Technologie-Zentrum Informatik

SIP Tutorial

Upperside SIP 2003 **Paris, 14 January 2003**

Dr.-Ing. Jörg Ott *{sip,mailto}:jo@tzi.org*

Technologie-Zentrum Informatik

Center for Computing Technology
Universität Bremen, Germany

- 10 professors, ~ 100 researchers, lots of students
- The 5 sectors of the TZI:
  - BV Image processing
  - BISS Dependable systems
  - ISI Software ergonomics and information management
  - IS Intelligent systems
  - DMN Digital Media and Networks

www.tzi.org

TZI Digital Media and Networks

Architectures, Protocols and Interfaces for computer-based communication und collaboration

- **Digital Networks (infrastructure)**
  - Internet technologies
  - Application: synchronous distributed groupware systems
  - Special interests: teleconferencing and Internet Telephony
- **Digital Media (content)**
  - Structured document communication (XML/SGML technologies)
  - Special interests: document transformation
- **IETF / ITU-T standardization**

ipDialog™

SIPTone IP telephone

- SIP endpoint
  - Speakerphone
  - Supports autoconfiguration
  - Built-in Ethernet bridge
  - Power over Ethernet
- Supplementary services
  - 3-way-calls, transfer, call hold, call waiting
  - message waiting, call forwarding, call return, …
- SIP URLs and E.164 numbers
  - Internal phonebook (web-based)
- Business feature set in progress
  - simple software updates
- Customization
  - different cases, displays, keypads
  - more sophisticated functionality
  - Web-based configuration

This Tutorial…

- Proposed Schedule:

1000 – 1110	Multimedia over IP, RTP, SIP Intro
1110 – 1130	Morning Break
1130 – 1230	SIP: Basics, Call Flows
1230 – 1330	Lunch Break
1330 – 1500	Security, SIP Services
1500 – 1530	Afternoon Break
1530 – 1700	Telephony, Deployment, 3GPP

- We know that there is nothing like a 20min coffee break...
  But please try anyway...

Tutorial Overview

- Internet Multimedia Conferencing Architecture

You are here

- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP

IP Multimedia Applications (1)

- Packet multimedia experiments since 1980s
  - A/V tools + protocols for A/V over IP
  - Conference control protocols

<u>Internet broadcasting (Mbone)</u>

- First IETF Audiocast (1992)

- Broadcasts of IETF WG sessions
  - audio + video + whiteboard (transparencies)
  - enables remote participation (even talks)

- Broadcasting special events
  - talks, concerts, NASA shuttle missions, ...

- Broadcasting “radio” and “television” programs
  - numerous channels available today

IP Multimedia Applications (2)

<u>Teleconferences</u>

- Traditional Internet focus: large groups
- Small groups supported as well

- Audio + video + data (whiteboards, editors, ...)
- Multimedia gaming sessions

- Examples:
  - seminars and lectures
  - project meetings
  - work group meetings between IETFs

- Gatewaying where needed (PSTN, ISDN, cellular, …)

IP Multimedia Applications (3)

IP Telephony

- "Special case" of teleconferences
  - point-to-point + conference calls
- Gatewaying to PSTN / ISDN / GSM
  - also H.323, MGCP, MEGACO
- Include "Supplementary Services"
  - what users are used to from their touch tone phone or PBX environment
- Include "Intelligent Network (IN)" services
  - To some degree trivial in an IP environment

IP Multimedia Applications (4)

Multimedia retrieval services

- "Video on demand"-style
  - including "VCR controls": pause/restart/cue/review
- Access to multimedia clips from web browsers
  - Commercial examples: RealAudio/RealVideo, IP/TV, Microsoft
- Often: Internet- / web-based access to live streams
  - "Big Brother", concerts, etc.
- Option: recording multimedia information

Common Requirements

Network infrastructure

- Multicast routing
- Real-time-capable packet forwarding
- Resource reservation

Transport protocols

- Real-time (audio / video) information
- Non-real-time (data / control) information

Media encoding standards

Security

Specific requirements

Control protocols

- Setup / teardown of communication relationships
- Conference control
- Remote control of devices (e.g. media sources)

Naming and addressing infrastructure

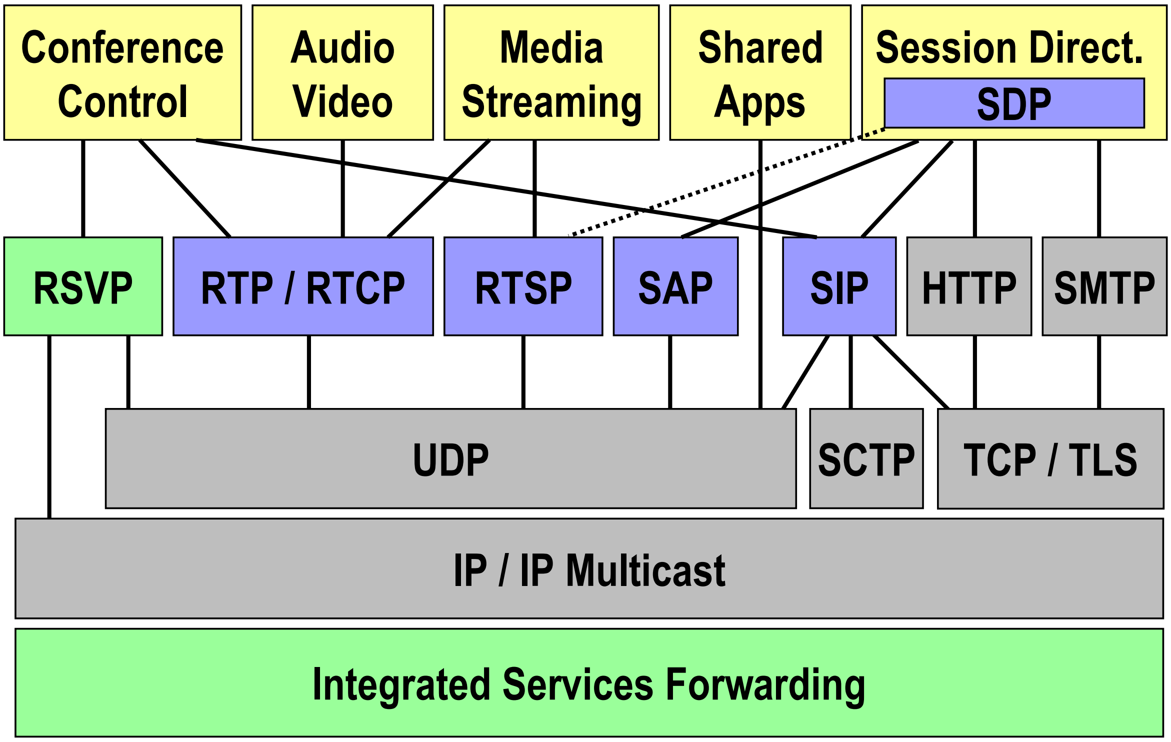
User (and service) location

Billing and accounting (and policing)

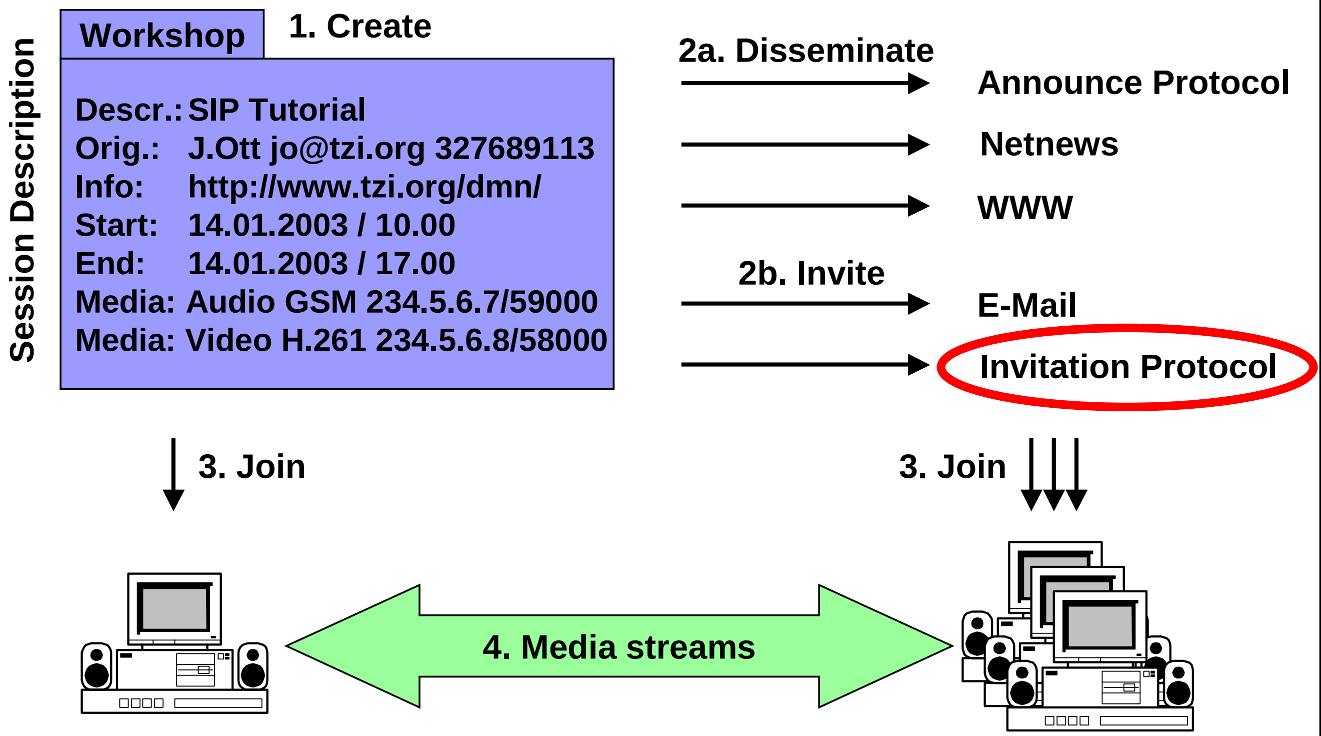
(Legal requirements)

Controversial!

Conferencing Architecture



Conference Establishment



Conference Description

Session Description Protocol (SDP, RFC 2327)

- All you need to know about a session to join
  - who? — convener of the session + contact information
  - what about? — name and informal subject description
  - when? — date and time, rep
  - where? — multicast addresses, port numbers
  - which media? — capability requirements
  - how much? — required bandwidth
  - …
- Grouped into three categories
  - 1 x session, m x time, n x media

Conference Discovery

Session Announcement Protocol (SAP)

- Advertise conference description regularly
  - announcement frequency depends on distribution scope
- Originally: avoid conflicts in multicast addresses
  - simple multicast address allocation (224.2.0.0/16)
- Optional security support
  - authentication header and encrypted payload

Alternatives

- Use conventional means
  - e-mail (SMTP), web servers (HTTP), Netnews (NNTP)
  - MIME type defined for SDP specification: “application/sdp”
- Session Invitation Protocol (SIP)

Tutorial Overview

- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport

You are here

- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP

Real-time Media over Packets

- Audio / Video are continuous media
- Packet networks transport discrete units
  - digitize media
  - compression
  - packetization
- No additional multiplex (beyond UDP/IP) needed:
  - no separate lines, bit allocations, etc.
  - transport different media in different packets
  - can give different quality of service to different media
  - allows different sites to receive different subsets

Sources of Delay

- **Sender**
  - Capturing / digitizing delay (+ operating system)
  - Encoding / compression delay
  - Packetization delay
- **Network (potentially highly variable!)**
  - Link propagation delay (order of speed of light)
  - Serialization delay
  - Queuing delay
- **Receiver**
  - buffering delay + potential delay for repair
  - decoding / decompression delay
  - rendering / replay delay (+ operating system)

Real-time Media over Packets

Little help needed from transport protocol:

- **Retransmission would take too long (interactivity!)**

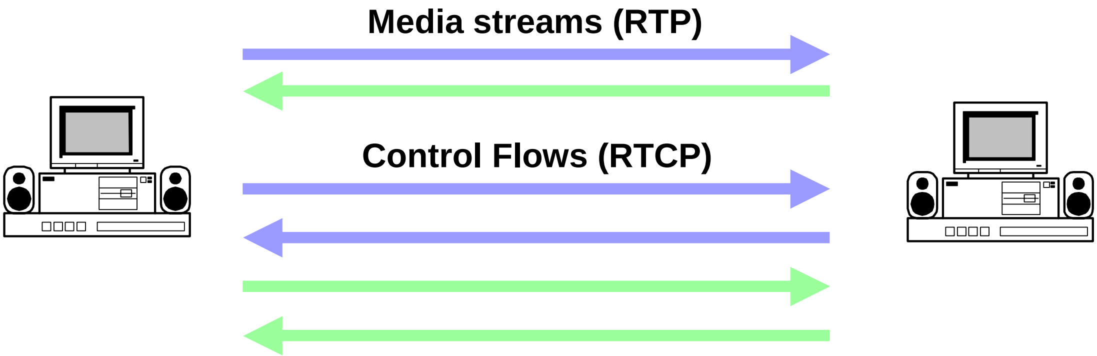
End systems must buffer before playout!

- **Jitter in transmission delay due to queueing**
- **Packet A/V rule #1:**
  - jitter is never a problem,
  - worst-case delay is!
- **Need a timestamp in packet to be able to play at right time**
  - intra-stream timing
  - optionally correlate for inter-stream timing (e.g. lip-sync)

Real-time Transport Protocol (1)

RTP Functionality (RFC1889)

- framing for audio/video information streams
- preserve intra- and inter-stream timing
- mechanisms for awareness of others in a conference
- ➲ RTP sessions



Real-time Transport Protocol (2)

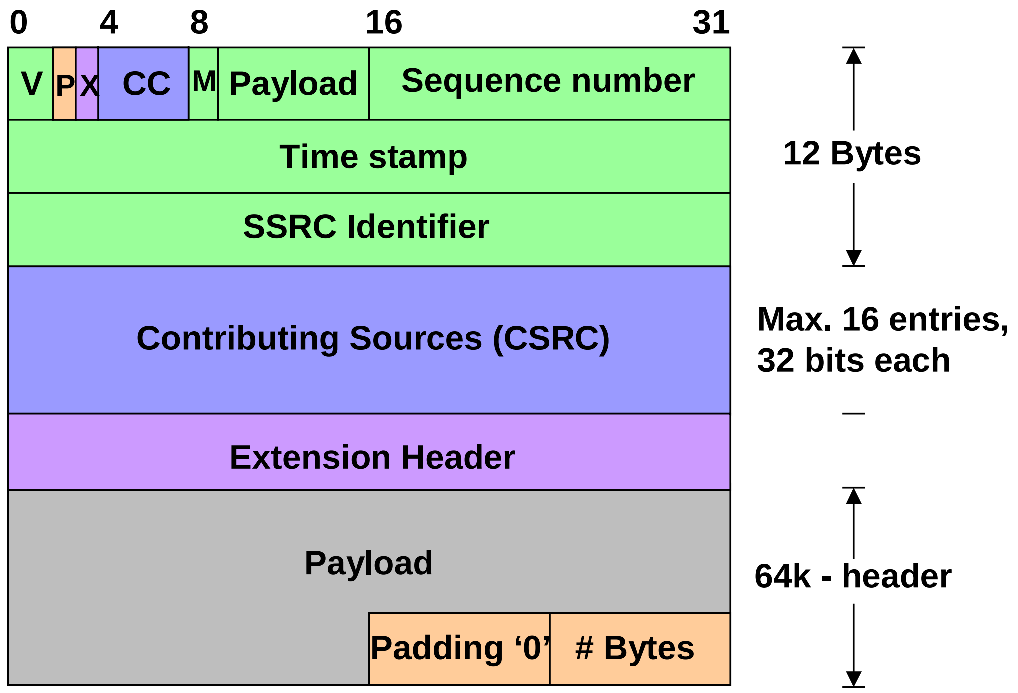
Standard RTP packet header

- independent of *payload type*
- possibly seconded by *payload header*

Mechanisms

- detect packet loss, cope with reordering
  - sequence number per media stream
- determine variations in transmission delays
  - media specific time stamp (e.g., 8 kHz for PCM audio)
  - allows receiver to adapt playout point for continuous replay
- source identification
  - possibly mixed from several sources
- payload type identifier

RTP Header



RTP Header Fields (1)

V: Version — version 2 defined in RFC 1889

P: Padding — indicates padding
bytes indicated in last byte

X: eXtension bit — extension header is present

Extension header — single additional header (TLV coded)

CC: CSRC count — # of contributing sources

CSRC: contributing sources —
which sources have been “mixed”
to produce this packet’s contents

RTP Header Fields (2)

M: Marker bit — marks semantical boundaries in media stream (e.g. talk spurt)

Payload type — indicates packet content type

Sequence # — of the packet in the media stream (strictly monotonically increasing)

Timestamp — indicates the instant when the packet contents was sampled (measured to media-specific clock)

SSRC: synchronization source — identification of packet originator

Real-time Transport Control Protocol

Mechanisms:

- Receivers constantly measure transmission quality
  - delay, jitter, packet loss
- Regular control information exchange between senders and receivers
  - feedback to sender (receiver report)
  - feed forward to recipients (sender report)
- Allows applications to adapt to current QoS
- Overhead limited to a small fraction (default: 5 % max.) of total bandwidth per RTP session
  - members estimate number of participants
  - adapt their own transmission rate

Obtaining sufficient capacity: outside of RTP!

RTP Payload Types

- 7-bit payload type identifier
  - some numbers statically assigned
  - dynamic payload types identifiers for extensions – mapping to be defined outside of RTP (control protocol, e.g. SDP "a=rtpmap:")

<u>Payload formats defined for many audio/video encodings</u>

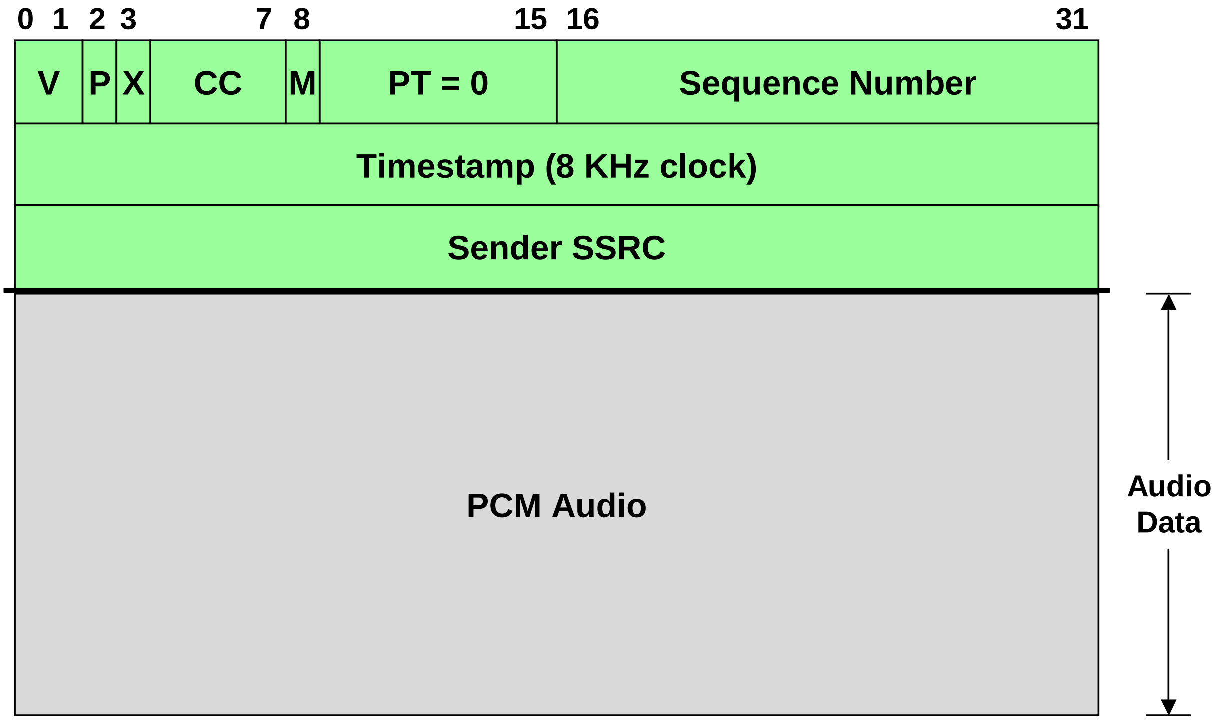
- Conferencing profile document RFC 1890
  - audio: G.711, G.722, G.723.1, G.728, GSM, CD, DVI, …
- In codec-specific RFCs
  - audio: Redundant Audio, MP-3, ...
  - video: JPEG, H.261, MPEG-1, MPEG-2, H.263, H.263+, BT.656
  - others: DTMF, text, SONET, ...
- Generic formats
  - generic FEC, (multiplexing)

Media Packetization Schemes (1)

<u>General principle:</u>

- Payload specific additional header (if needed)
- Followed by media data
  - packetized and formatted in a well-defined way
  - trivial ones specified in RFC 1890
  - RFC 2029, 2032, 2035, 2038, 2190, 2198, 2250, 2343, 2429, RFC 2431, RFC 2435, 2658, 2733, 2793, 2833, 2862, and many Internet Drafts
  - Guidelines for writing packet formats: RFC 2736
- Functionality
  - enable transmission across a packet network
  - allow for semantics-based fragmentation
  - provide additional information to simplify processing and decoding at the recipient
  - maximize possibility of independent decoding of individual packets
  - Typically, little to do for audio!

Audio over RTP: PCM



Video over RTP: H.261

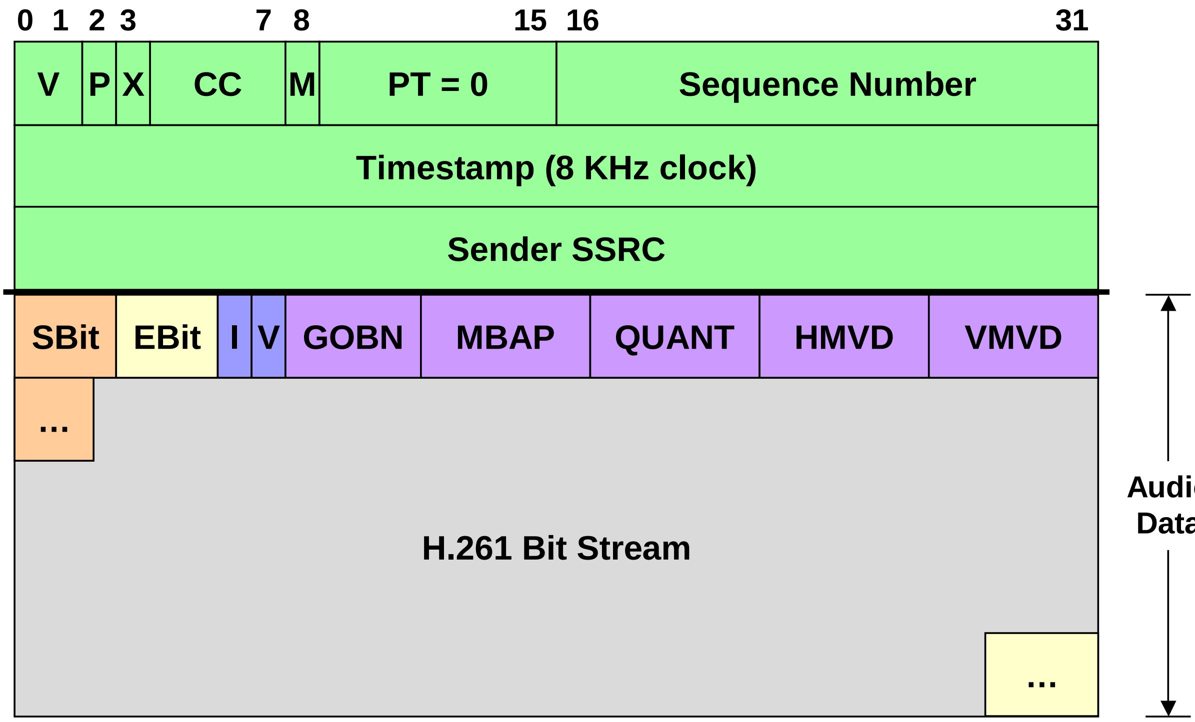
Additional payload-specific header preceeds payload

- To avoid expensive bit shifting operations
  - Indicate # invalid bits in first (SBit) and last (EBit) octet of payload
- Indicate Intra encoding (I bit)
- Indicate the presence of motion vector data (V bit)
- Carry further H.261 header information to enable decoding in the presence of packet losses

Further mechanisms for video conferencing

- FIR: Full Intra Request
  - Ask sender to send a full intra encoded picture
- NACK: Negative Acknowledgement
  - Indicate specific packet loss to sender

Video over RTP: H.261 (2)



Media Packetization Schemes (2)

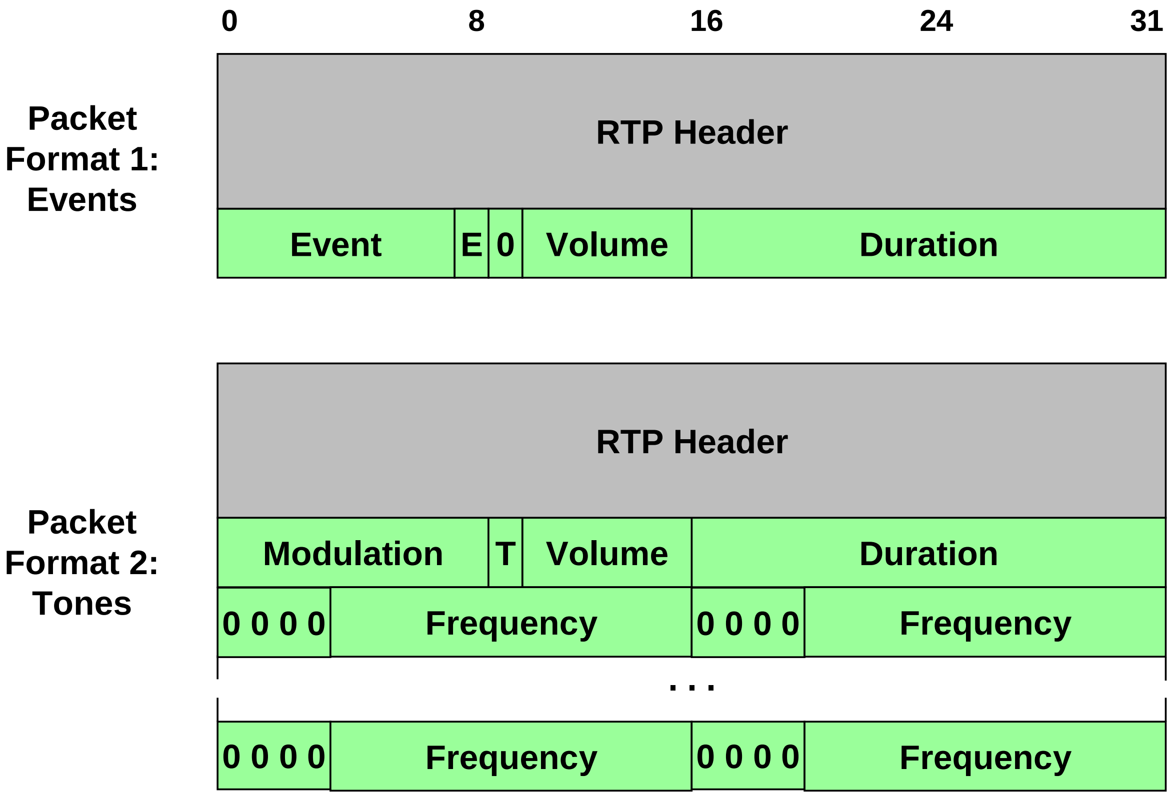
Error-resilience for real-time media

- Input: Observation on packet loss characteristics
- Generic mechanisms (RFC 2354)
  - Retransmissions
    - in special cases only (interactivity!)
  - Interleaving
  - Forward Error Correction (FEC)
    - media-dependent vs. media-independent
    - Generic FEC: RFC 2733
- Feedback loops for senders
  - based upon generic and specific RTCP messages
  - adapt transmission rate, coding scheme, error control, ...

