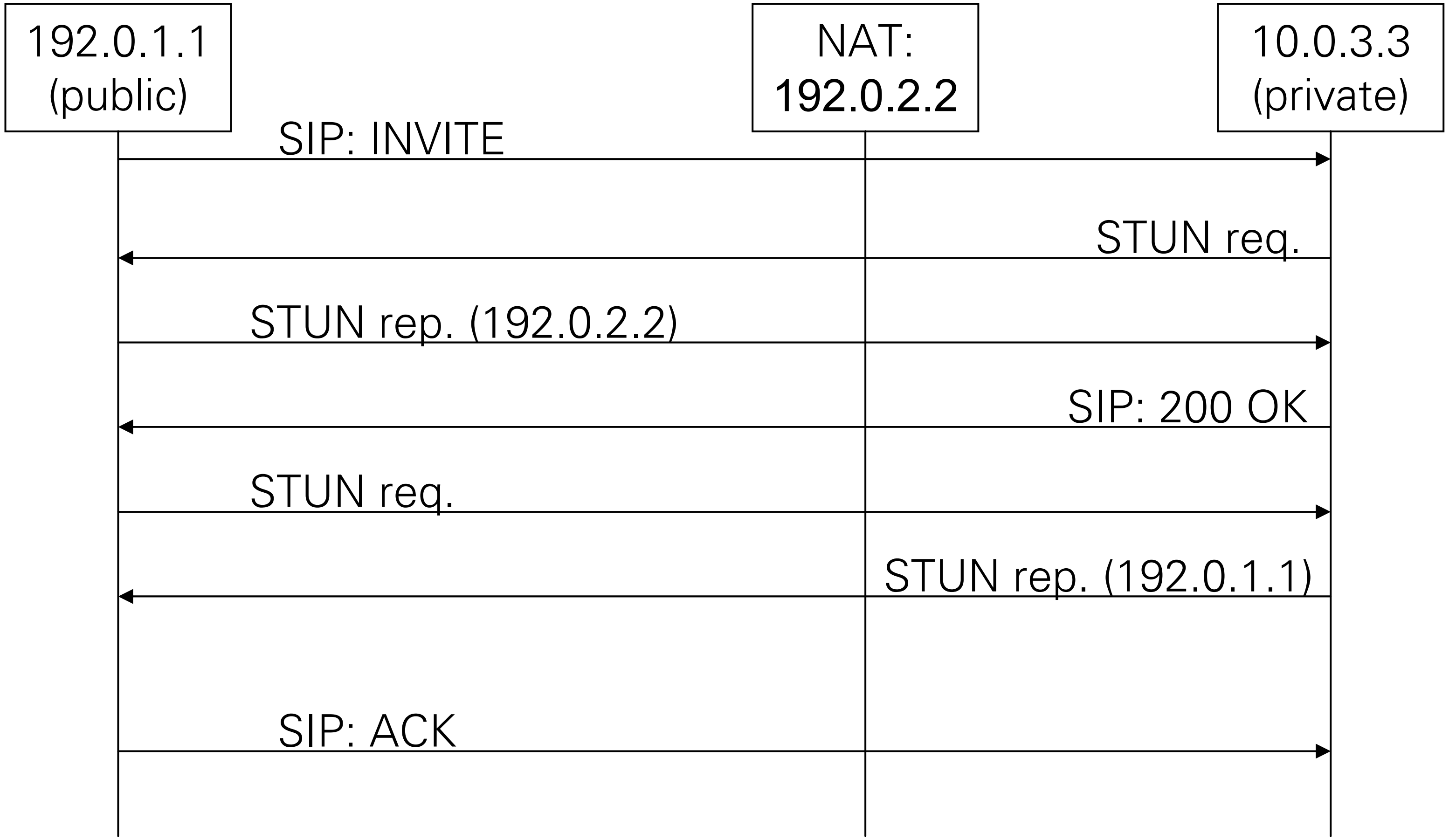
# ICE (4)