DTMF over RTP (1)

- **DTMF digits, telephony tones, and telephony signals**
  - two payload formats
  - 8 kHz clock by default
  - audio redundancy coding for reliability
- **Format 1: reference pre-defined events**
  - 0 - 9 * # A - D Flash [17]
  - modem and fax tones [18]
  - telephony signals and line events [43]
    - dial tones, busy, ringing, congestion, on/off hook, …
  - trunk events [44]
  - specified through identifier (8-bit value), volume, duration
- **Format 2: specify tones by frequency**
  - one, two, or three frequencies
  - addition, modulation
  - on/off periods, duration
  - specified through modulation, n x frequency, volume

DTMF over RTP (2)



Tutorial Overview

- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP

You are here

SIP: Session Initiation Protocol

From HTTP and Session Invitation to Setup and Control for Packet-based Multimedia Conferencing

History of Mbone conference initiation

Session Invitation Protocol

(Handley/Schooler)

– Participant location
– Conference invitation
– Capability negotiation during setup

Simple Conference Invitation Protocol

(Schulzrinne)

– Participant location
– Conference invitation
– Capability negotiation during setup
– Changing conference parameters
– Terminate/leave conference

1996 → ***Session Initiation Protocol***

Session Initiation Protocol (SIP)

First draft in December 1996

- Joint effort to merge SIP and SCIP
- IETF WG MMUSIC
  (Multiparty Multimedia Session Control)

Application-layer call signaling protocol:

- Creation, modification, termination of teleconferences
- Negotiation of used media configuration
- Re-negotiation during session
- User location → personal mobility
- Security
- Supplementary services

RFC 3261
– June 2002
– obsoletes RFC 2543

Timeline: 1996

Initial Internet Drafts:
Session Invitation Protocol (SIP) – M. Handley, E. Schooler
Simple Conference Invitation Protocol (SCIP) – H. Schulzrinne

22 Feb 1996

SIP: Setup + Caps Negotiation

SCIP: Setup + Caps Modify + Terminate

4-8 Mar 1996

Presentations at 35th IETF, Los Angeles

2 Dec 1996

Merged Draft: SIP -01

Main Features set:
TCP/UDP, Forking, Redirection, addrs
INVITE,CAPABILITY
From: To: Path:

Timeline: 1997

27 Mar 97

Draft SIP -02

Formal syntax
CAPABILITY → OPTIONS
Path: → Via:
Ideas for Alternates:

31 Jul 97

Draft SIP -03

SIP URL: sip://jo@…
CONNECTED, BYE, REGISTER
Call-ID: Sequence:
Allow: Expires:

11 Nov 97

Draft SIP -04

CONNECTED → ACK
UNREGISTER
Sequence: →CSeq:
Call-Disposition:
Require:

Dec 97

IETF Action: Split SIP into base spec and extensions

Timeline: 1998

SIP -05 (14 May)
- CANCEL
- UNREGISTER ➔ ∅
- URL sip://jo ➔ sip:jo
- Record-Route:
- IANA assignments
- Security Cons. Sect.

SIP -06 (17 Jun)

SIP -07 (16 Jul)

SIP -08 (8 Aug)
- Clarifications & fixes
- Cleaning up the spec
- Call-ID: MUST
- tag parameter

SIP -09 (18 Sep)
- Call Hold SDP

IETF Action: Last Call for Proposed (28 Sep)

14 May | 17 Jun | 16 Jul | 8 Aug | 18 Sep | 28 Sep

TZi Technologie-Zentrum Informatik
Universität Bremen

Timeline: 1998/99

SIP -10 (12 Nov 98)
- No more DNS MX
- URI: RFC 2396

SIP -11 (15 Dec 98)
- Update on SDP part

SIP -12 (15 Jan 99)
- DNS Lookup
- Tidying up

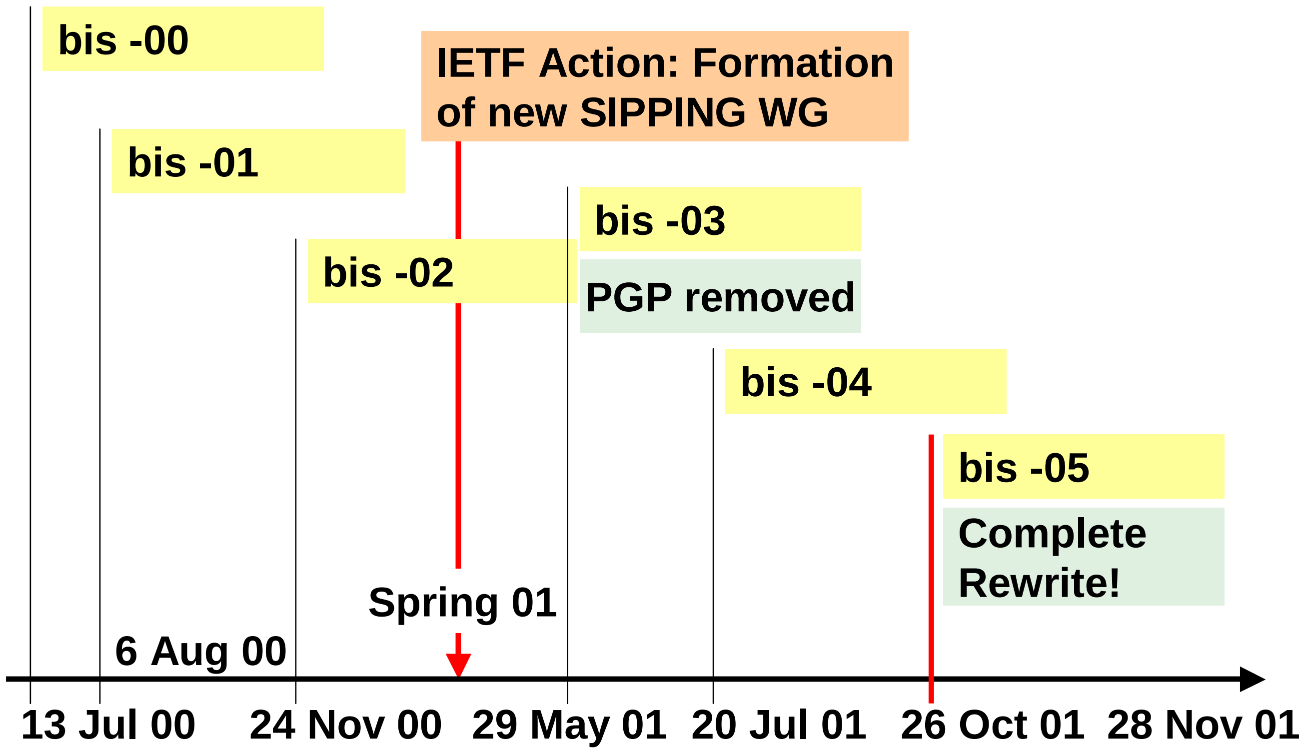
IETF Action: Approval for Proposed Standard (2 Feb 99)

IETF Action: Published as RFC 2543 (17 Mar 99)

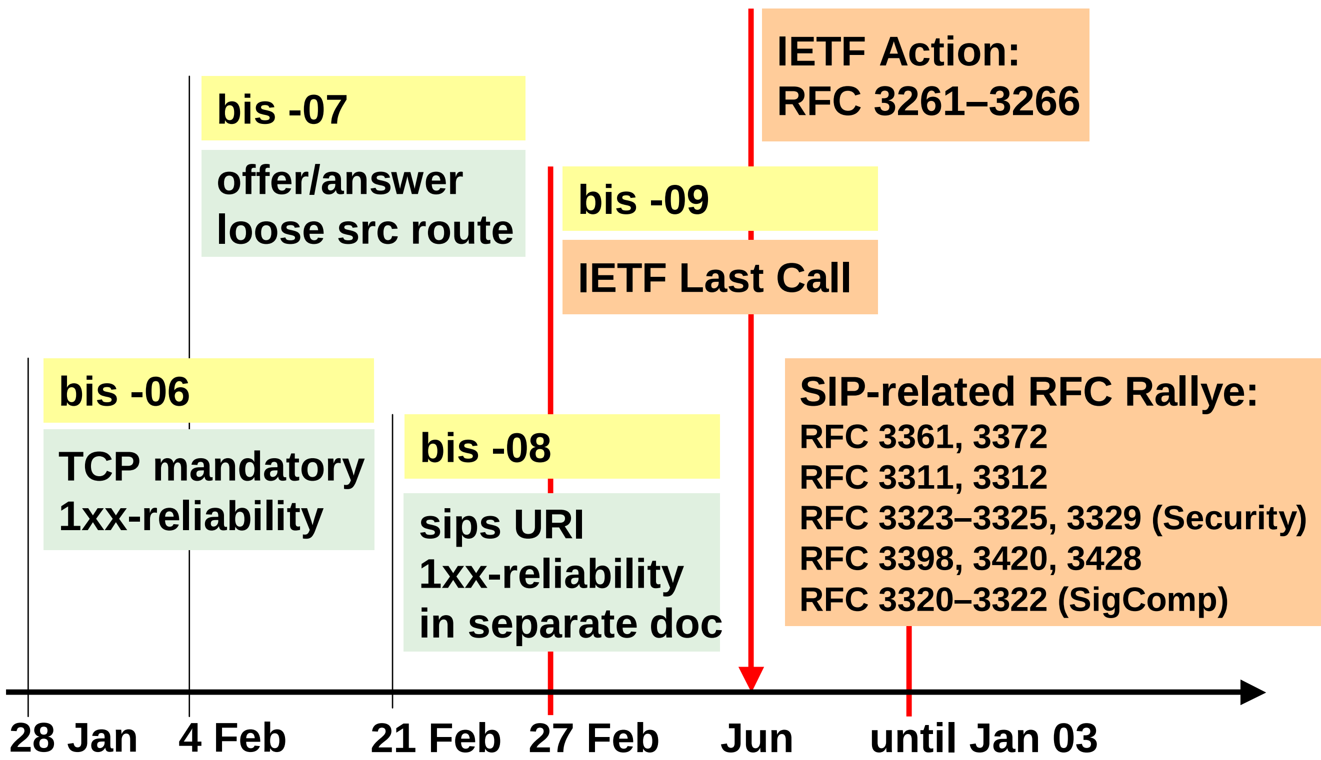
IETF Action: SIP WG formed (Sep 99)

12 Nov 98 | 15 Dec 98 | 15 Jan 99 | 2 Feb 99 | 17 Mar 99 | Sep 99

Timeline: RFC2543bis (2000/2001)



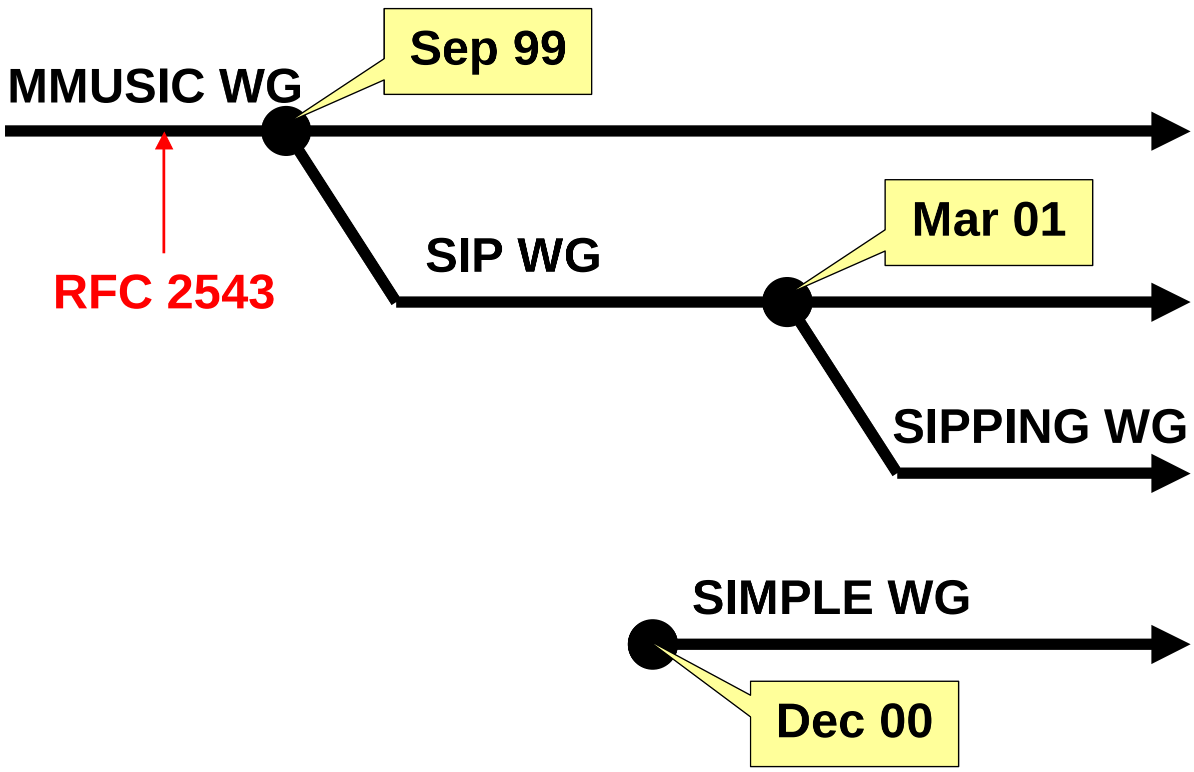
Timeline: RFC2543bis, RFC3261 (2002)



RFCs related to SIP

- **Base spec**
  - RFC 3261: SIP: Session Initiation Protocol
  - RFC 3263: Locating SIP Servers
  - RFC 3264: An Offer/Answer Model with SDP
- **Extended Features**
  - RFC 2976: The SIP INFO Method
  - RFC 3262: Reliability of Provisional Responses in SIP
  - RFC 3265: SIP-specific Event Notification
  - RFC 3311: SIP UPDATE Method
  - RFC 3326: Reason Header
  - RFC 3327: Registering Non-Adjacent Contacts
  - RFC 3428: Instant Messaging
- **Security**
  - RFC 3323: A Privacy Mechanism for SIP
  - RFC 3325: Private Extension for Asserted Identity in Trusted Networks
  - RFC 3329: Security-Mechanism Agreement for SIP
- **Others**
  - RFC 3312: Integration of Resource Management and SIP
  - RFC 3361: DHCP Option for SIP Servers
  - RFC 3398: ISUP to SIP Mapping
  - RFC 3420: Internet Media Type message/sipfrag
  - SDP, RTP, ENUM, SigComp, CMS, AKA, ...

IETF SIP Working Groups (1)



IETF SIP Working Groups (2)

MMUSIC WG →
- SDP extensions
- SDPng

SIP WG →
- SIP core spec maintenance
- SIP protocol extensions

SIPPING WG →
- Requirements for SIP
- Specific SIP application services

SIMPLE WG →
- SIP for Presence and Instant Messaging

SIP is not …

- Suitable for conference control
  - No floor control
  - No participant lists
  - No policies, voting, …
- Designed for distribution of multimedia data
  - Some extensions allow for carrying images, audio files, etc.
- A generic transport protocol!
- Another RPC mechanism
  - SIP has no inherent support for distributed state information
- …
- (but proposals for “misuse” show up again and again)

Tutorial Overview

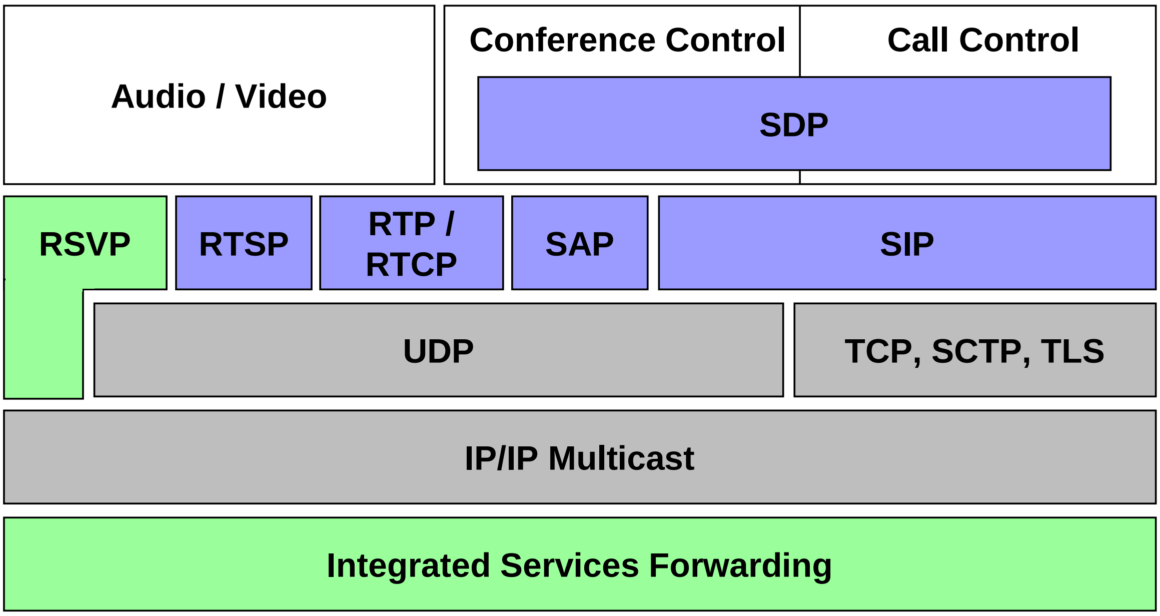
- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony and Deployment Issues
- SIP in 3GPP

You are still here

Protocol Characteristics

- Transaction oriented
  - Request–response sequences
- Independent from lower layer transport protocol
  - Works with a number of unreliable and reliable transports
    - UDP, TCP, SCTP
    - Secure transport: TLS over TCP, IPSec
  - Retransmissions to achieve reliability over UDP
  - Optionally use IP multicast → anycast service
- Independent of the session to be (re-)configured
- Re-use syntax of HTTP 1.1
  → Text-based protocol (UTF-8 encoding)
- Enable servers maintaining minimal state info
  - Stateless proxies
  - Transaction-stateful proxies
  - Dialog (call) state in endpoints (optional for proxies)

SIP and the Multimedia Conferencing Architecture



Terminology

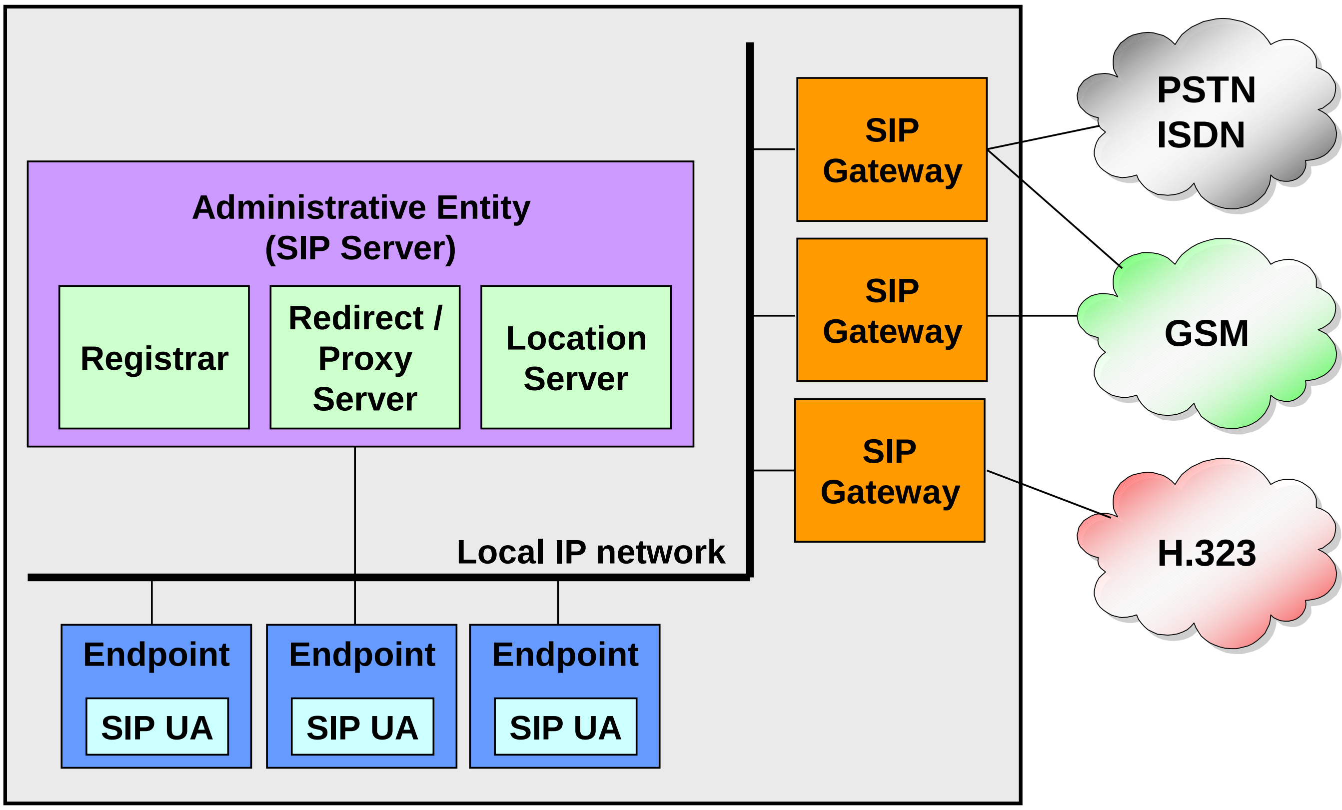
- **User Agent Client (UAC):**
  - Endpoint, initiates SIP transactions
- **User Agent Server (UAS):**
  - Handles incoming SIP requests

User Agent

- **Redirect server:**
  - Retrieves addresses for callee and returns them to caller
- **Proxy (server):**
  - UAS/UAC that autonomously processes requests
    $\rightarrow$ forward incoming messages (probably modified)
- **Registrar:**
  - Stores explicitly registered user addresses
- **Location Server:**
  - Provides information about a target user's location

- **Back-to-Back User Agent (B2BUA)**
  - Keeps call state; more powerful intervention than proxy

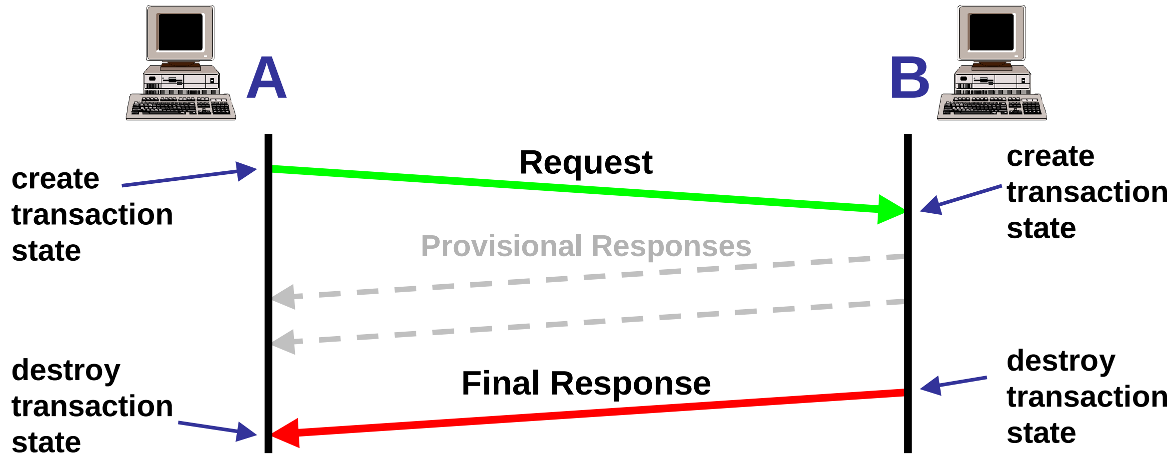
Local SIP Architecture



Functional Layers

Transaction User	session creation, application-specific processing
Transaction	Transaction handling request retransmission
Transport	send/receive SIP messages
Syntax/Encoding	Message parsing
UDP / TCP / SCTP (TLS over TCP and SCTP)	Transport Protocol

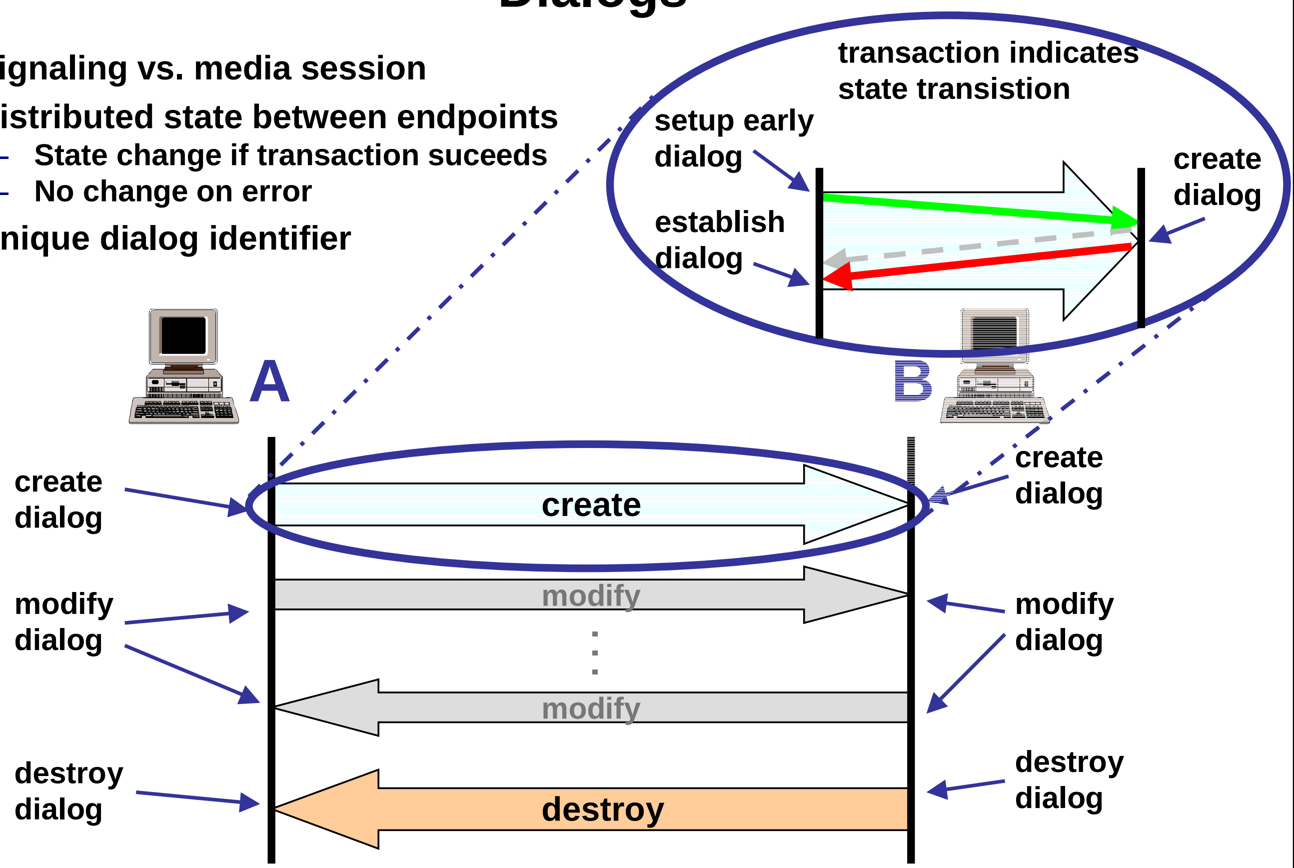
SIP Transactions



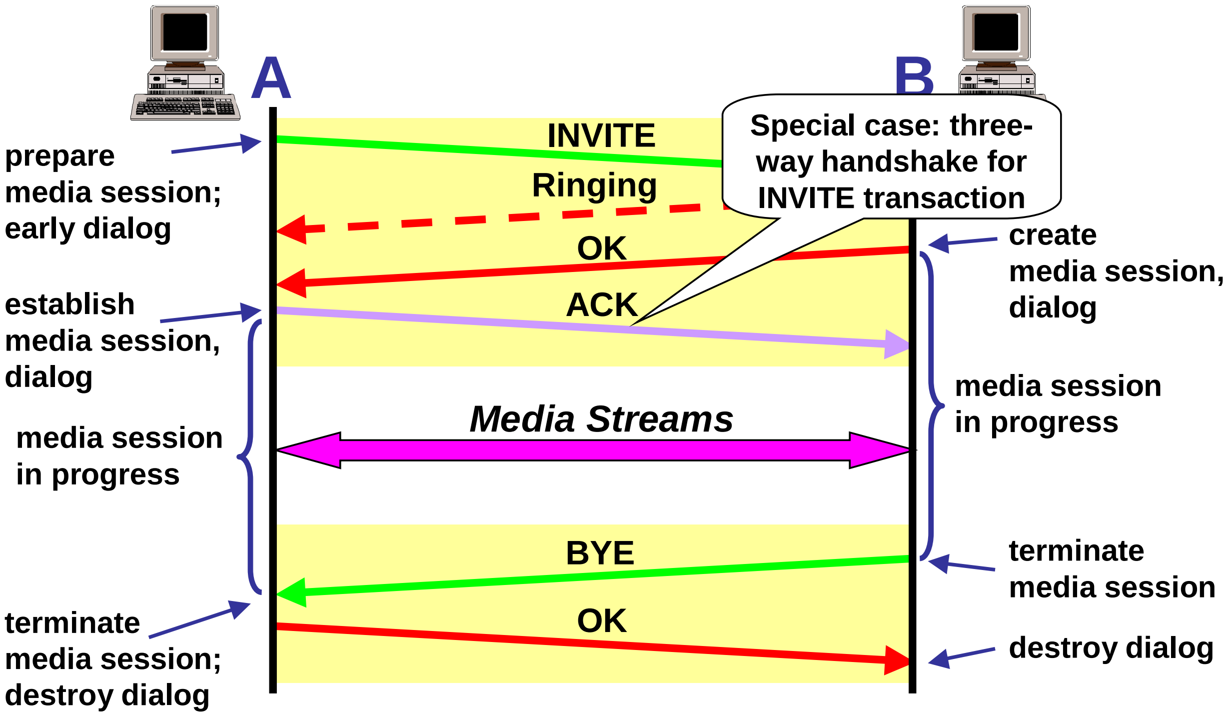
- **RPC-like approach:**
  - **Initial request**
  - **Wait for final response**
- **Provisional responses:**
  - **Additional status information**
  - **May be unreliable**
- **Unique identifier (transaction id)**
  (originator, recipient, unique token, sequence number, ...)
- **Independent completion**

Dialogs

- **Signaling vs. media session**
- **Distributed state between endpoints**
  - **State change if transaction suceeds**
  - **No change on error**
- **Unique dialog identifier**



Dialogs Example: Media Sessions



SIP Message Syntax: Request

Start line

```
INVITE sip:user@example.com SIP/2.0
```

Message headers

```
To: John Doe <sip:user@example.com>
From: sip:jo@tzi.uni-bremen.de;tag=4711
Subject: Congratulations!
Content-Length: 117
Content-Type: applicaton/sdp
Call-ID: 2342344233@134.102.218.1
CSeq: 49581 INVITE
Contact: sip:jo@134.102.224.152:5083
          ;transport=udp
Via: SIP/2.0/UDP 134.102.218.1
```

Message body (SDP content)

```
v=0
o=jo 75638353 98543585 IN IP4 134.102.218.1
s=SIP call
t=0 0
c=IN IP4 134.102.224.152
m=audio 47654 RTP/AVP 0 1 4
```

SIP Message Syntax: Response

Start line

Message headers

Message body
(SDP content)

```
SIP/2.0 200 OK
To: John Doe <sip:user@example.com>;tag=428
From: sip:jo@tzi.uni-bremen.de;tag=4711
Subject: Congratulations!
Content-Length: 121
Content-Type: applicaton/sdp
Call-ID: 2342344233@134.102.218.1
CSeq: 49581 INVITE
Contact: sip:jdoe@somehost.domain
Via: SIP/2.0/UDP 134.102.218.1

v=0
o=jdoe 28342 98543601 IN IP4 134.102.20.22
s=SIP call
t=0 0
c=IN IP4 134.102.20.38
m=audio 61002 RTP/AVP 0 4
```


SIP Addressing Scheme

- SIP URI: generic syntax specified in RFC 2396
- Two roles:
  - Naming a user; typically sip:user@domain
  - Contact address of a user or group; typically contains host name or IP address, port, transport protocol, ...
- May contain header fields for SIP messages
- Support for telephone subscribers instead of user
  → use phone number as specified in RFC 2806

´sip:´ [user [´:´ passwd] ´@´] host [´:´ port] params [´?´ headers]

params ::= (´;´ name [´=´ value])*
headers ::= field ´=´ value? [´&´ headers]

SIP Addressing Examples

```
sip:tzi.org
sip:192.168.42.1
```

Registration domain or IP address

```
sip:john@example.com
sip:john@example.com:5060
sip:foo@example.com:12780
```

Userinfo + domain/host optional port number

URI parameters may carry detailed information on specific URI components:

```
sip:john@Example.COM;maddr=10.0.0.1
sip:+155123456@tel-gw.myitsp.com;user=phone
```

Use URI scheme 'sips' to request secure transmission.

Service Description

Encapsulation

```
sip:sip%3Ajo%40192.168.42.5%3Bmaddr=134.102.3.99@example.com
```

Need to encode reserved characters

Service indication example

```
sip:voicemail.replay=ab1x817m@media-engine;msgid=78
```

Additional header fields (line breaks inserted for readability)

```
sip:sales@warehouse.com;method=INVITE \
?Subject=gw%20c2651&Call-ID=c239xa2-as921b%40warehouse.com

sip:jo@example.com?Replaces=abcd@example.com%3B   \
from-tag%3D28%3Bto-tag%3D234ab1&Accept-Contact=   \
%3Csip%3Ajo%40134.102.218.1%3C%3Bonly%3Dtrue
```

Separator characters

Further Common URI Schemes

Telephony (RFC 2806)

```
tel:+1-555-12345678
tel:7595;phone-context=+49421218
```

ITU-T H.323 Protocol

```
h323:user@example.com
```

Instant Messaging

```
im:user@example.com
```

Presence

```
pres:user@example.com
```


URIs in Header Fields

URI-parameters vs. header parameters

```
Contact:   sip:bob@p2.example.com:55060
            ;methods="NOTIFY"
            ;expires=3600
```

?

→ angle brackets:

URI parameter

```
Contact: < sip:bob@p2.example.com:55060
            ;methods="NOTIFY" >
            ;expires=3600
```

Header parameter

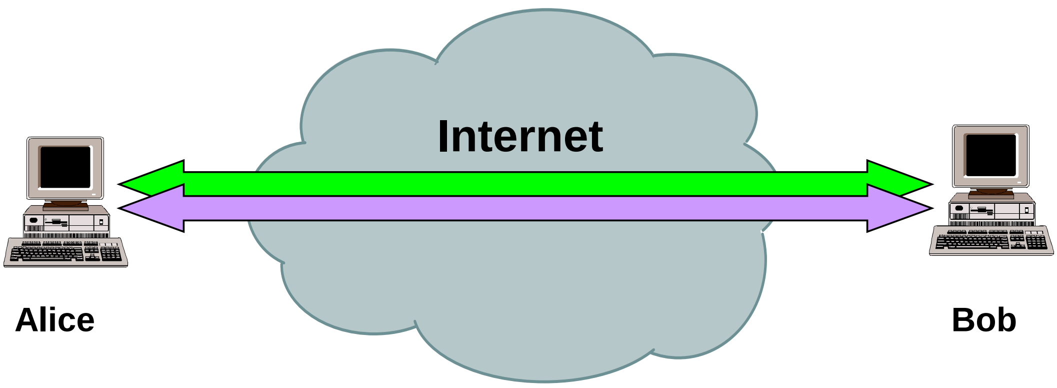
Required if
- URI contains comma, question mark or semicolon
- The header field contains a display name

Tutorial Overview

- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP

You are here

Application Scenario 1: Direct Call UA–UA

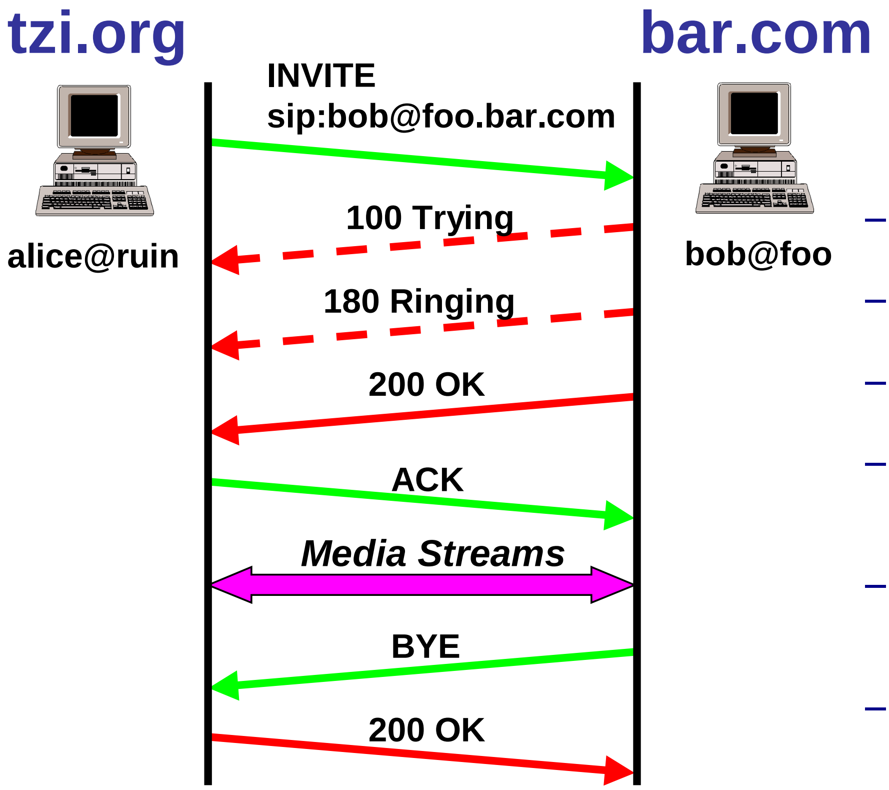


Call signaling

Media streams

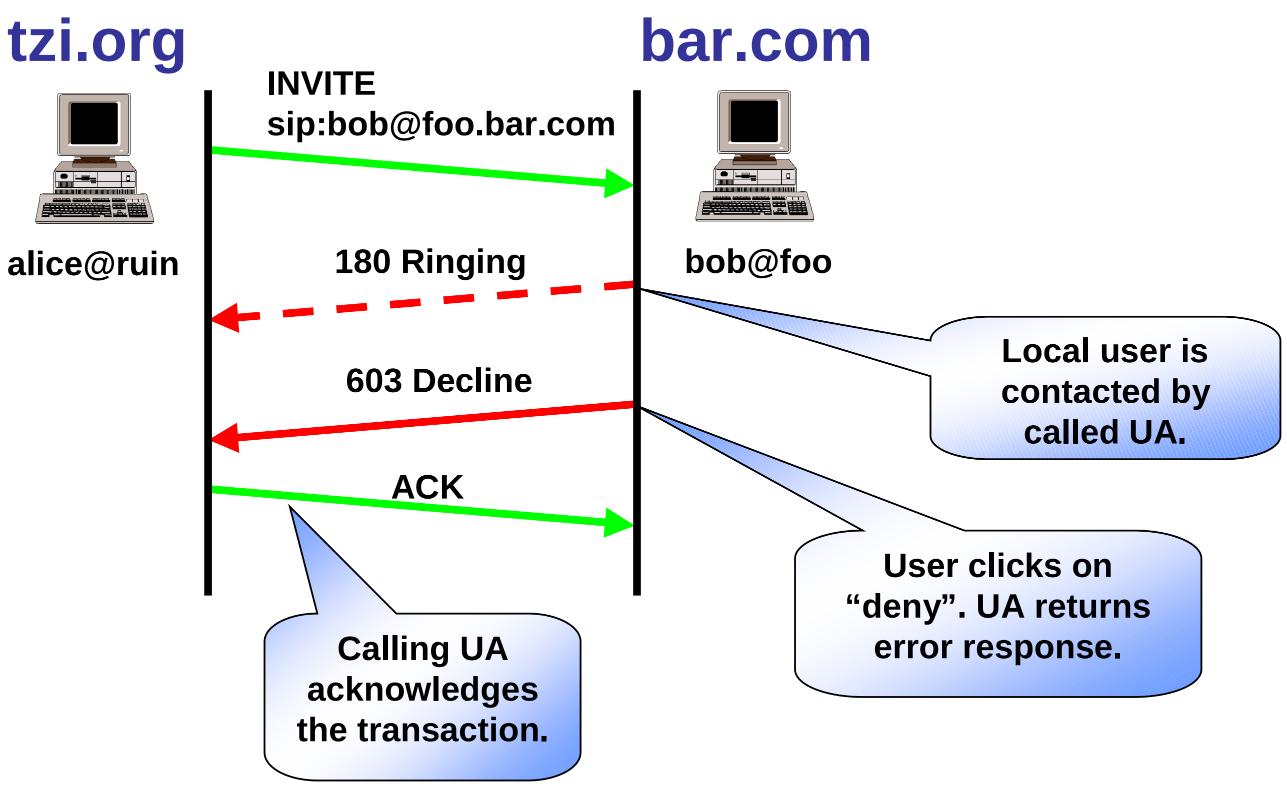
Direct Call

Note:
Three-way handshake is performed only for INVITE requests.

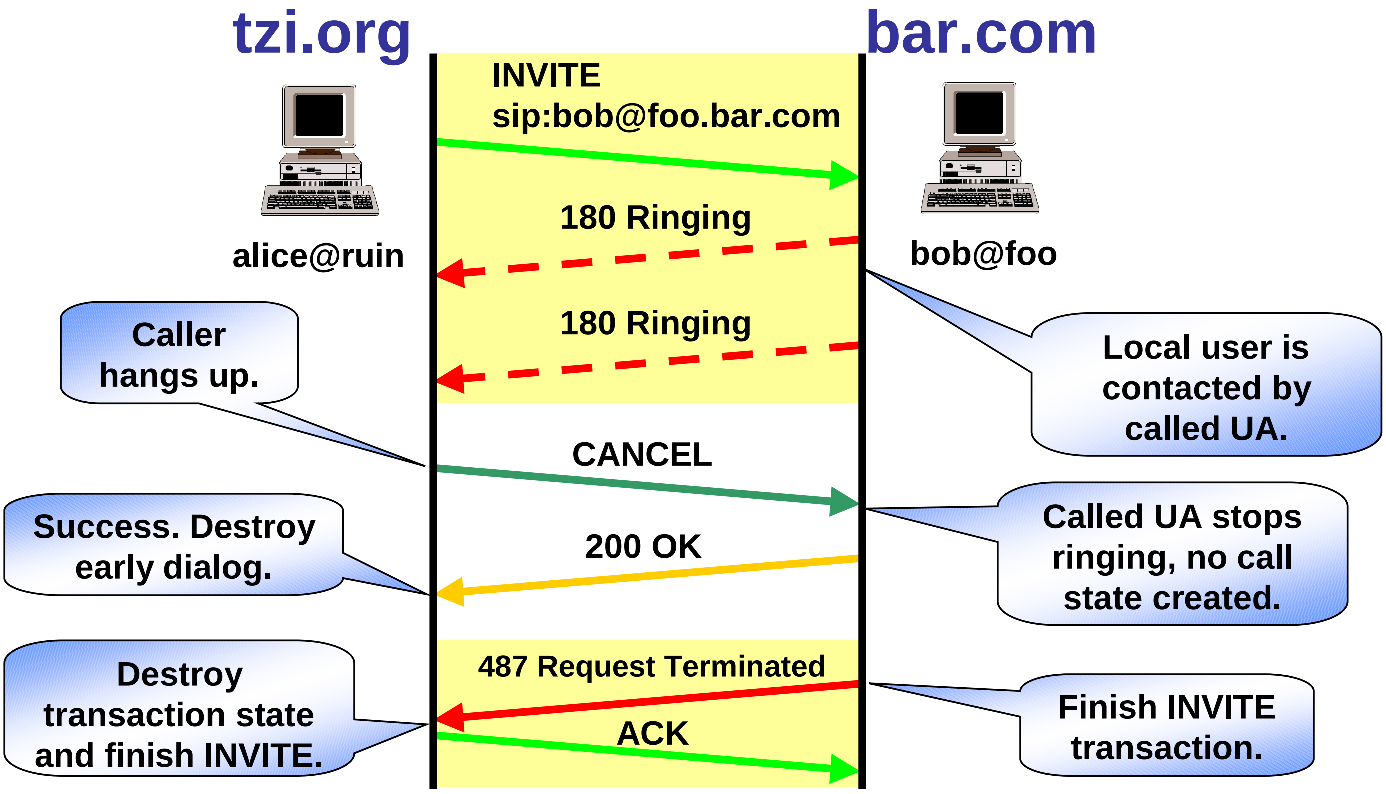


- Caller knows callee's hostname or address
- Called UA reports status changes
- After Bob accepted the call, OK is signaled
- Calling UA acknowledges, call is established
- Media data are exchanged (e. g. RTP)
- Call is terminated by one participant