- overview (cont.):
  - when SIP client 1 sends STUN requests to each alternative IP address / port combination to check for connectivity, it may discover new own IP address / port combinations
  - SIP clients may update the transmitted address / port combinations later by SIP UPDATE messages

## ICE (5)

- example:
  - pc2 is reachable through NAT via SIP



## ICE (6)



- example (cont.):
  - INVITE contains local address / port combination of the computer on the left
  - computer on the right learns from STUN reply:
    1. connectivity to other side (on the left): okay
    2. a new own contact address / port combination
  - SIP response (okay) contains new contact address / port combination
  - computer on the left learns from STUN reply:
    1. connectivity to other side (on the right): okay
    2. no new own contact address / port combination

# ICE (7)

- example (cont.):

```
INVITE sip:B@example.com SIP/2.0
Content-Length: ....
Content-Type: application/sdp

v=0
o=alice 2890844730 2890844731 IN IP4 host.example.com
s=
c=IN IP4 192.0.1.1
t=0 0
m=audio 54344 RTP/AVP 0
a=mid:1
a=qvalue:1.0
a=stun:user 9kksj==
```

# ICE (8)

- example (cont.):

```
SIP/2.0 200 OK
Content-Length: ....
Content-Type: application/sdp

v=0
o=bob 280744730 28977631 IN IP4 host2.example.com
s=
c=IN IP4 192.0.2.2
t=0 0
m=audio 6886 RTP/AVP 0
a=mid:1
a=qvalue:1.0
a=stun:user asd8866
a=derived:1
```

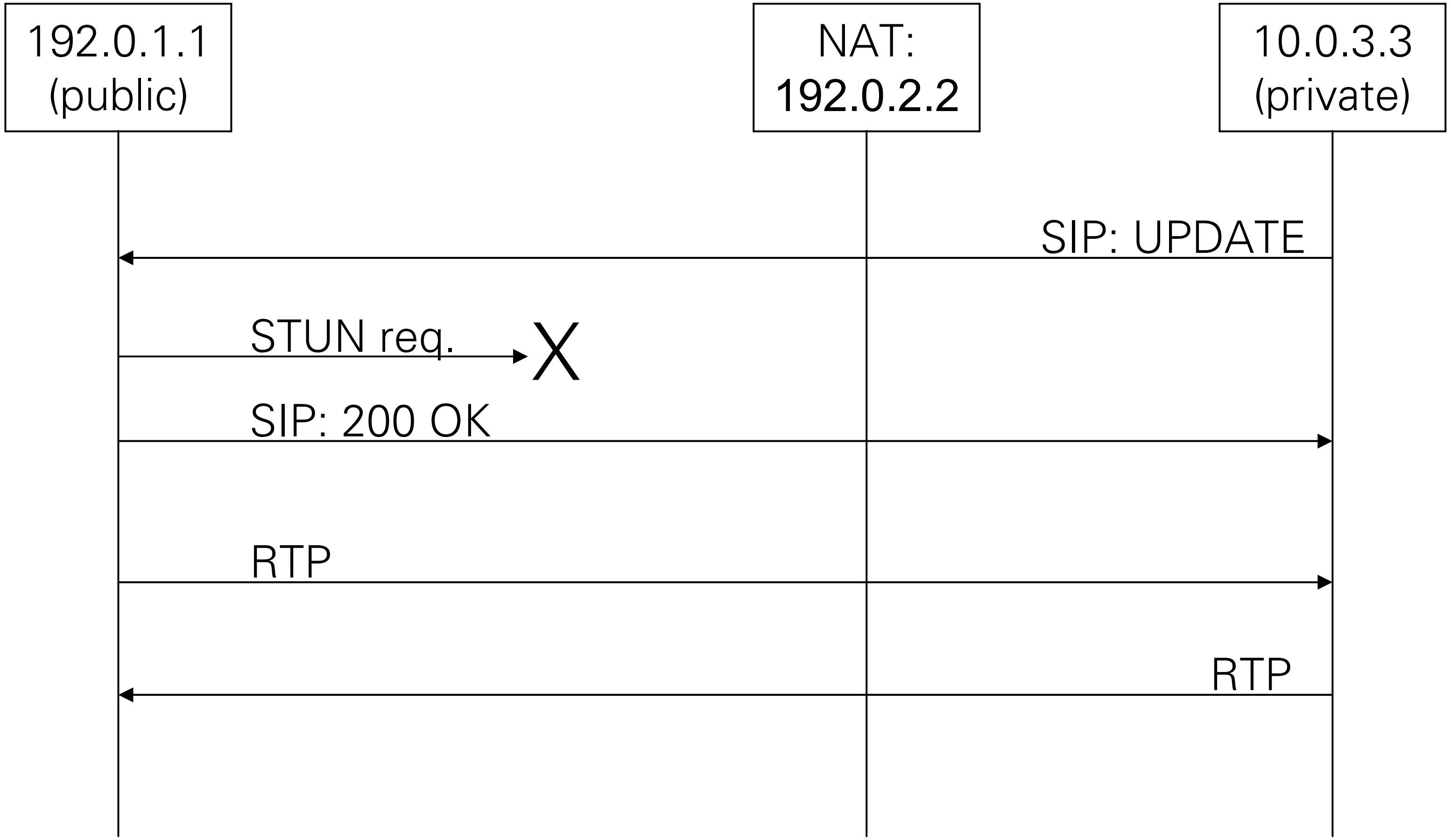
## ICE (9)