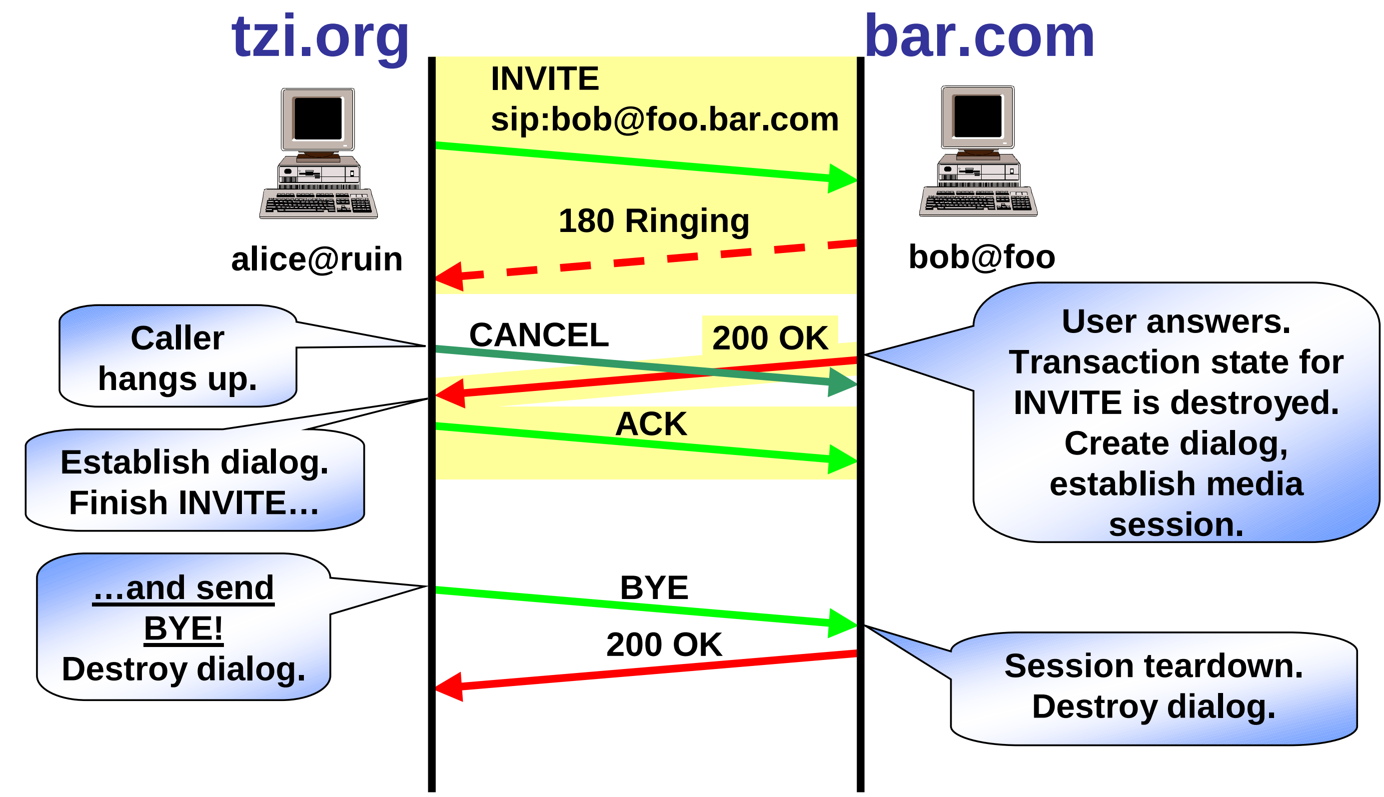
Callee Declines Call



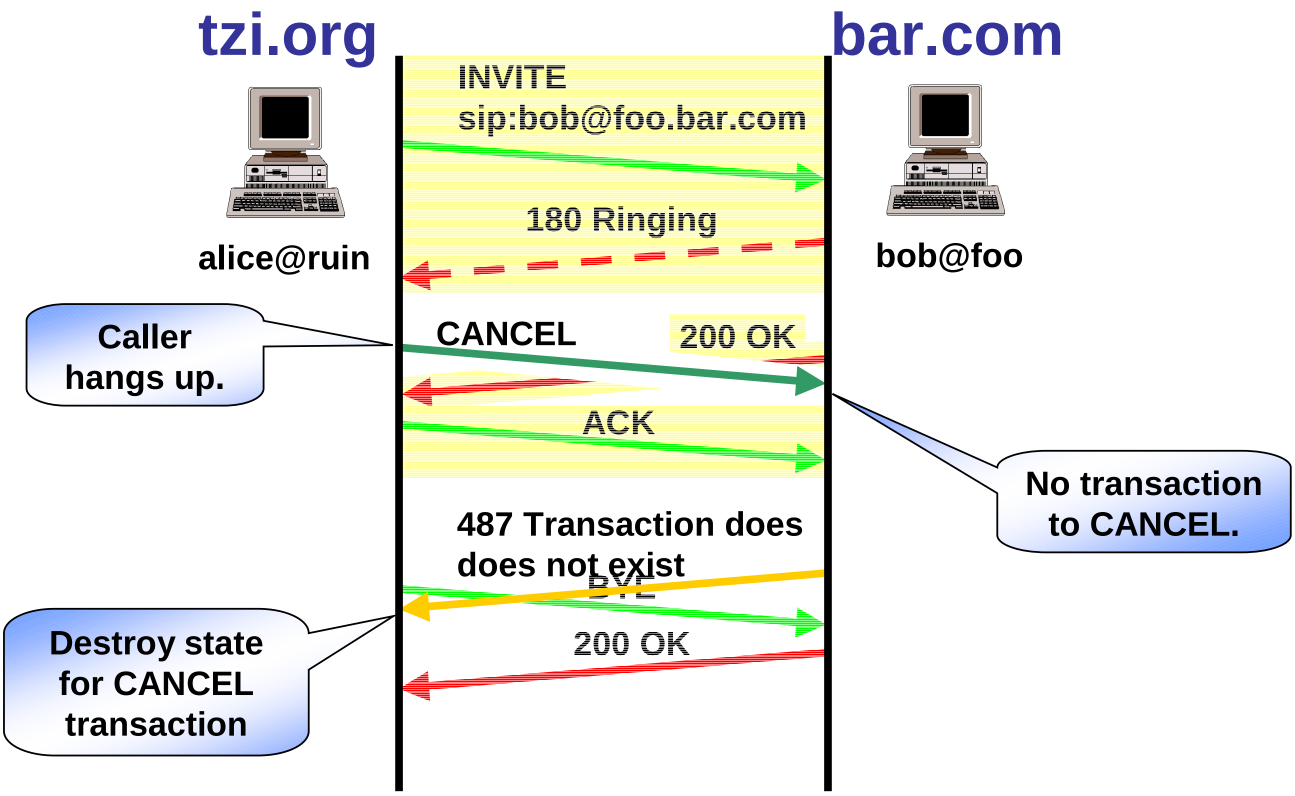
Caller Gives Up



Caller Gives Up While Call Established



Caller Gives Up While Call Established



How to Find The Callee?

- Direct calls require knowledge of callee's address
- SIP provides abstract naming scheme:

sip:user@domain

→ Define mapping from SIP URI to real locations:
  - Explicit registration:
    UA registers user's name and current location
  - Location service:
    Use other protocols to find potentially correct addresses
- Caller sends INVITE to any SIP server knowing about the callee's location
- Receiving server may either redirect, refuse or proxy

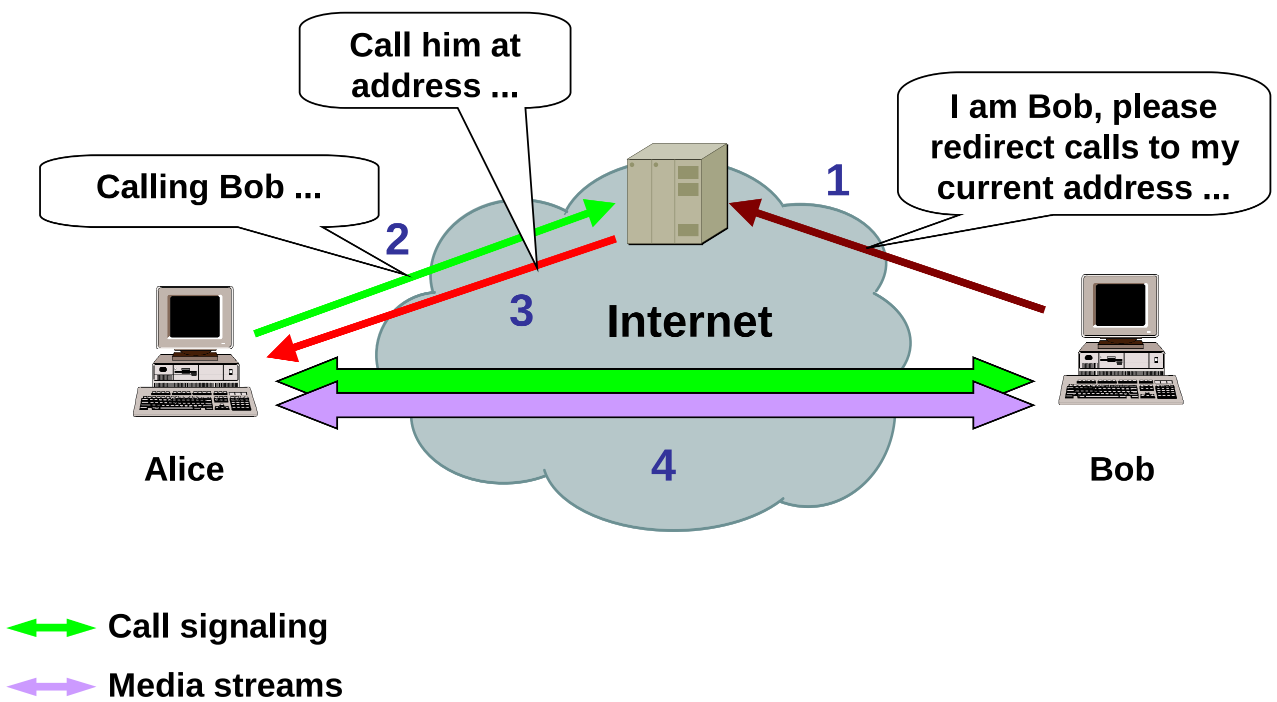
Finding the Next Hop

- UAC may use a configured outbound proxy
- If request URI contains IP address and port, message can be sent directly

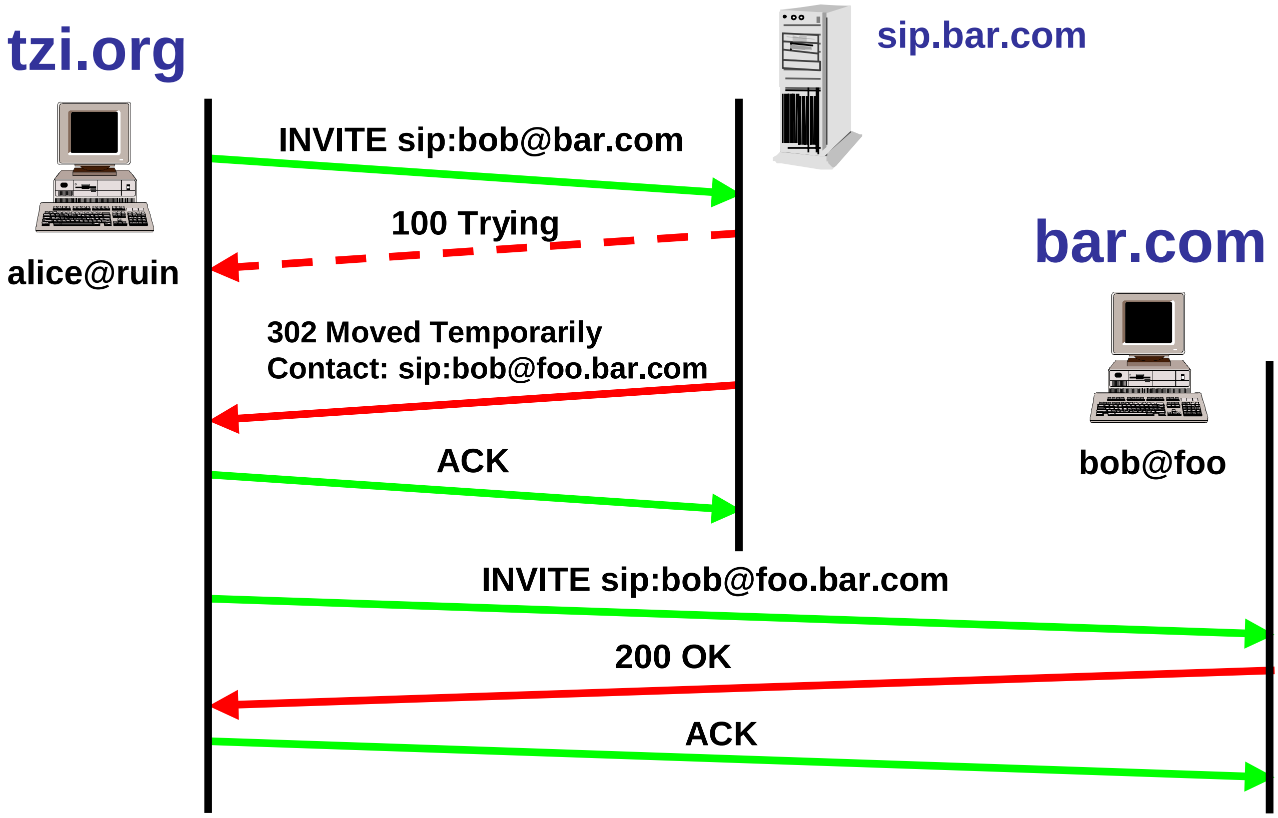
UDP/TCP 5060
TLS 5061

- Otherwise, determine far-end SIP server via DNS
  - Optionally use NAPTR RR
    eventually get ordered list of protocol, IP address, port
  - Query for SRV RR: _sip._udp, _sip._tcp
  - If entries found, try as specified in RFC 2782
- Last resort: query A or AAAA records
  - For specified domain name
  - (Deprecated: For specified *sip.domain*)

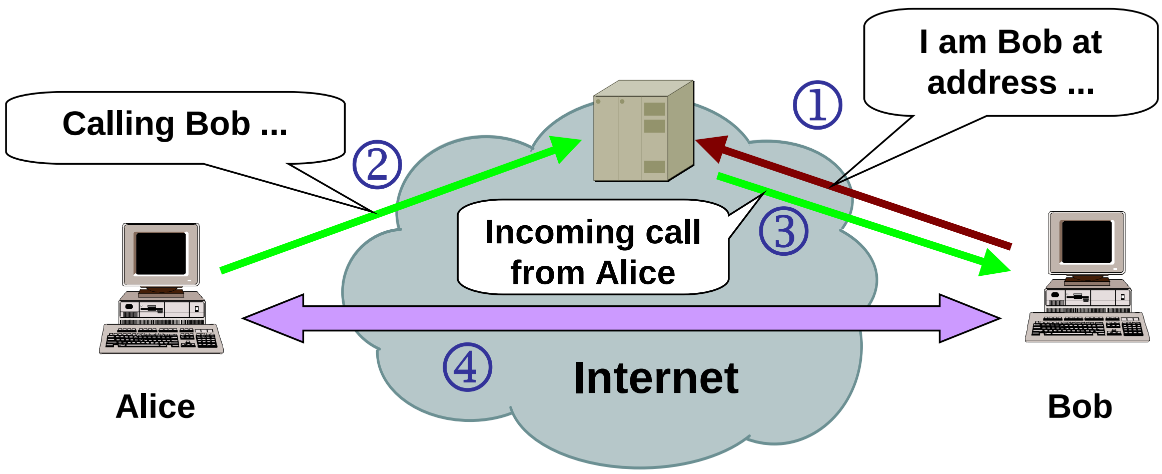
Application Scenario 2: Redirected Call



Redirected Call



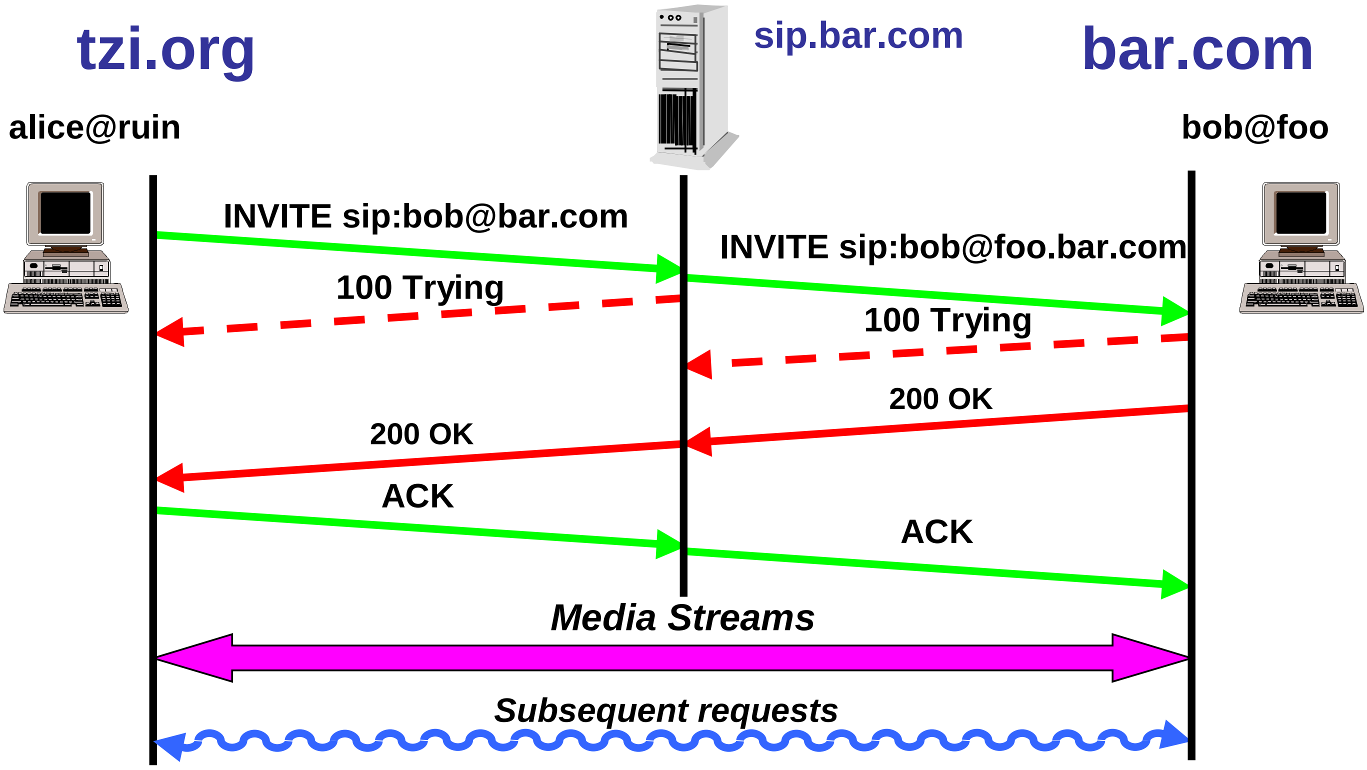
Application Scenario 3: Proxied Call



Call signaling

Media streams

Proxied Call



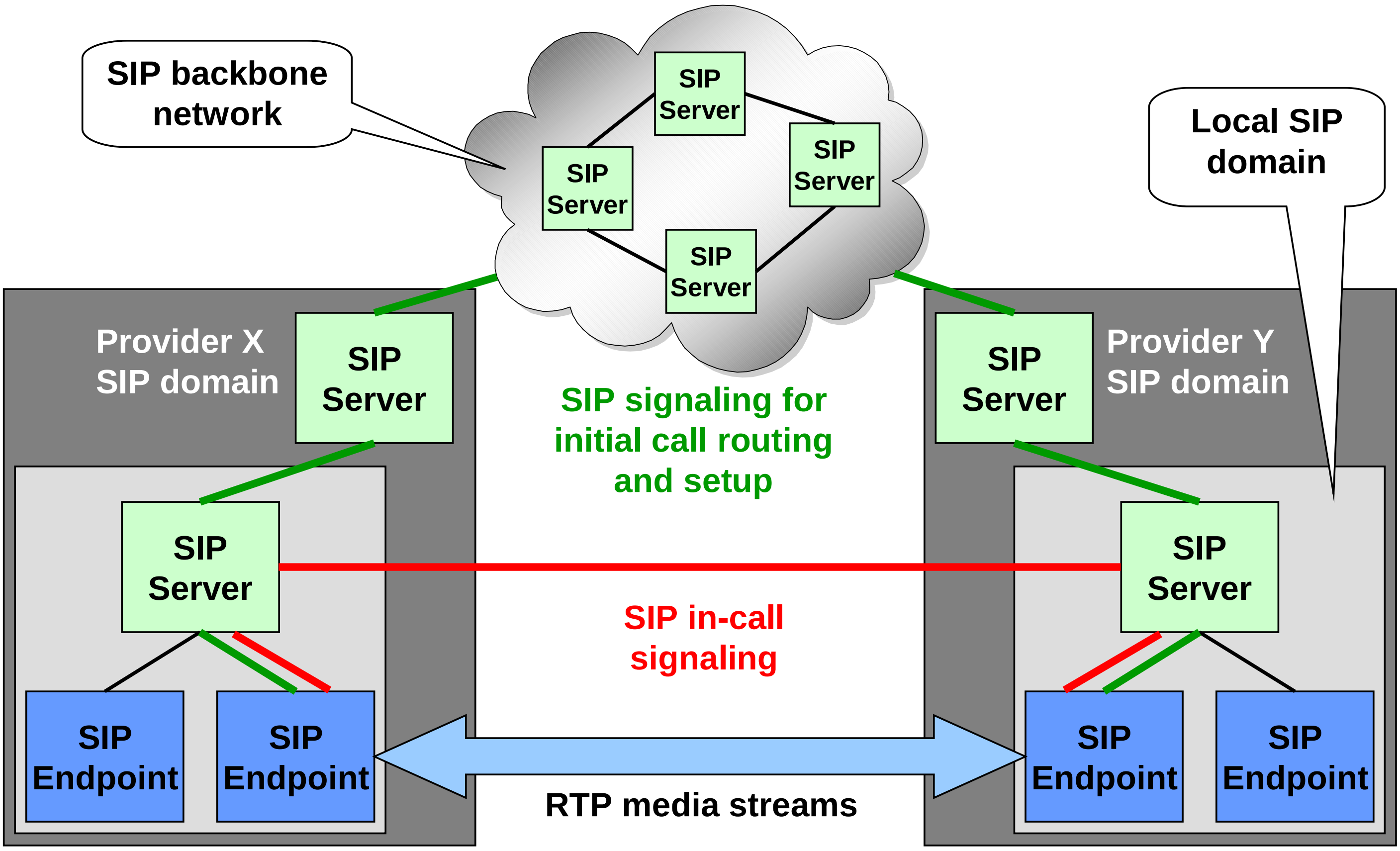
SIP Proxy Functionality

- **Stateless vs. stateful**
  - Stateless: efficient and scalable call routing
  - Stateful: service provision, firewall control, ...

Some roles for proxies

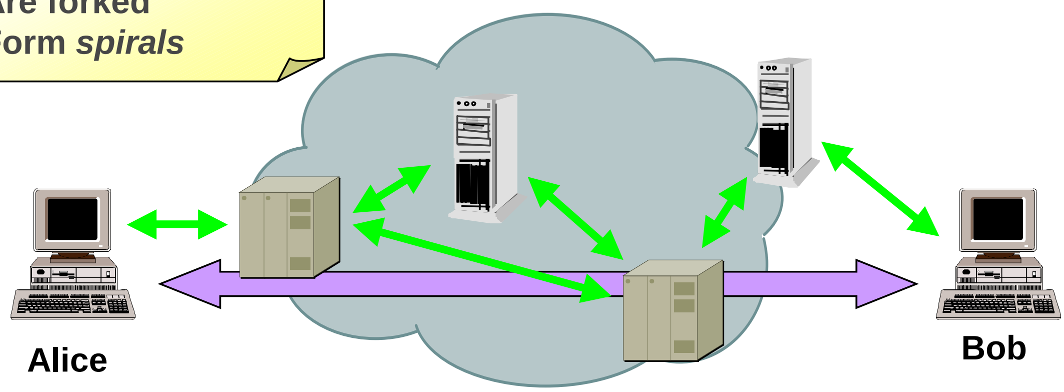
- **Outbound proxy**
  - Perform address resolution and call routing for endpoints
  - Pre-configured for endpoint (manually, DHCP, ...)
- **Backbone proxy**
  - Essentially call routing functionality
- **Access proxy**
  - User authentication and authorization, accounting
  - Hide network internals (topology, devices, users, etc.)
- **Local IP telephony server (IP PBX)**
- **Service creation in general...**

Global SIP Architecture



Application Scenario 4: Proxied Call (Real World)

Requests typically
- Take different paths
- Are forked
- Form *spirals*

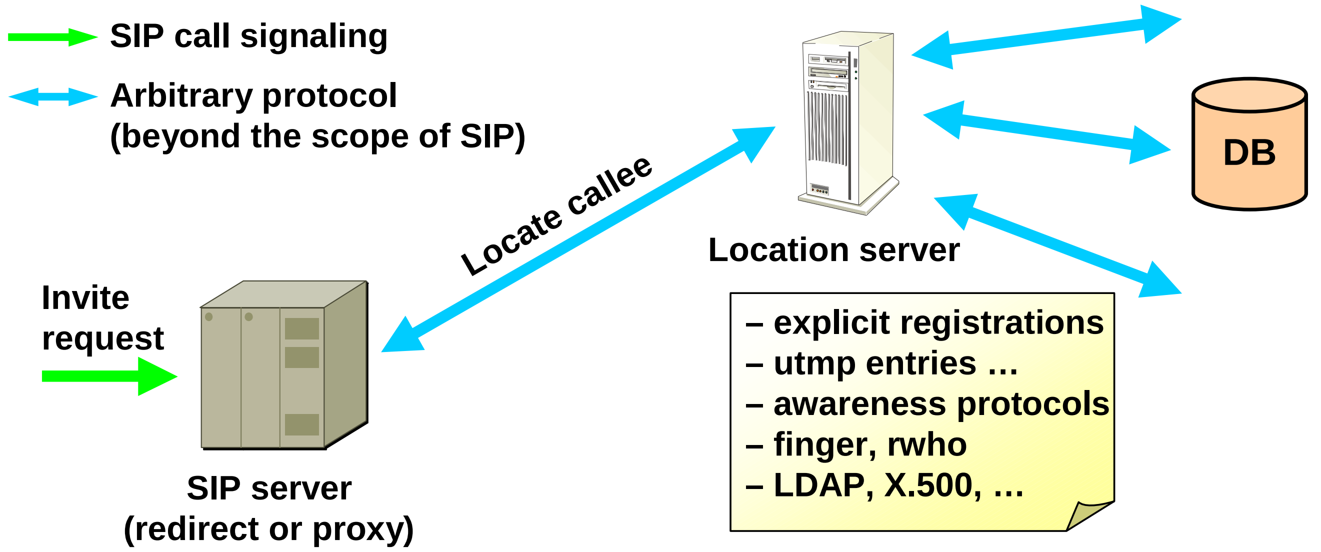


Responses always
Take the reverse path of the originating request

Call signaling

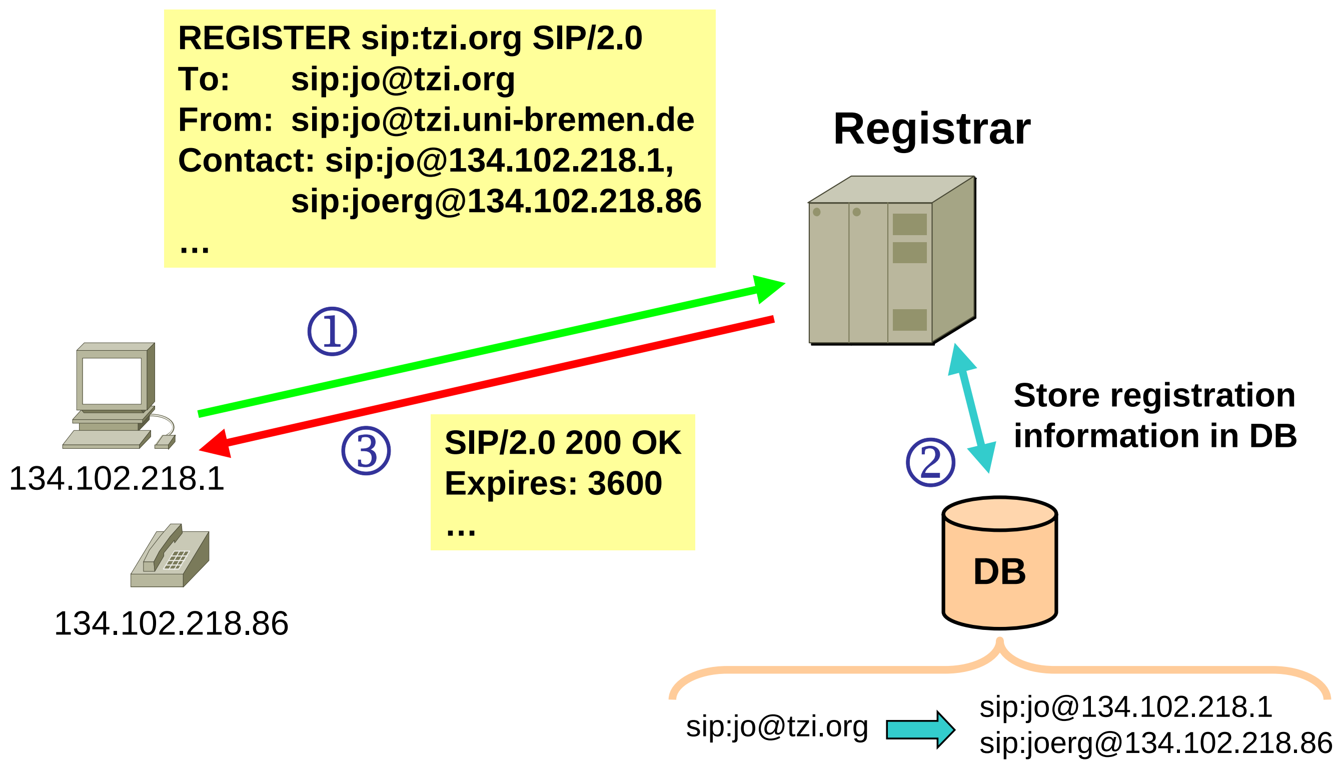
Media streams

User Location



- SIP server asks location server where to find callee
- Location server returns list of contact addresses
- SIP server proxies or redirects request according to address list

User Registration (1)



User Registration (2)

- Send REGISTER request to registrar
- Request URI sip:domain
  → registrar may refuse requests for foreign domains

```
REGISTER sip:tzi.org SIP/2.0
To:      sip:jo@tzi.org
From:    sip:jo@tzi.uni-bremen.de
Contact: sip:jo@134.102.218.1,
         sip:joerg@134.102.218.86
…
```

- To: canonic name for registered user (usually sip:user@domain)
- From: responsible person
  (may vary from To: for third party registration)
- Contact: contact information for the registered user
  → address, transport parameters, redirect/proxy

- Specified addresses are merged with existing registrations
- Registrar denotes expiration time in Expires: header
- Client refreshes registration before expiry

Registration Expiry

- Client requests lifetime
  - Contact:-header parameter expires
  - SIP message header field Expires:
  - Relative duration (seconds) or absolute date (RFC 1123, only GMT)
  - Default if no expiry time requested: 3600 seconds

- Registrar may use lower or higher value, indicated in OK response
  - Registrar must not increase expiry interval, may decline request with “423 Registration Too Brief” and Min-Expiry: header

- After expiration, registrar silently discards corresponding database entries

Add/update Registration Entries

- Retrieve all entries for user@domain (specified in To:-header) from database
- Compare with Contact:-addresses according to scheme-specific rules:
  - Add addresses for which no entries exist
  - Entries being equal to a contact address are updated
- Otherwise return 200 OK response
  - Include all entries for user@domain

Lookup and Delete

- Lookup entries:
  - No Contact:-header in request
    → Current registrations are returned in OK response
  - For further processing:
    - Client uses q-parameter to determine relative order
- Delete entries:
  - Expires: 0
    - Delete URIs specified in Contact:-Header from database
    - Delete all entries for user: Contact: *

Finding a Local Registrar Server

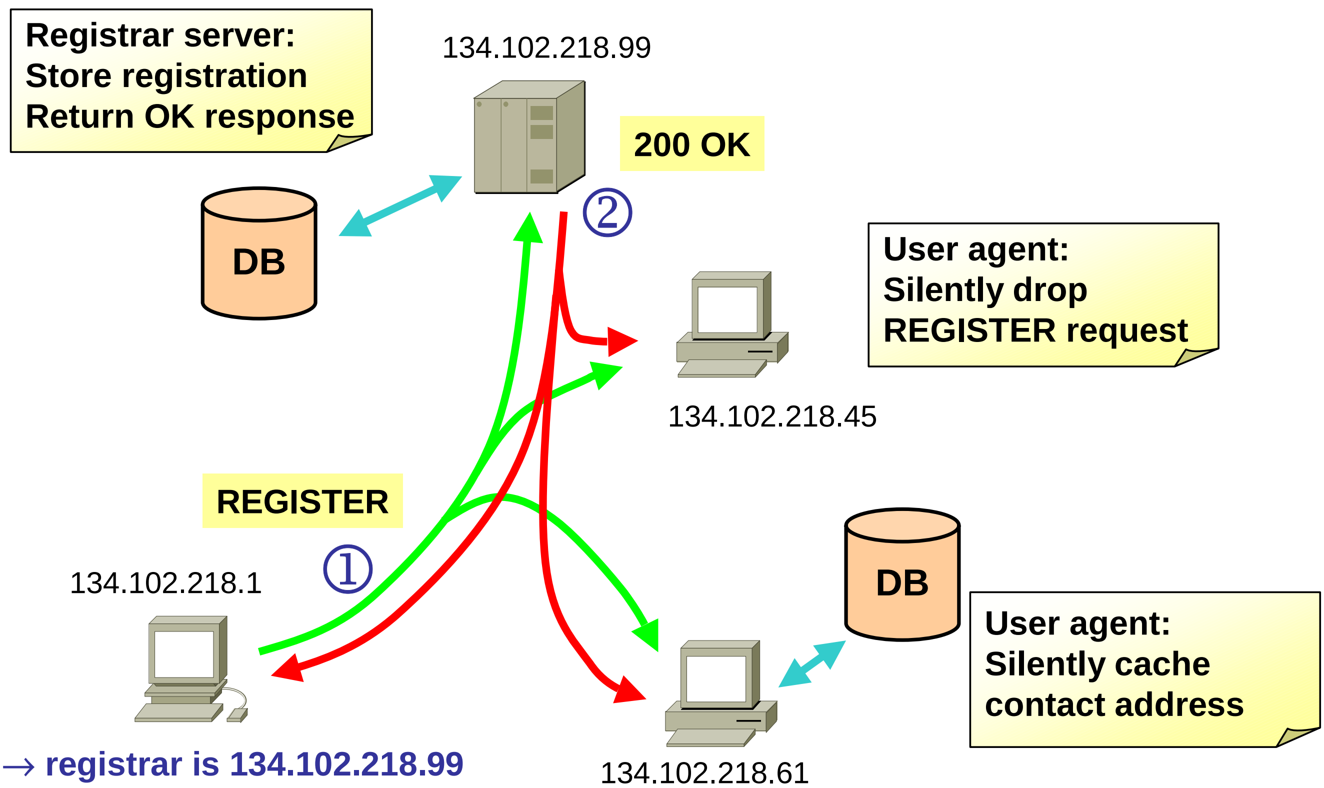
- **Multicast REGISTER request**
  - Send link-local request to sip.mcast.net (224.0.1.75)
  - Use address of first server that responds with OK
  - If other OK responses, use sender addresses as fallback

- **Alternatively, use configured registrar address**

Other mechanisms out-of-scope of core spec. Examples:

- **DHCP**
  - DNS SRV lookup on domain name obtained via DHCP
  - If no SIP server found, query A or AAAA record
- **Service Location Protocol (SLP)**
- **…**

Link-local Multicast Registration

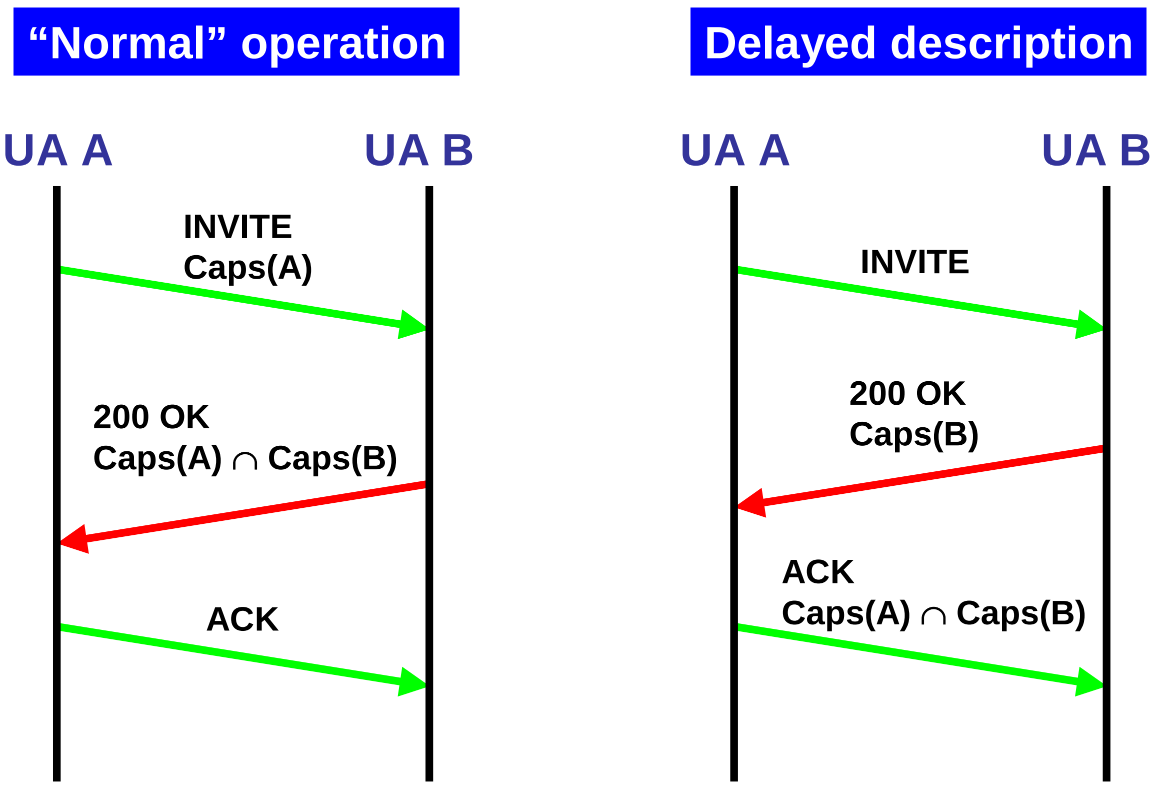


→ registrar is 134.102.218.99

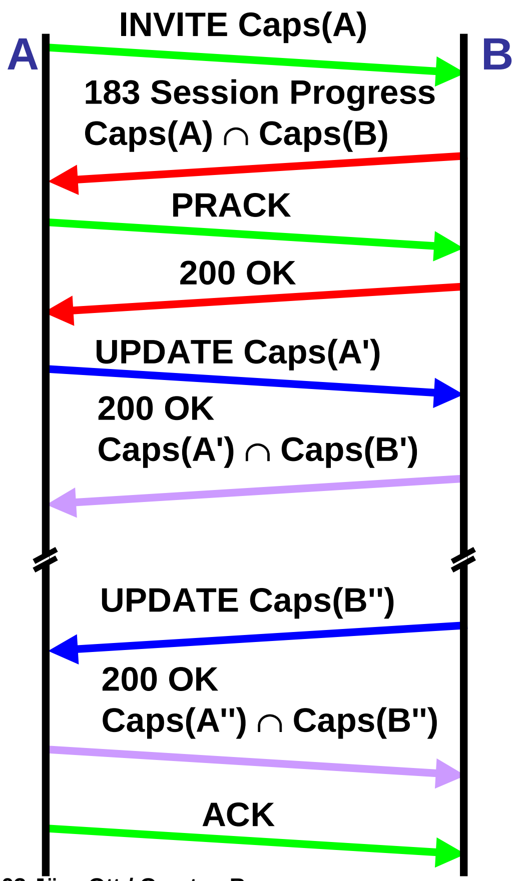
Capability Negotiation

- SDP: Session Description Protocol, RFC 2327
- Caller includes SDP capability description in INVITE
  - Time information may be set to "t=0 0" or omitted
  - For RTP/AVT, use of *rtpmap* mappings is encouraged
- For each media stream (*m*-part of SDP message), callee returns own configuration in response
  - Indicate destination address in *c*=-field
  - Indicate port and selected media parameters in *m*=-field
  - Set port to zero to suppress media streams
- UA may return user's capability set in 200 OK response when receiving an OPTIONS request

Media Negotiation During Call Setup



Use of UPDATE in Offer/Answer Model



- Offerer sends session description
- Answerer must match against local capabilities
- Mid-dialog UPDATE request:
  - establish dialog state first (final or reliable provisional response)
  - No effect on dialog state
- No offer allowed as long as pending offers exist
- Any party may send offer

SDP Example

Length of Time represented by Media in a single Packet

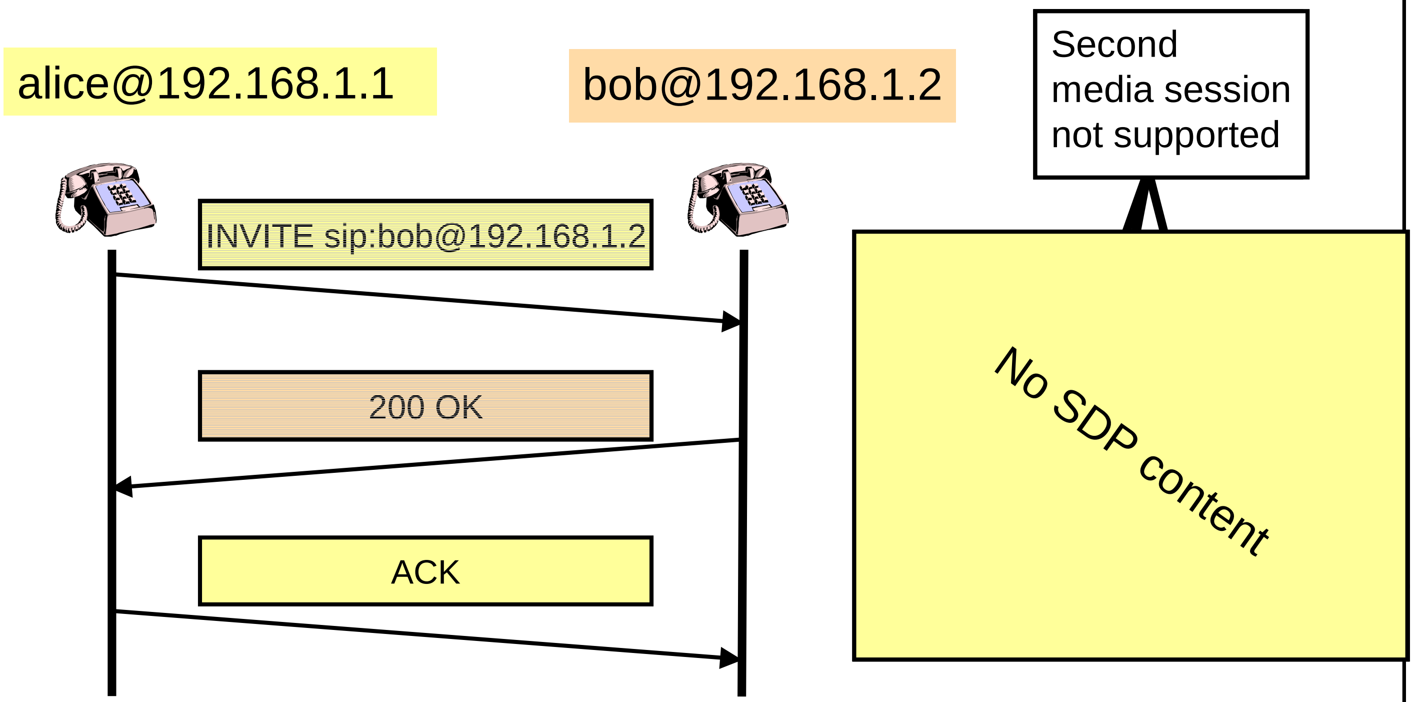
(in SIP: address where originator wants to receive data)

```
v=0
o=llynch 3117798688 3117798739 IN IP4 128.223.214.23
s=UO Presents KWAX Classical Radio
i=University of Oregon sponsored classical radio station KWAX-FM
u=http://darkwing.uoregon.edu/~uocomm/
e=UO Multicasters multicast@lists.uoregon.edu
p=Lucy Lynch (University of Oregon) (541) 346-1774
t=0 0
a=tool:sdr v2.4a6
a=type:test
m=audio 30554 RTP/AVP 0
c=IN IP4 224.2.246.13/127
a=ptime:40
```

Session Level

Media Level

SDP in SIP Applications

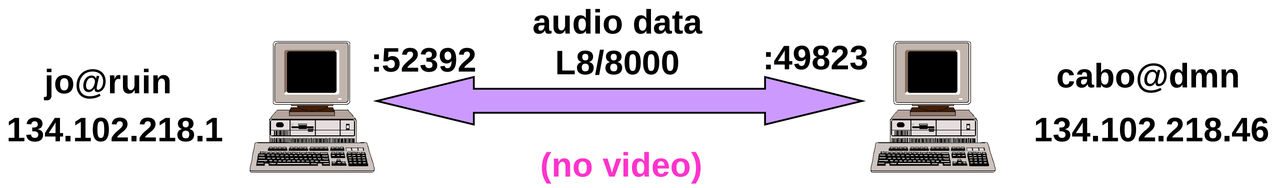


Example SDP Alignment

```
v=0
o=jo 7849 2873246 IN IP4 ruin.inf…
s=SIP call
t=0 0
c=IN IP4 134.102.218.1
m=audio 52392 RTP/AVP 98 99
a=rtpmap:98 L8/8000
a=rtpmap:99 L16/8000
m=video 59485 RTP/AVP 31
a=rtpmap:31 H261/90000
```

```
v=0
o=cabo 82347 283498 IN IP4 dmn.inf…
s=SIP call
t=0 0
c=IN IP4 134.102.218.46
m=audio 49823 RTP/AVP 98
a=rtpmap:98 L8/8000
m=video 0 RTP/AVP 31
```

Resulting configuration:



Send/Receive Only

- **Media streams may be unidirectional**
  - Indicated by *a=sendonly*, *a=recvonly*
- **Attributes are interpreted from sender's view**
- **Sendonly**
  - Recipient of SDP description should not send data
  - Connection address indicates where to send RTCP receiver reports
  - Multicast session: recipient sends to specified address
- **Recvonly**
  - Sender lists supported codecs
  - Receiver chooses the subset he intends to use
  - Multicast session: recipient listens on specified address
- **Inactive**
  - To pause a media stream (rather than deleting it)

Change Session Parameters

- **Either party of a call may send a re-INVITE that contains a new session description**
  - Use other connection address, port
  - Add/remove codecs
  - Append new media streams at the message's end
- **Recipient re-aligns session description with current values**
  - Change media parameters
  - Delete media streams (port has zero-value)
  - Add new streams

→ **Gateways may use re-INVITE when session parameters are unknown during call setup**

UPDATEs for Session State

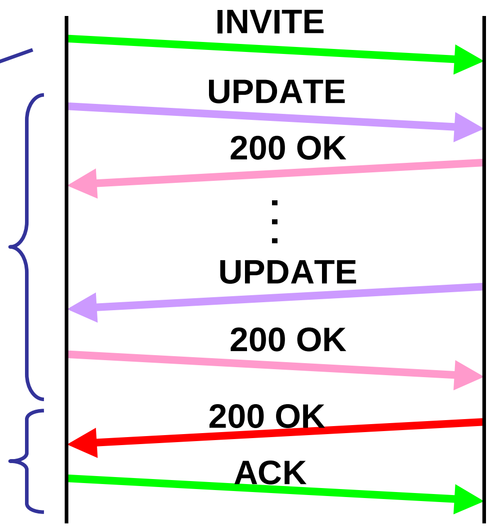
- **UPDATE as generic mechanism to modify session state**
  - Align different proposals: early media, resource reservation, ...
  - Issues with heterogeneous error responses (HERFP) addressed
  - Even useful to establish security associations
  - Dialog state vs. session state
- **Offer/Answer-Model**

Request dialog state update

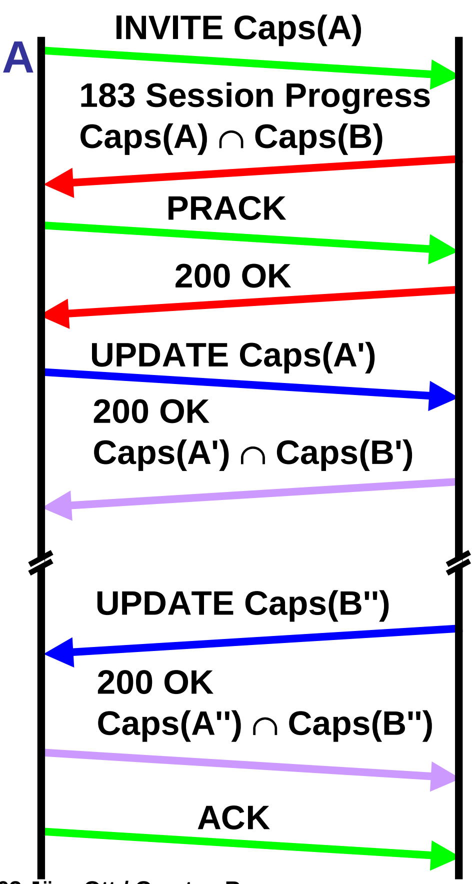
Request session state update

Multiple O/A cycles possible
No overlapping offers
O/A in 183, PRACK etc. allowed
→ Tight relationship between offer and answer necessary

Finish dialog state update



Use of UPDATE in Offer/Answer Model



- Offerer sends session description
- Answerer must match against local capabilities
- Mid-dialog UPDATE request:
  - establish dialog state first (final or reliable provisional response)
  - No effect on dialog state
- No offer allowed as long as pending offers exist
- Any party may send offer

Tutorial Overview

- Internet Multimedia Conferencing Architecture

- Packet A/V Basics + Real-time Transport

- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation

You are here

- SIP in Telephony
- SIP in 3GPP

SIP and Security

SIP entities are potential target of a number of attacks, e.g.

- Spoofing identity
- Eavesdropping
  - Media streams
  - Call signaling
- Traffic analysis
- Theft of service
- Denial of service (DoS)

Typical threats for distributed applications using the public Internet

Some countermeasures:

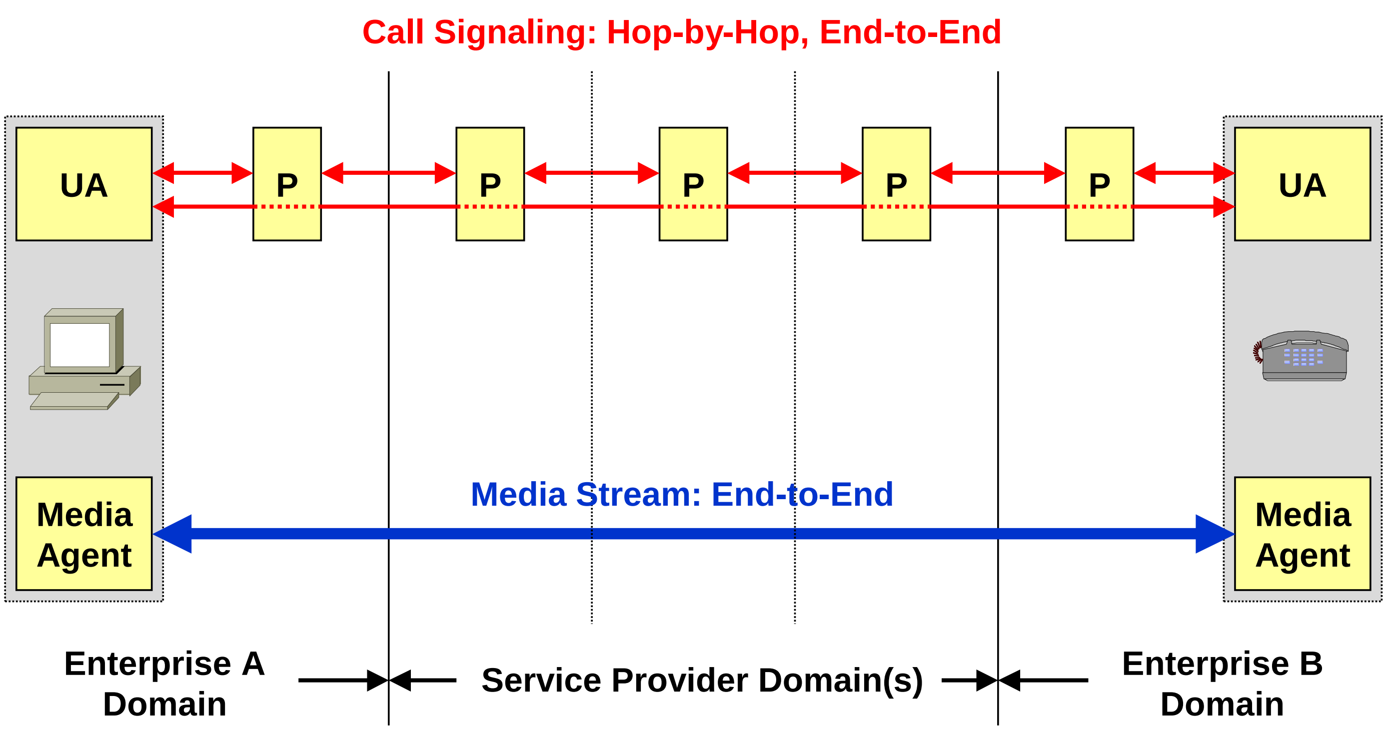
- Client and server authentication
- Request authorization
- Encryption
- Message integrity checks + replay protection

Reuse existing security mechanisms.