- example (cont.):
  - computer on the right sends UPDATE message for its local address / port combination

# ICE (10)

- example (cont.):



# ICE (11)

- example (cont.):

```
UPDATE sip:A@host.example.com
Content-Length: ....
Content-Type: application/sdp

v=0
o=bob 280744730 28977631 IN IP4 host2.example.com
s=
t=0 0
a=group:ALT 1 2
m=audio 6886 RTP/AVP 0
c=IN IP4 192.0.2.2
a=mid:1
a=qvalue:1.0
a=stun:user asd8866
a=derived:1
m=audio 22334 RTP/AVP 0
c=IN IP4 10.0.3.3
a=mid:2
```

## VoIP – Advanced Techniques

- QoS
- Security (ICE)
- Addressing (DNS SRV/NAPTR – ENUM)

## Addressing (1)

- optimal solution: use of DNS (as for other applications like e-mail and WWW)
- DNS: Domain Name Service:
  - hierarchical name structure (order: bottom-to-top) (example: www.fh-trier.de)
  - mapping of different kinds of information:
    - A: host names => IP addresses (`morrison.fh-trier.de 43200 IN A 143.93.53.136`)
    - MX: mailbox names => mail servers (host names)
    - PTR (inverse A): IP addresses => host names
    - NS: domain names => DNS servers (host names)

## Addressing (2)

- simplest form:
  sip:oechsle@sipserver.fh-trier.de
  => lookup sipserver.fh-trier.de as A record
- in e-mail, e.g., mail addresses do not depend on mail server name
  => smarter addressing needed:
- two possible addressing schemes:
  - e-mail like addresses:
    example: mailto:oechsle@fh-trier.de => sip:oechsle@fh-trier.de
    => DNS SRV (and NAPTR)
  - telephone numbers:
    => ENUM
- current problem: different VoIP systems / providers use different addressing schemes