Why Carriers Need SIP Security

- **Ensure privacy**
  (media encryption, anonymous calls, personalized services, ...)
- **Billing and accounting**
  (probably pay for assured bandwidth etc.)
- **Regulatory requirements**
  - Call id blocking
  - Prosecute abuse (call tracing facility)
  - Emergency call service
  - Multi-Level Priorization and Preemption

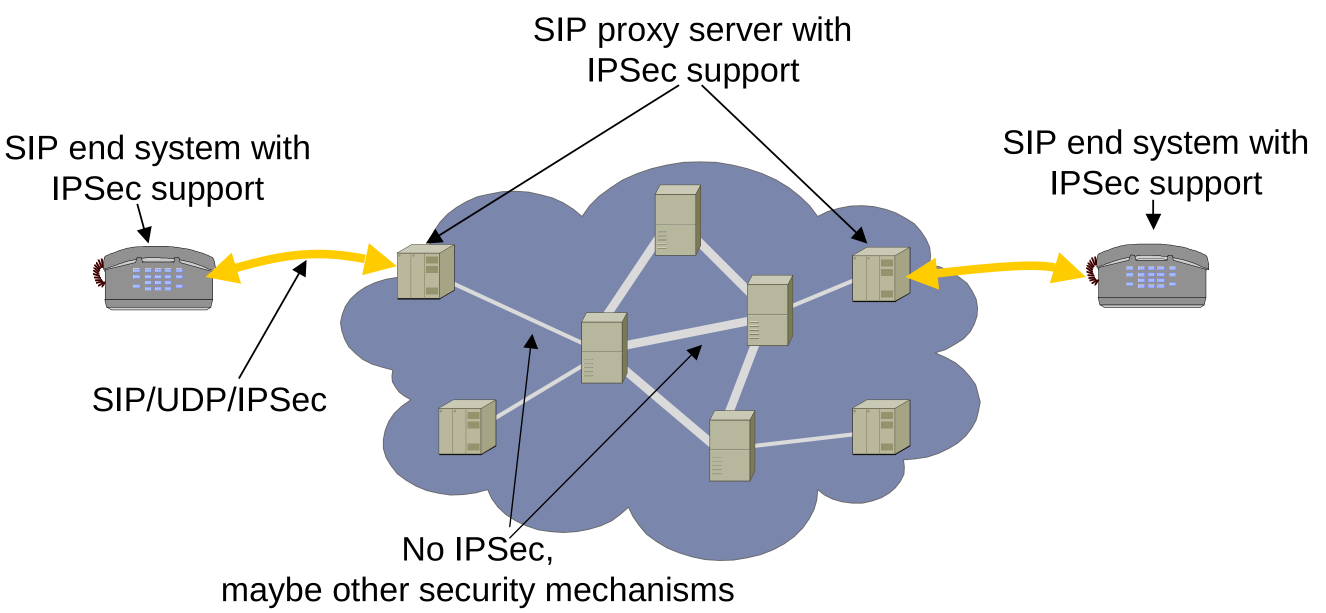
SIP Security Overview



Hop-by-hop Encryption of SIP Messages

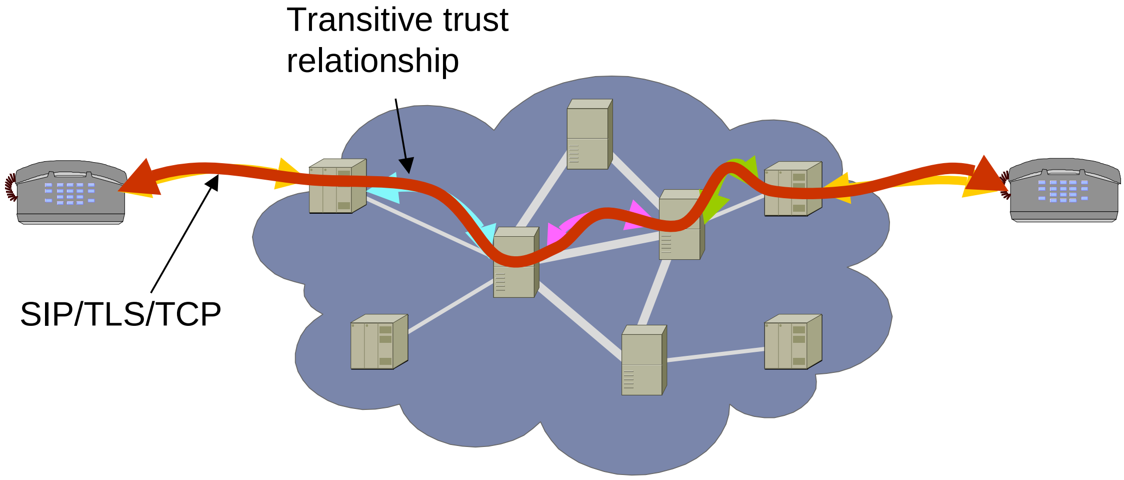
- **Lower layer mechanisms**
  - Applicability depens on link layer technology
- **VPN-like tunnel using IPSec**
  - Suitable e.g. for coupling sites of a company
  - Need OS-support (required for IPv6 anyway)
- **SIP over TLS (Transport Layer Security)**
  - Access to outbound proxy
  - Call routing to ITSP
  - Call routing between neighboring ITSPs (agreements!)
  - In most cases, only servers have certificates
- **Chain of trust: suitable also for authentication**

Hop-by-hop encryption with IPSec



- **Possible deployment scenario**
  - IPSec between hosts inside an administrative domain
  - Established trust relationships, pre-shared keys
- **Security functions independent of SIP layer**

SIP and TLS



- Hop-By-Hop security
  - TLS secures connections between SIP entities
  - TCP only!
- Useful for hosts with no pre-existing trust association
- Usage of TLS is coupled with SIP
  - Needs to be signalled, e.g, in Via headers
  - MUST be implemented by proxy servers, redirect server and registrars
- Basic model
  - A UA establishes a trust association with his outbound proxy (TLS connection)
  - The proxy build trust associations with other proxies/UAs
    - May require certificate exchange and validation
- Problems
  - Call setup delays
  - Resources for open TLS connections

End-to-End Encryption of SIP Messages

- Impossible for entire message: proxies do call-routing
- S/MIME
  - Tunneling SIP messages
  - Repeat important headers outside multipart body

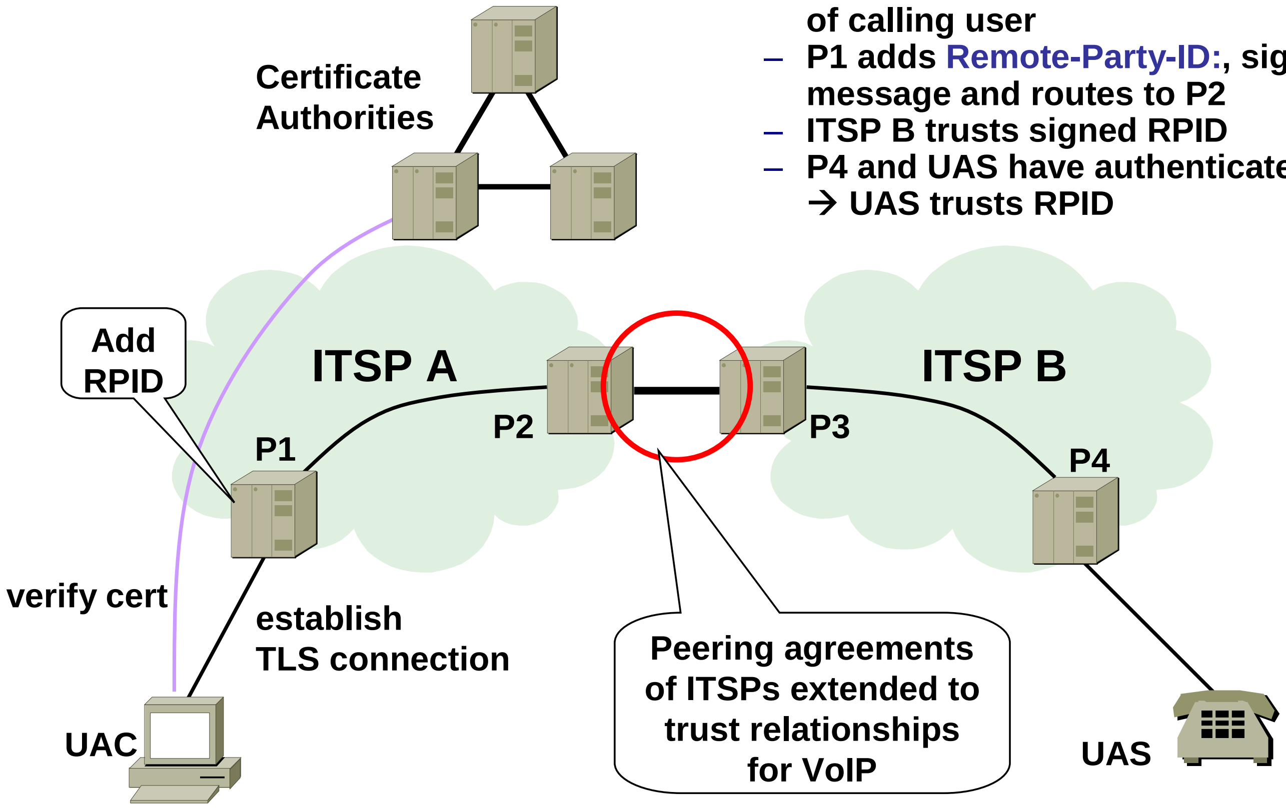
    → complex
- Difficult anyway: Key distribution problem *prior to call*
  - No global PKI
  - Certificates for SIP proxies, typically not for users
  - Typically no shared secrets

SIP and S/MIME

- **Securing MIME bodies in SIP messages**
  - Can be used to achieve end-to-end security
    - E.g., when the network cannot be trusted or other mechanisms are not available
  - Does only protect the message body, not the header fields
- **Tunneling SIP in S/MIME**
  - Encapsulating entire SIP messages within the bodies of other SIP messages
  - Apply MIME security to message body
  - Does also authenticate header fields
  - Encapsulated header fields are a copy of the "outer" header fields
    - Some outer header fields remain visible
      - `To, From, Call-ID, Cseq, …`
    - Some outer header fields may also be changed in transit
      - `Request-URI, Via, Record-Route, …`
- **Issues**
  - Validation of end user certificates will have to rely on public key infrastructures (or pre-configured certificates)

Transitive Trust-Relationships

- UAC establishes TLS connection
  - Proxy provides certificate
  - UAC verifies cert
- Password-based authentication of calling user
- P1 adds Remote-Party-ID:, signs message and routes to P2
- ITSP B trusts signed RPID
- P4 and UAS have authenticated → UAS trusts RPID



Privacy of Calling Party Information

- Extension header **Remote-Party-ID:** (RPID)
  - Intermediate authenticates request initiator and adds RPID (failed authentication will be explicitely indicated)
  - Message is forwarded across trust boundaries only if requested privacy is guaranteed (RPID-Privacy header)
  - Initiator and intermediary can specify a required level of privacy
  - Multiple headers for different types of RPID information (user, subscriber, terminal)

- Extension header **Anonymity**
  - IP address privacy using anonymizer

- Signed RPID can be used to carry authenticated address of request initiator
  → trust relationships between ITSPs

SIP Media Privacy

- Encryption of (RTP) media streams
  - use old RTP encryption scheme
  - use Secure RTP (SRTP) profile
    - Currently finalized within the IETF

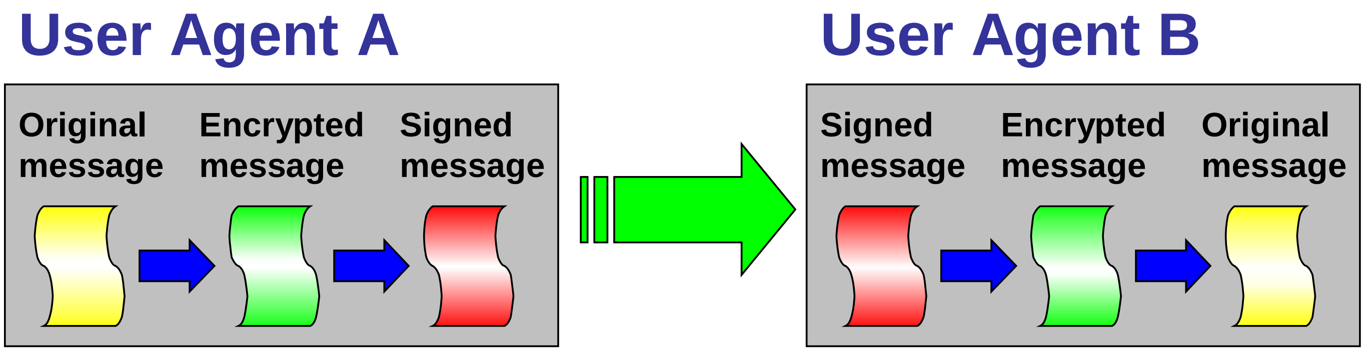
- Secure key distribution between endpoints *in a call*

- SDP allows for per media key field (“k=“)
  - Requires encrypted SDP in SIP message body
  - Not really feasible today

- SDP extensions for media key
  - Allows for end-to-end negotiation of keys
  - Protection of the exchanged information within SDP

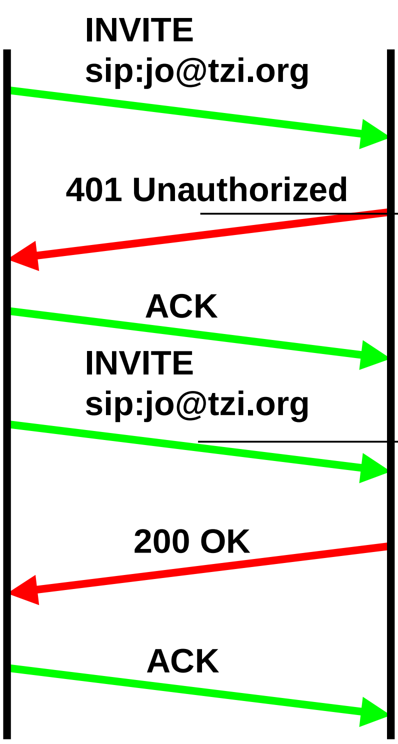
SIP Authentication

- HTTP *digest* authentication
- Challenge with WWW-Authenticate: header field
- Authorization: header field
  - End-to-end authentication of UAC
  - Digitally sign message body and header fields following the *Authorization:* header
  - Use canonical form
  - Fields to be changed must precede *Authorization*



TZi Technologie-Zentrum Informatik
Universität Bremen

Authentication: Example Call Flow



WWW-Authenticate: Digest realm="TZI",
domain="sip:tzi.org", nonce="qf73…",
stale=FALSE, algorithm=MD5

Authorization: Digest username="jo",
realm="TZI", nonce="qf73…",
response="50c6a6071bc8..."

- Security domain is identified by realm and original request URI
- Caller must create new request with incremented CSeq value
- Attention: parameters separated by "," — header-split not allowed!

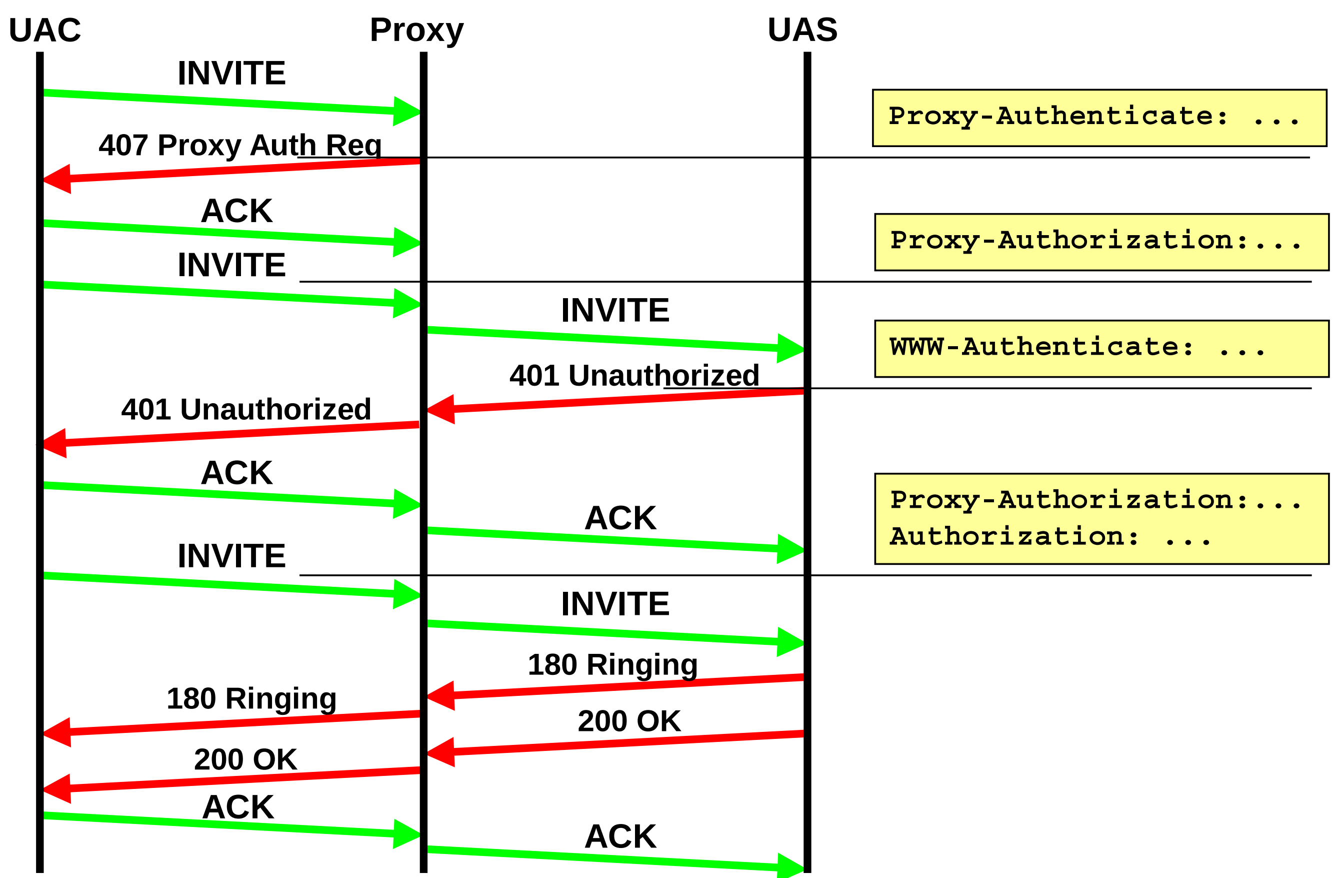
Authentication for Proxies

- Similar to endpoints (HTTP Digest)
- Proxy rejects client request with “407 Proxy Auth required”
  - Proxy-Authenticate: header
  - Multiple proxies along the path may challenge
- Client resubmits request with credentials for Proxy
  - in Proxy-Authorization: header
  - Multiple headers with credentials may need to be included
- Finally:

HTTP Basic is deprecated! Forbidden in RFC2543bis

Do not send passwords “in the clear”

Multiple Authentication Steps



Tutorial Overview

- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation

You are here

- SIP in Telephony and Deployment Issues
- SIP in 3GPP

Extension Mechanisms

- Proxies forward unknown methods and headers
- UAS' ignore unknown headers, reject methods
- Feature negotiation
  - Headers: Require, Proxy-Require, Supported
  - Option tags for feature naming (see below)
  - Error responses:
    - 405 Method not allowed
    - 420 Unsupported
    - 421 Extension Required

  } UAC and UAS/proxy must agree on common feature subset
- Option tags
  - Identified by unique token
  - Prefix reverse domain name of creator
  - IANA: implicit prefix *org.ietf.*

Some Current SIP Extensions

- Reliable provisional responses
- Session Timers
- Early Media
- Adjusting session state: UPDATE
- INFO method
- REFERing peers to third parties
- SIP for subscriptions and event notifications
- Instant messaging
- ...

- Addressed later today where appropriate...

Reliable Provisional Responses

- Signaling Gateways, ACD systems etc.
  may depend on provisional responses

  → need optional reliability

- Option tag 100rel:
  - End-to-end reliability of provisional responses > 100
  - Retain order of reliable responses

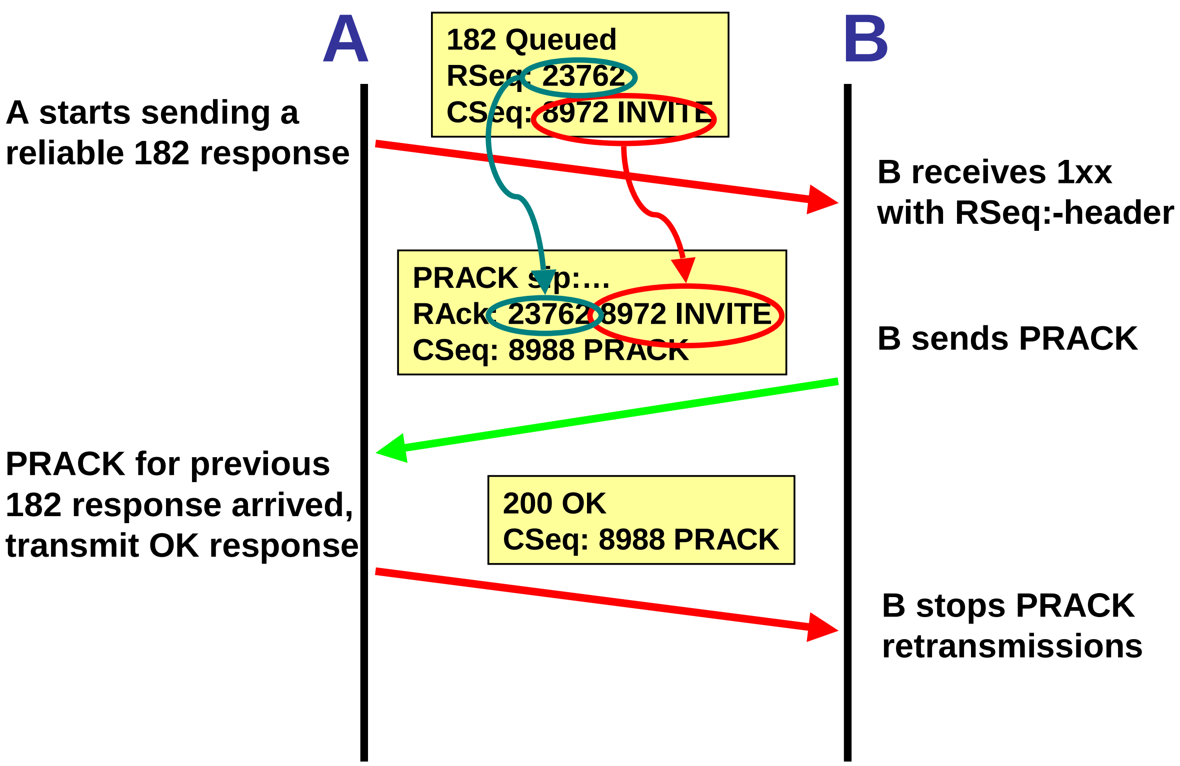
- Implementation:
  - Windowing: RSeq/RAck headers
  - Explicit acknowledge: PRACK request

Use of PRACK

- Insert RSeq:-header with random integer value
  - new responses must increment RSeq: by 1
- Retransmit with exponential backoff
- No subsequent reliable provisional responses until first PRACK
  - enables re-ordering at receiver side

- PRACK request
  - RAck: contains RSeq: and CSeq: values to acknowledge
  - May contain body with additional information

Example Call Flow: PRACK



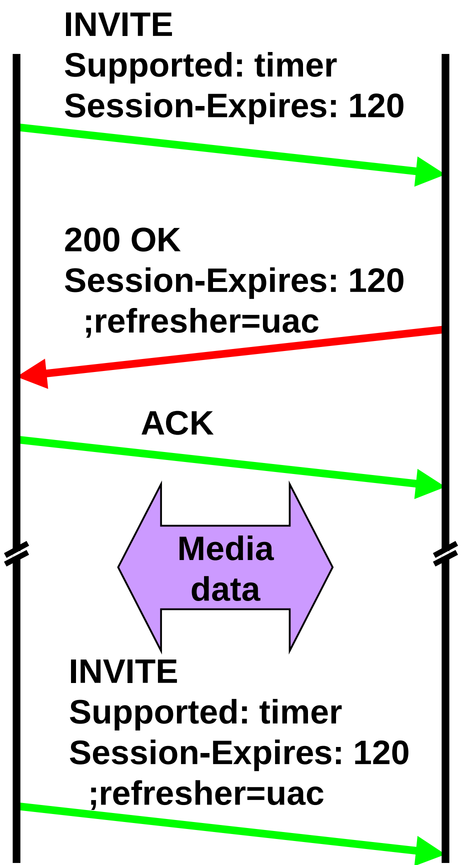
Session Timers

- **No keep-alive mechanism for SIP calls**
  - Proxies may preserve outdated state information
  - Close dynamically created holes in firewalls
  - Need timeout-mechanism to perform cleanup