# DNS SRV

- RFC 2782
- DNS SRV records:
  - usage here: mapping SIP address => SIP server
  - general form:
    `_service._proto.name TTL class SRV priority weight port target`
  - example:
    `_sip._udp.fh-trier.de. 6000 IN SRV 10 20 5060 sip.fh-trier.de.`
  - ttl: caching time (here: 100 minutes = 100 * 60 = 6000 seconds)
    priority: lower values => higher priorities: query order of SIP servers
    weight: load balancing factor for equal priority entries
  - DNS SRV may be used by VoIP applications without any other support, because the lookup name can be deduced from the SIP address, if the service and protocol are known
    => otherwise: use NAPTR

# DNS NAPTR

- DNS NAPTR records:
  - NAPTR: Naming Authority Pointer
  - mapping domain => SRV record, for contacting a server with the specific protocol
  - general form:
    `domain-name TTL class NAPTR order pref flags srv regexp target`
  - example:
    `fh-trier.de. 6000 IN NAPTR 60 50 ”s” ”SIP+D2U” ”” _sip._udp.fh-trier.de.`
  - ttl: caching time (here: 100 minutes = 100 * 60 = 6000 seconds)
    priority: lower values => higher priorities: query order of SIP servers
    weight: load balancing factor for equal priority entries
  - NAPTR may be used with DNS SRV, or without
    - srv: ”s” => pointer to SRV entry
      ”u” => pointer to URI (example: ”sip:xyz@abc.de”)

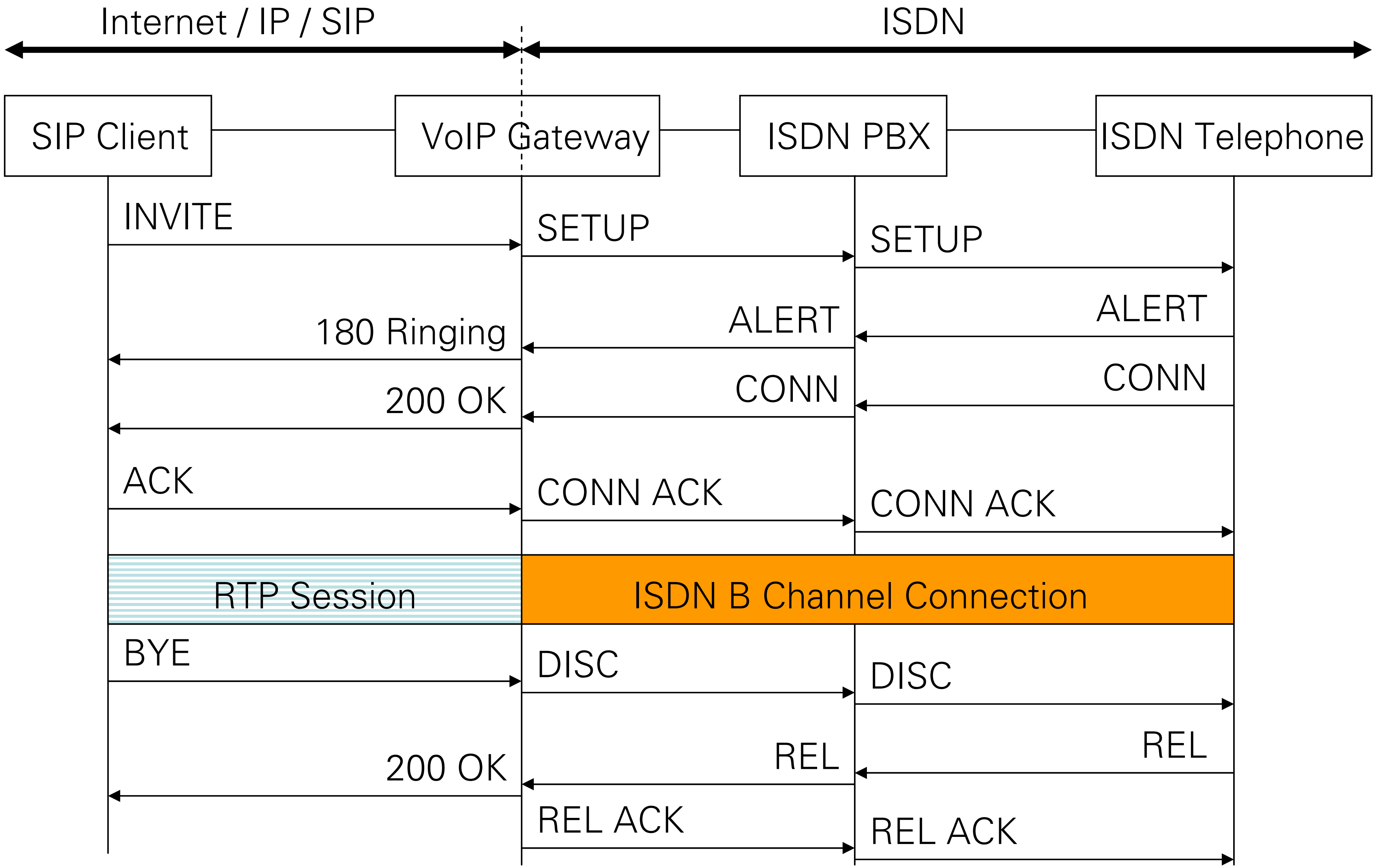
# ENUM



- Electronic NUMbering or
  tElephone NUmber Mapping, RFC 3761
- mapping: telephone number (E.164 number) => SIP server name, mobile phone number, e-mail address, WWW URL, ...
  => one number for all kinds of contact information
- same idea as for PTR records: mapping IP addresses => host names
  - lookup 143.93.53.136
    => lookup 136.53.93.143.in-addr.arpa
  - additional domain name hierarchy for IP addresses
- ENUM:
  - lookup +49/651/8103508
    => lookup 8.0.5.3.0.1.8.1.5.6.9.4.e164.arpa
- type of DNS resource records: NAPTR

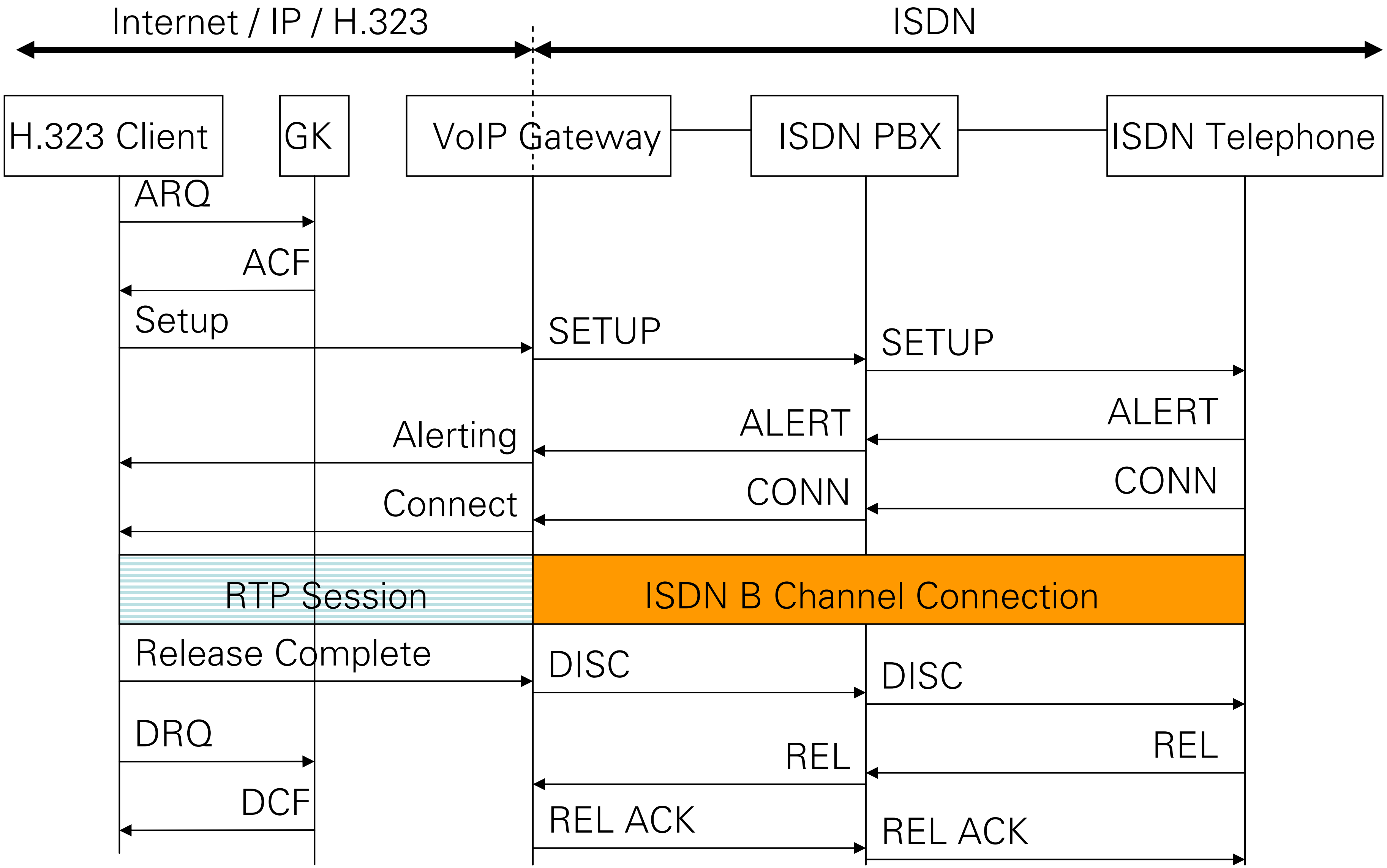
## Gateways between ISDN and VoIP Networks(1)