- **SIP Extension for Session timers**
  - Option tag timer
  - Negotiation mechanism for expiry time and refresh responsibility
    - Session-Expires: duration and role
    - 422 response with Min-SE: lower bound for expiry interval
  - *Responsible UA* sends re-INVITE before timer expires

- **Session is terminated if no re-INVITE arrived in time**
  - *Responsible UA* sends BYE
  - Proxies silently drop state information

Example: Keep-alive



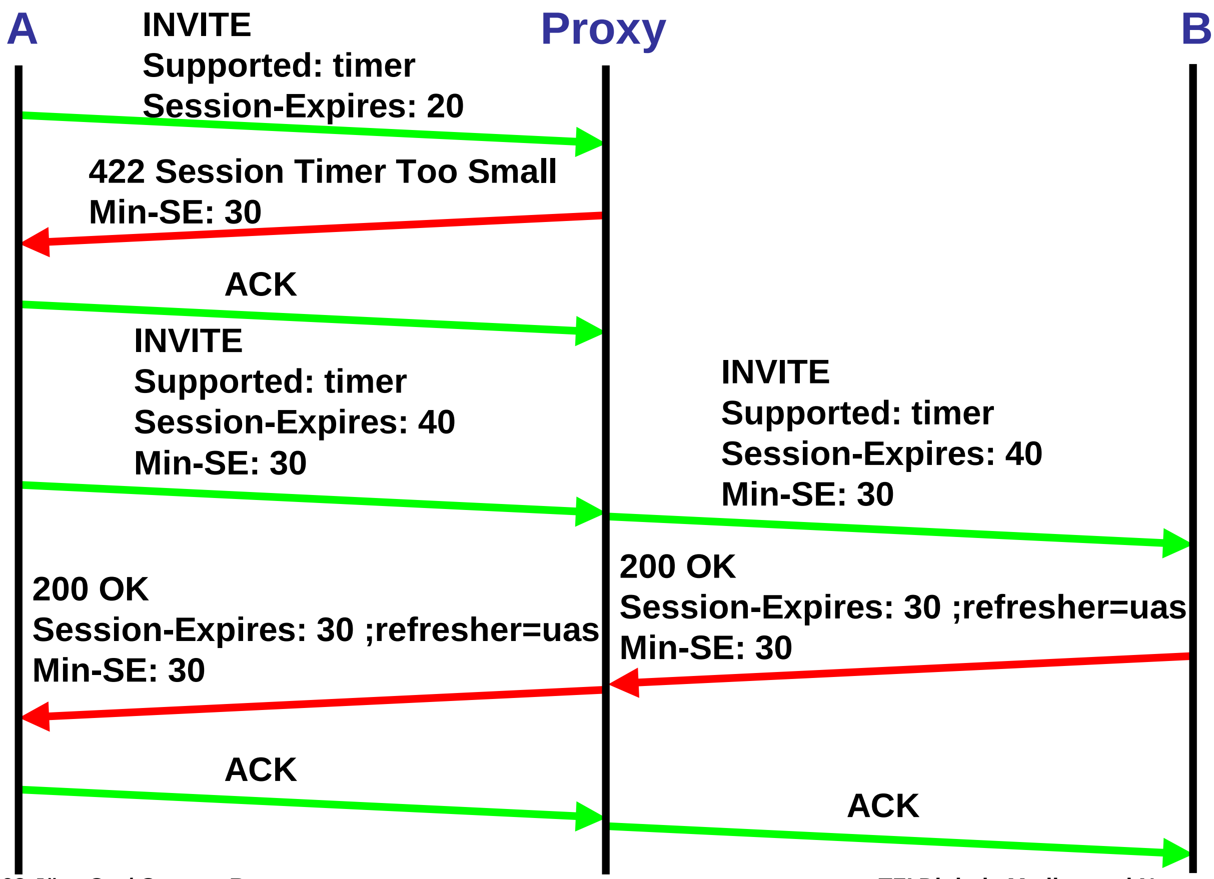
- Caller indicates support of session timers in INVITE
- Propose initial timer value

- Called UA supports timers, accepts proposed value, asks the caller to send refresh

- Setup complete, media streams are established

- After n/2•timer calling party sends re-INVITE
- Call continues after 200 OK and ACK

Example: Interval Negotiation

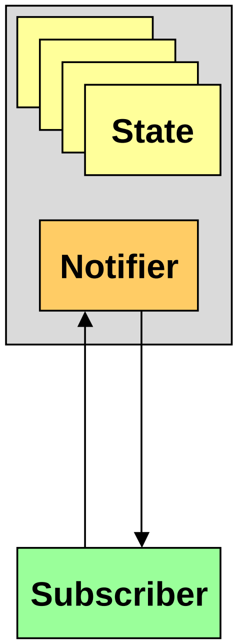


Event Notifications

- **Need for flexible event notification**
  - Enable presence information
  - Better support for mobile SIP applications
  - Home network appliances
  - Feedback about progress of other calls, conference state, etc.
  - NOTIFY method, Event:-header
- **Event subscription**
  - SUBSCRIBE, Event:
  - Events may be call-related or third-party generated
  - Security issues: Sensitive Events
    - Privacy
    - Authentication
- **Used for personal presence applications**
- **Augmented by MESSAGE method for Instant Messaging**

Event Concept

- **Piece of state information S**
  - Identified by some name (“package”)
- **SIP entities interested in S**
  - Query for the current state
    - “polling”
  - Be notified about changes to S
    - SUBSCRIBE
  - Subscriptions may be created implicitly
    - By means of other (SIP or non-SIP) protocol activity
- **Information about S carried in message body**
  - NOTIFY
  - Formats to be defined specific to S
- **Protection of S**
  - Keep control of who gains access, who has access; for how long



SUBSCRIBE

- **SUBSCRIBE used to registered interest in a piece of state**
  - Event: identifies the state information to be retrieved
    - 489 Bad Event
    - Syntax reflects package concept:
      `package ( '.' template )*` → e.g. ”`Event: presence.winfo`”
  - Allow-Events: used to indicate which event packages are supported
  - Expires: how long to subscribe
    - Subscriptions are soft-state; need to be refreshed periodically
    - May be shortened or lenthened by server
  - Expires: 0
    - Poll the state information once; do not estabilish a subscription
  - Body may indicate desired notification policy
- **Each SUBSCRIBE triggers at least one NOTIFY**
  - May contain no (useful) information if access not yet authorized

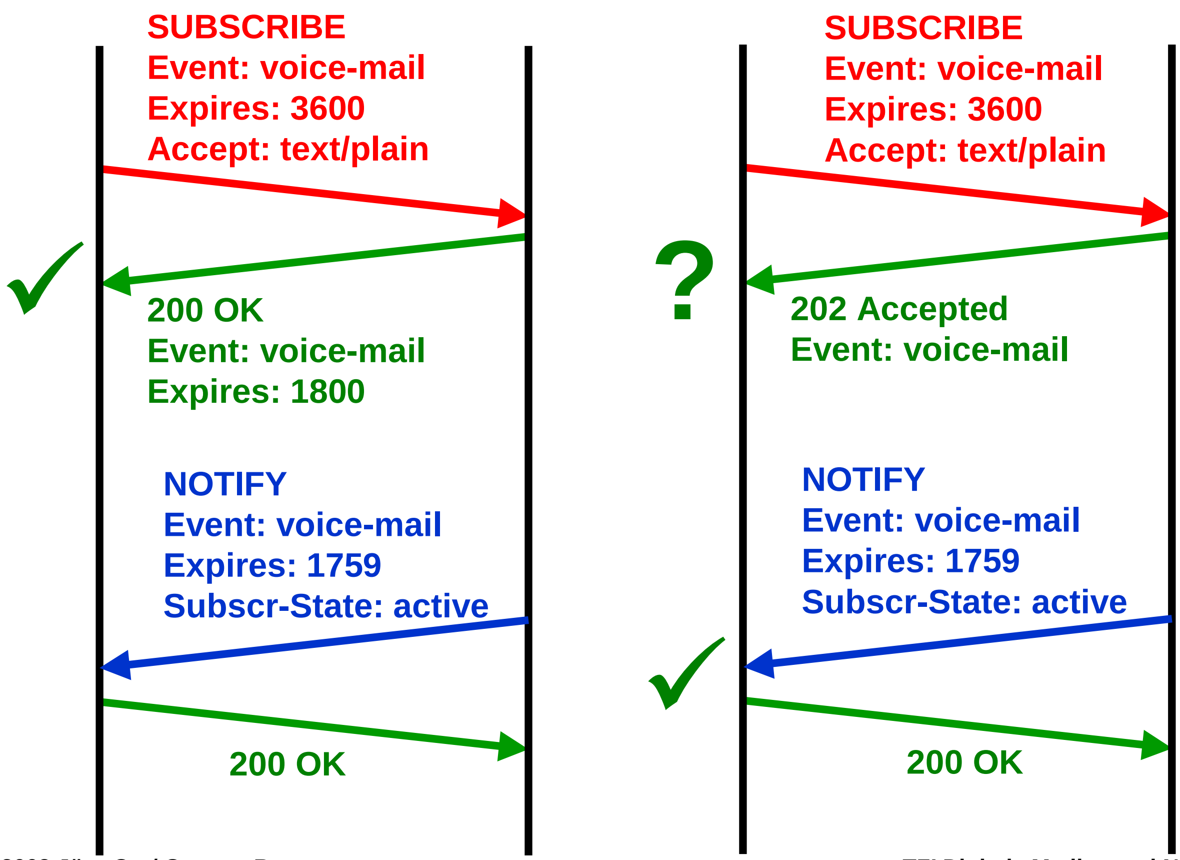
SUBSCRIBE Response

- **SUBSCRIBE Responses**
  - 200 OK: Everything's fine: event understood, client authorized, etc.
  - 202 Accepted : Request received, working on a final result
  - 401, 603, … if applicable
  - Acceptance or rejection to be reported in NOTIFY
- **155 Update Request**
  - Typically for authorization instead of 401
- **Initiate dialog after receiving 155 or 2xx response**
  - Mid-dialog SUBSCRIBE and UPDATE to refresh subscription

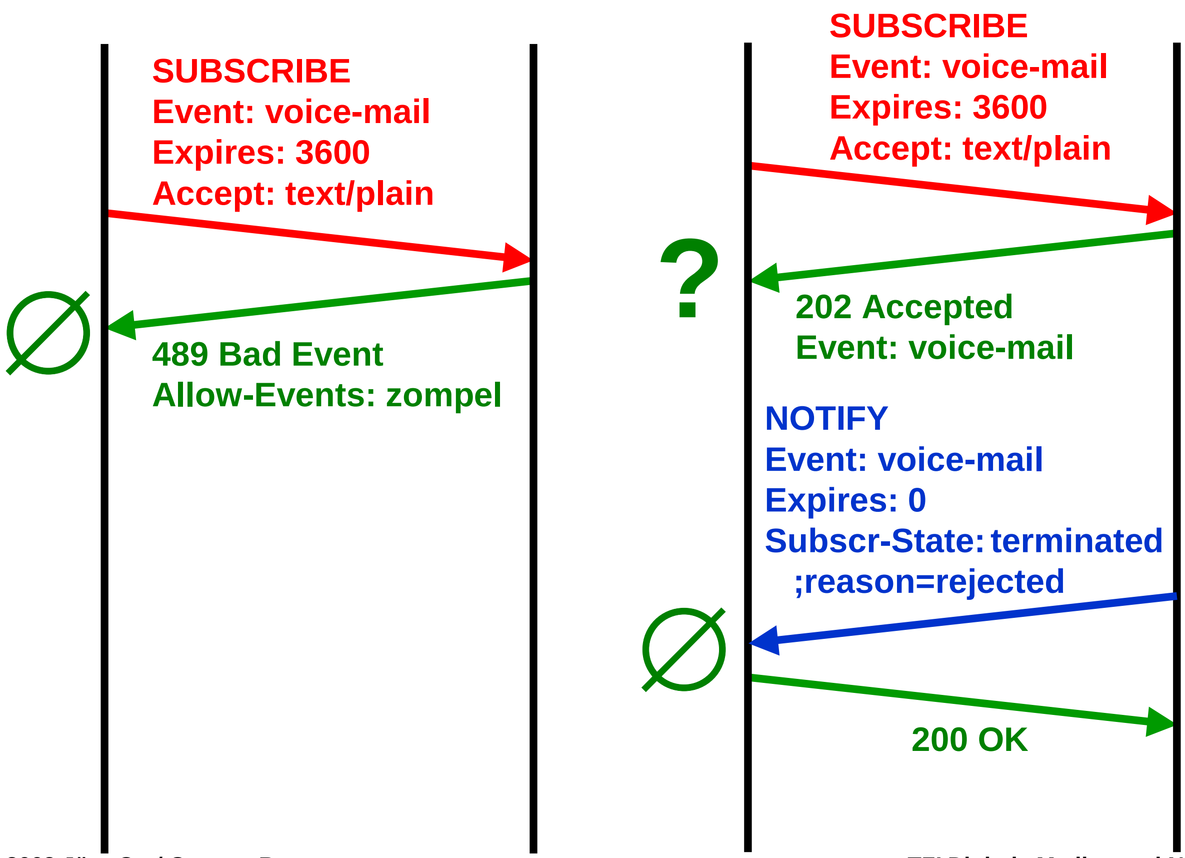
NOTIFY

- **NOTIFY carries state information in message body**
  - Format depending on Accept: header from SUBSCRIBE
  - Full state vs. state deltas
  - Subscription-State: active/pending/terminated
- **NOTIFY indicates acceptance or rejection of SUBSCRIBE**
  - If no immediate response could be supplied
- **NOTIFY may be used to terminate subscriptions**
  - Initiated by the notifier
  - Includes reason code parameter
- **NOTIFY may be sent without SUBSCRIBE**
  - Implicit subscription
- **Response to NOTIFY**
  - 200 OK vs. 481 Subscription does not exist
- **Notification rate: risk of network congestion**
  - Handling to be specified in event packages

Example: Successful SUBSCRIBE



Example: Unsuccessful SUBSCRIBE

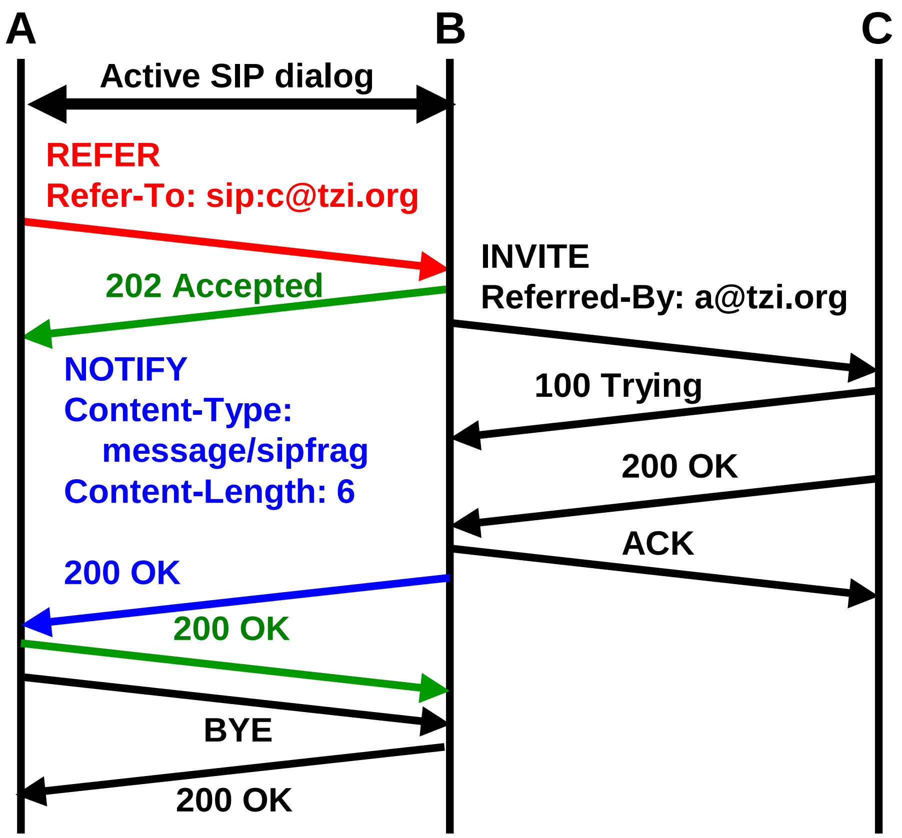


REFER Method

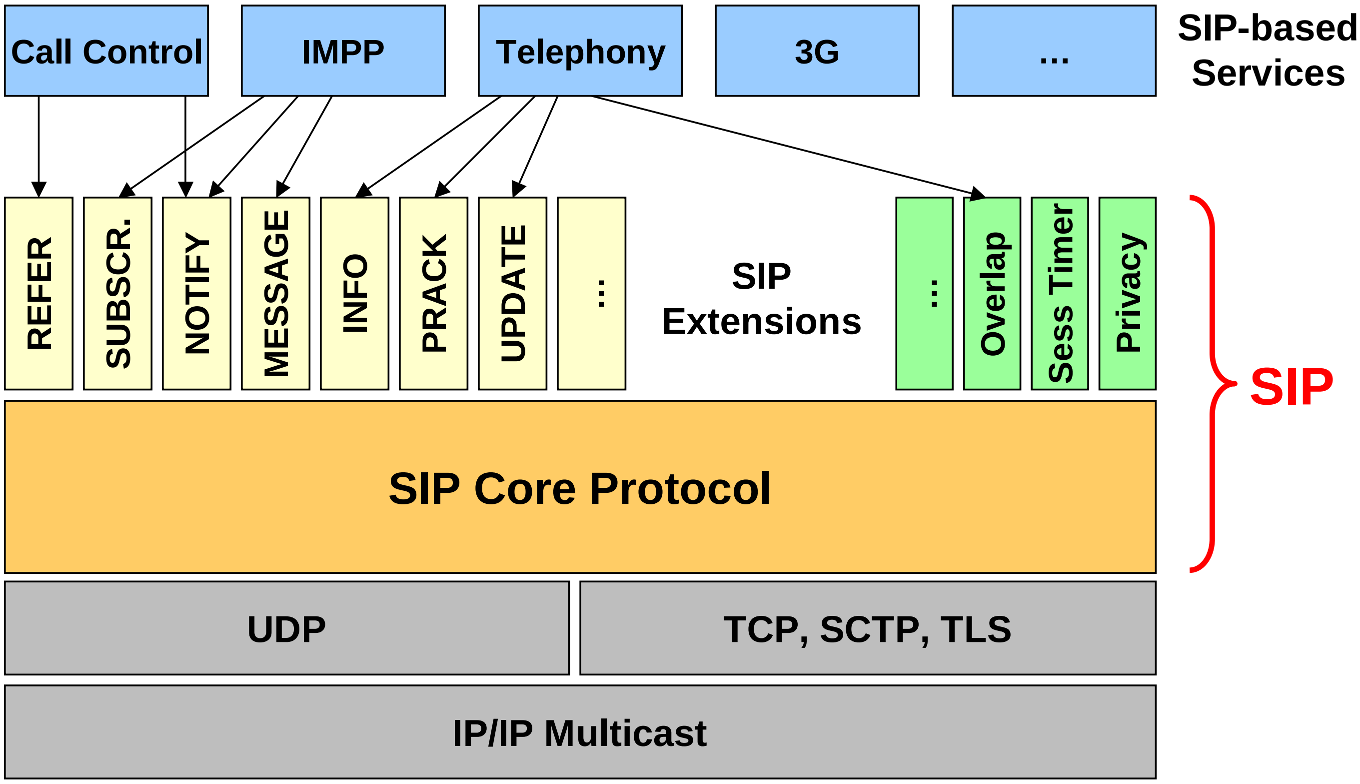
- Original motivation: Call Transfer
- General idea: make a SIP entity contact a third party

- Originator sends REFER method to peer
  - indicates *refer target* in Refer-To: header
- Referred party accepts or declines immediately
  - 202 Accepted or uses some SIP error code
  - Accepting establishes an *implicit subscription* on the new dialog
- Contacts *refer target* in an entirely independent dialog
  - May indicate who caused the call in Referred-By: header
  - New INVITE – 200 OK – ACK sequence, unrelated
- Reports outcome of new call to originator
  - (exactly) one NOTIFY
  - Message body contains SIP message fragments from new call, e.g.
    - 200 OK 486 Busy Here 503 Service unavailable

REFER Example



SIP Service Creation Model



Tutorial Overview

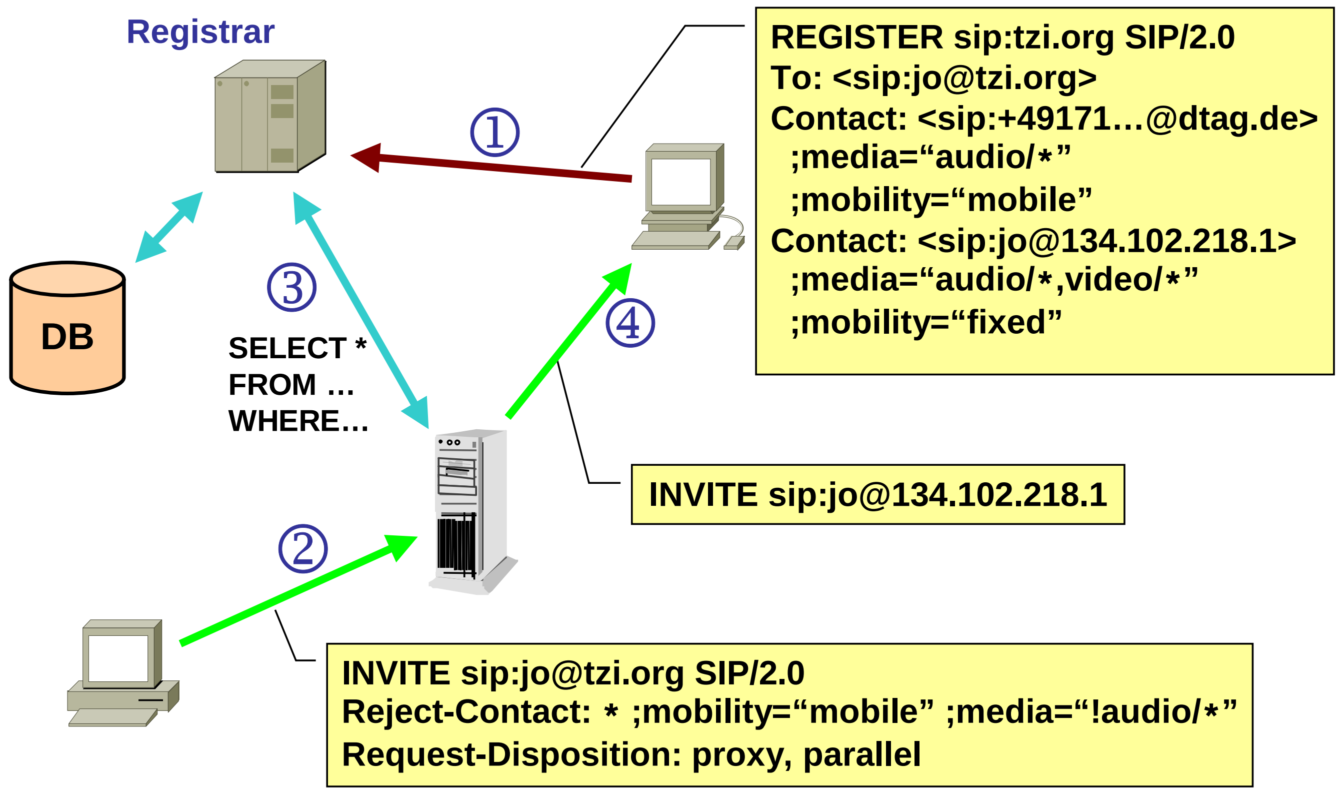
- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation ← You are still here
- SIP in Telephony and Deployment Issues
- SIP in 3GPP

Caller Preferences/Callee Capabilities

- **Caller may provide preferences for request routing**
  - URI-based Proxy or redirect
  - Refuse particular URIs
  - Forking
  - Recursive search
  - Parallel or sequential search

- **SIP-Extensions**
  - Request-Disposition: Request routing in SIP servers
  - Accept-Contact: Change address ordering
  - Reject-Contact: Reject particular URIs
  - Additional parameters for Contact:-headers