# Gateways between ISDN and VoIP Networks(2)

FACHHOCHSCHULE TRIER
Hochschule für Technik, Wirtschaft und Gestaltung
University of Applied Sciences
Informatik



## VoIP Hardware / Software (1)



- clients:
  - IP telephones:
    - telephones with Ethernet / WLAN interface
    - ATA (Analog Telephone Adapter):
      "box" with Ethernet / WLAN interface and telephone socket
  - soft phones (PC / notebook + client software) used with
    - microphone / loud speaker
    - head set
    - telephone (USB interface)

## VoIP Hardware / Software (2)



- servers:
  - SIP, H.323, IAX server
  - location server
  - billing server
  - gateways

## Business Aspects (1)

- Why VoIP?
  - for users: cheaper
  - for companies: single net infrastructure
  - for providers: single net infrastructure
    - Deutsche Telekom (German Telecom):
      2012: PSTN will be totally replaced by IP net
    - USA: 30 % of telephone traffic is VoIP
  - new VoIP providers: freenet, gmx, nikotel. sipgate, web.de

## Business Aspects (2)

- Why VoIP? (cont.)
  - new applications:
    - click-to-dial feature: link on a web page: click => call setup
    - push-to-talk (walkie-talkie)
    - unified messaging (telephony, e-mail, sms, chat, ...)
    - universal reachability
    - exchange of additional data:
      - audio and video conferences
      - shared applications

# Summary



- PSTN: Classical Circuit-Switched Telephony
- VoIP: Packet-Switched Telephony – Basic Techniques
- VoIP: Advanced Techniques
- Gateways between ISDN and VoIP Networks
- VoIP Hardware / Software
- Business Aspects

# Outlook

- Asterisk: "soft PBX":
  - SIP / H.323 / IAX2 client
  - SIP / H.323 / IAX2 server (including call forwarding, ...)
  - location server
  - gateway (between SIP, H.323, IAX2, and PSTN)
  - codec translator
  - can be programmed as voice mail server, ...