Preference/Capability Matching



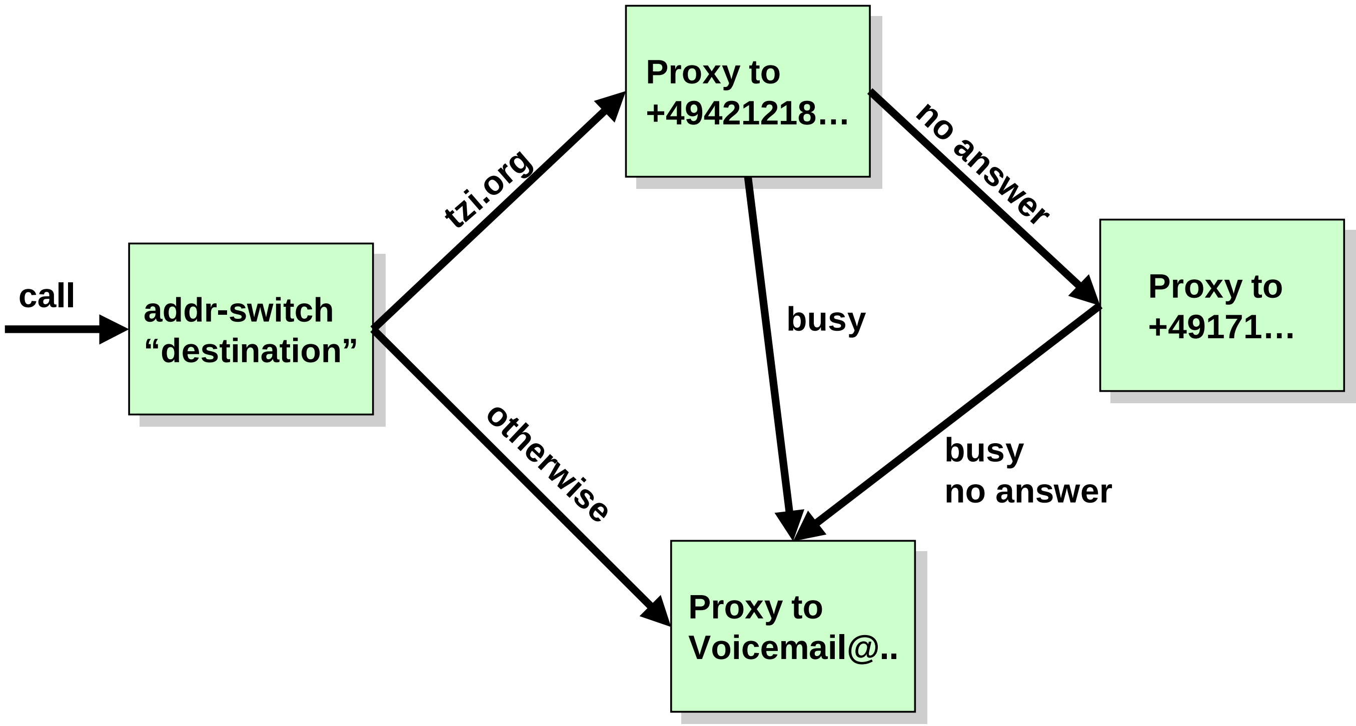
Call Processing Language (CPL)

- Per-user configuration of server behavior
  - Specification of call-related actions
  - No side-effects
  - No loops or recursion
- Graphical representation, stored as XML document
- CPL scripts are associated with user's URI
  → registration
- Design goals
  - Light-weight: minimal use of server's resources
  - Safety: do not break running server
  - Extensibility: add features without breaking existing scripts

Language Features

- Server provides information about incoming message
  - Destination
  - Originator
  - Caller preferences
  - Contents of several important header fields
  - Media description
  - Security parameters
- CPL scripts can specify several actions to take
  - Reject, redirect or proxy incoming message
  - Set timeout values for actions
  - Context-dependent choice of different actions
  - Perform location lookups

Example CPL Script



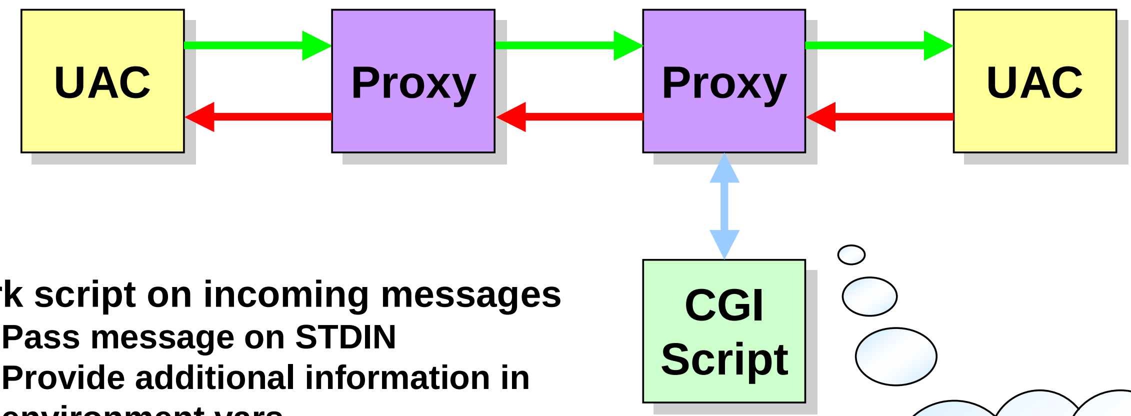

```
<?xml version="1.0" ?>
<cpl>
  <subaction id="vm">
    <location url="sip:voicemail@dmn.tzi.org">
      <proxy />
    </location>
  </subaction>
  <incoming>
    <address-switch field="destination" subfield="host">
      <address subdomain-of="tzi.org">
        <location url="tel:+49421218…">
          <proxy timeout="10">
            <busy> <sub ref="vm" /> </busy>
            <noanswer>
              <location url="tel:+49171…">
              […]
            </noanswer>
          </proxy>
        </location>
      </address>
    </address-switch>
  </incoming>
</cpl>
```

Service Creation

- **Flexible processing of SIP messages in *proxy***
  - Rapid development of supplementary services
  - Pass incoming requests to external script for processing

→ **SIP Common Gateway Interface (CGI)**
  - Adaptation of HTTP CGI
  - Support SIP's idiosyncrasies:
    - Transport protocols UDP, TCP
    - Central role of proxy servers
    - Persistent transaction state
    - Registrations
    - Request forking, recursive search, timeouts
    - …

SIP CGI Architecture

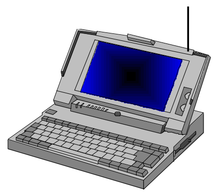


- Fork script on incoming messages
  - Pass message on STDIN
  - Provide additional information in environment vars
- Process commands from script's STDOUT
  - Message forwarding
  - Create new messages
  - Add/delete message headers
- Use cookies to preserve state information

Script is invoked for subsequent messages of same transaction only if explicitly requested.

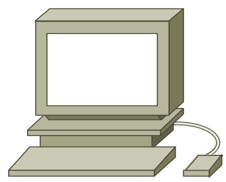
SIP for Presence and Instant Messaging

A Role for Presence in "VoIP"

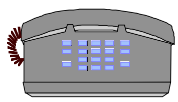


Awareness of other users: availability, location, …

- **To perform existing functions**
  - User location, call routing, follow-me, ...



- **To improve existing services**
  - Call completion ratio
  - Indicate availability



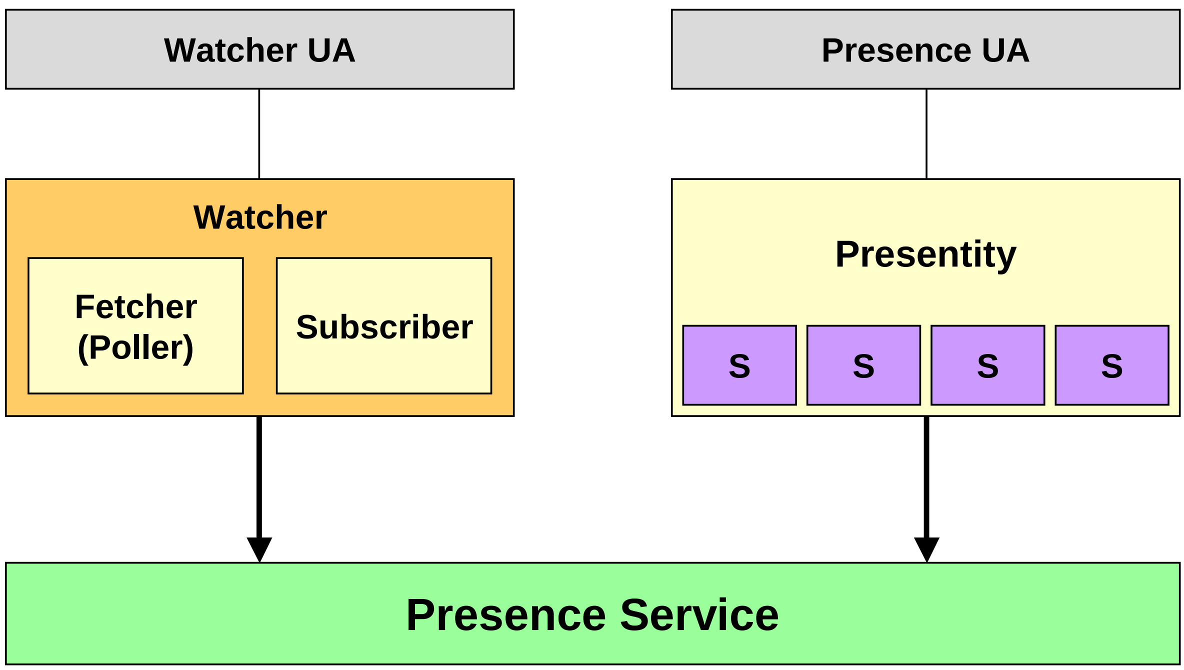
- **To enable new services**
  - Presence per se: Simplify meeting people
  - Messaging per se: "SMS"
  - Presence and Messaging as basis for other applications
  - Location-based services

- **To rescue the "VoIP" industry…**

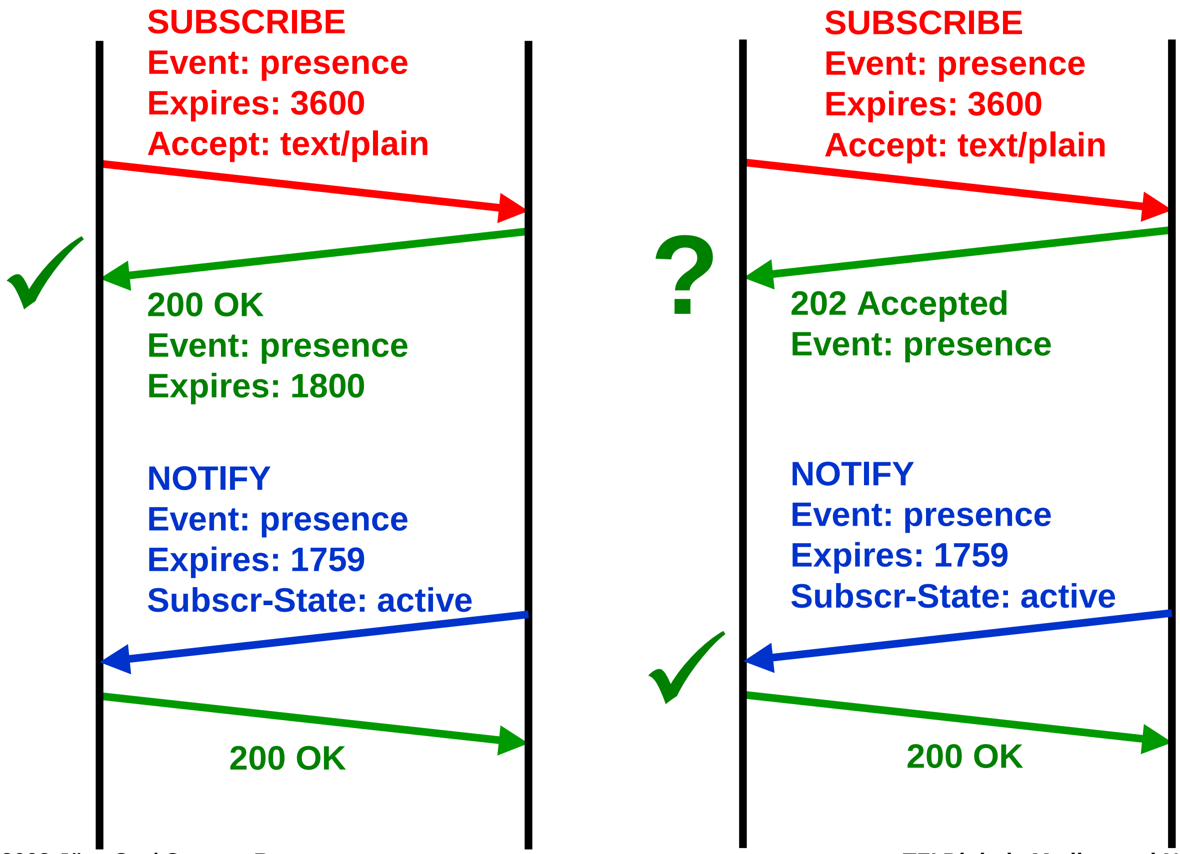
SIP: Personal Presence and Instant Messaging

- "Buddy Lists + Chat"
- Idea: re-use SIP infrastructure
  - Maintain user locations
  - Route messages
  - Contact users
- SIP Event package for "presence"
  - SUBSCRIBE / NOTIFY fit well
  - Define presence format (cpim+xml)
    - shared with IMPP group
- Define a new method for Instant Messaging
  - MESSAGE
  - Define basic message content format (text/plain)

Presence Model



Example: SIP for Presence



SIP Presence Notification

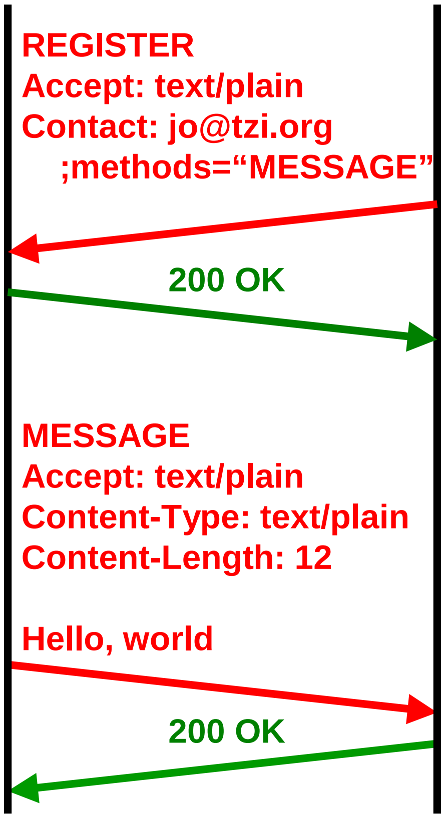
```
<presence ... entity="pres:jo@tzi.org">
  <tuple id="mobile-im">
    <status>
      <basic>open</basic>
    </status>
    <contact priority="0.8">im:jo@sms-gw.tzi.org</contact>
    <note xml:lang="en">Don't Disturb Please!</note>
    <note xml:lang="fr">Ne dérangez pas, s'il vous plaît</note>
    <timestamp>2003-01-14T10:49:29Z</timestamp>
  </tuple>
  <tuple id="interactive-mm">
    <status>
      <basic>closed</basic>
    </status>
    <contact priority="1.0">sip:jo@tzi.org</contact>
  </tuple>
  <note>I'll be in Paris next week</note>
</presence>
```

SIP for Instant Messaging (IM)

- UAs may send and receive messages
  - Similar model to Presence
- Receivers indicate support when registering

  Contact: sip:jo@tzi.org;methods="MESSAGE"

- Senders just send
  - Use sip: or im: URLs
  - Congestion control is important!
- For longer "chat sessions"
  - Create a persistent IM session
    - Just another media type
  - May not be able use TCP or TLS
  - Use SIP-based session instead?
    - Lightweight SIP just as transport protocol



(Issue:) Security

- Authentication of (Subscription) Requests
  - Standard SIP mechanisms: 401/407 responses
- Authorization: Users must stay in control of subscriptions
  - May subscribe to their own subscription state ("watcher info")
  - Receive notifications for every subscription attempt
  - May authorize each subscription
  - May cancel existing subscriptions
  - May retrieve lists of subscribers
- Authentication of Instant Messages
  - May include contents to be automatically acted upon
  - User / system needs to validate originator
- Meta-issue: end-to-end authentication using PKI…

Tutorial Overview

- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP

You are here

SIP for Telephony

…yet another set of SIP services…

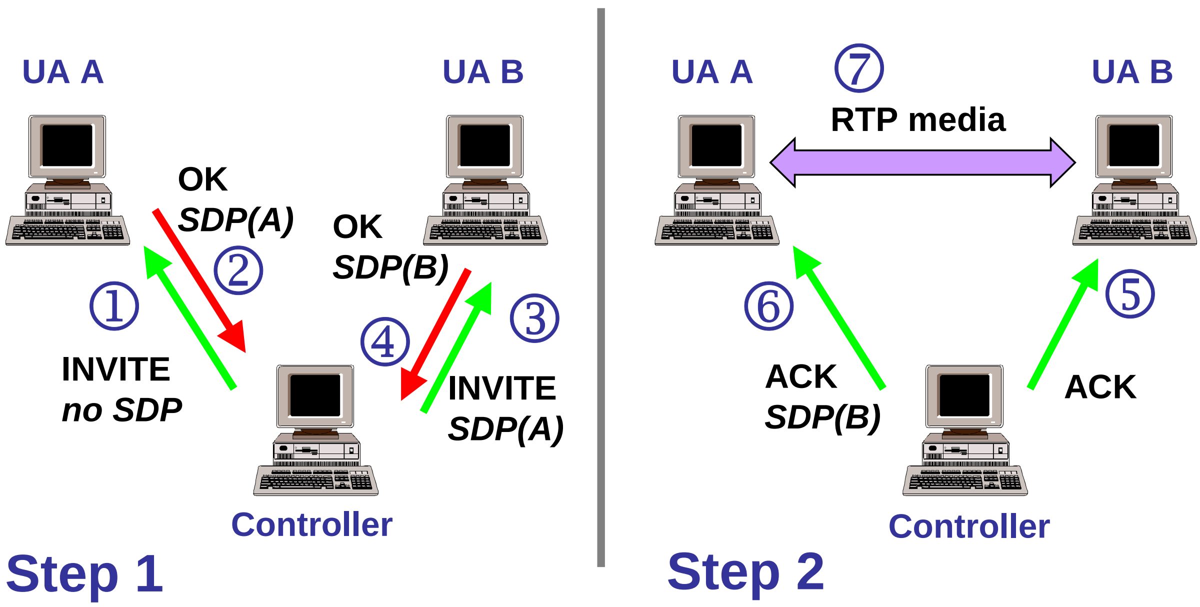
"Supplementary Services"

- **Call Diversion**
  - Intrinsic support (301/302 redirection)
- **Call Park & Pickup**
  - Distributed state of user agents; embed call state in cookies
- **Call Hold and Retrieve**
  - User agents sends re-INVITE to mute other parties (a=inactive)
- **Call Waiting**
  - Implemented in endpoints
- **3rd party call control**
- **Call transfer**
  - Use new REFER method to indicate a party to place a call to
- **Conferencing**
- **Message waiting**
  - Specific event package: message-summary

TZI Technologie-Zentrum Informatik
Universität Bremen

Third-Party Call Control

- **"Click-to-dial"**
- **Conference bridge initiation**



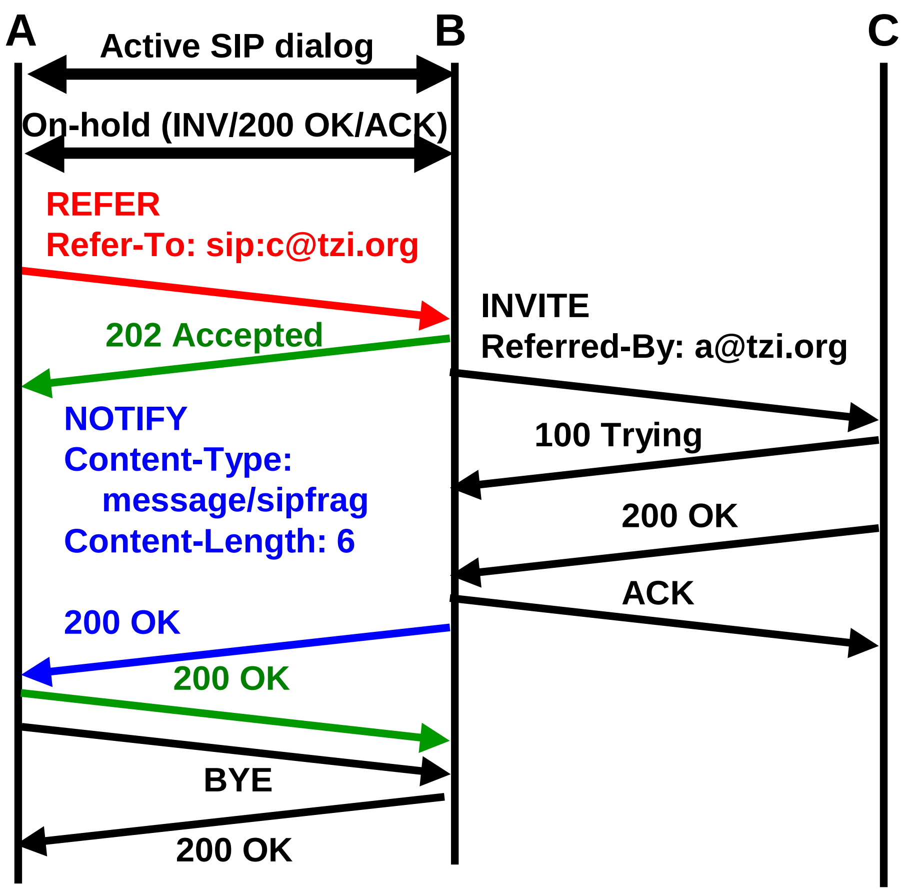
Call Transfer

- **Part of Call Control Framework**

- **Uses basic SIP protocol features and extensions**
  - **REFER** method to invoke another (INVITE) transaction
  - **NOTIFY** (with implicit subscription) to indicate success or failure
  - New **Replaces:** header to indicate substitution of an existing call

- **Supports numerous variants**
  - Attended
  - Unattended
  - Intermediate three-way calling
  - Optional protection of transfer target

Simple Unattended Transfer



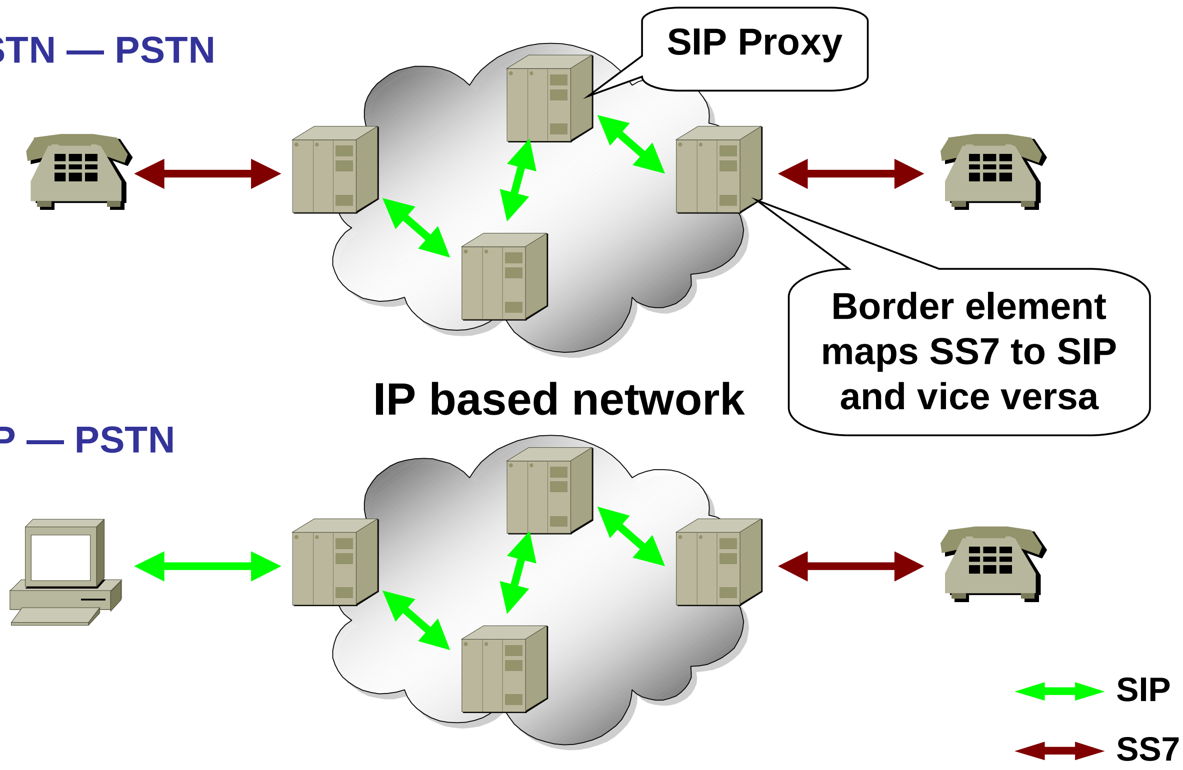
Message Waiting Indication

- Asynchronously notify endpoint(s) about messages
  - Voice, video, image, text, ...
- Define a new SIP event package
  - Subscribe to one or more mailboxes
  - Results from many sources may be merged
- Content-Type: application/simple-message-summary
  - General indicator for new messages
  - Message type followed by new/old and (new-urgent/old-urgent)
  - Encoding as plain text
- Example:

```
Messages-Waiting: yes
Voicemail: 4/8 (1/2)
Email: 238/42116 (0/1)
```


Interfacing to the PSTN

1. PSTN — PSTN
2. SIP — PSTN

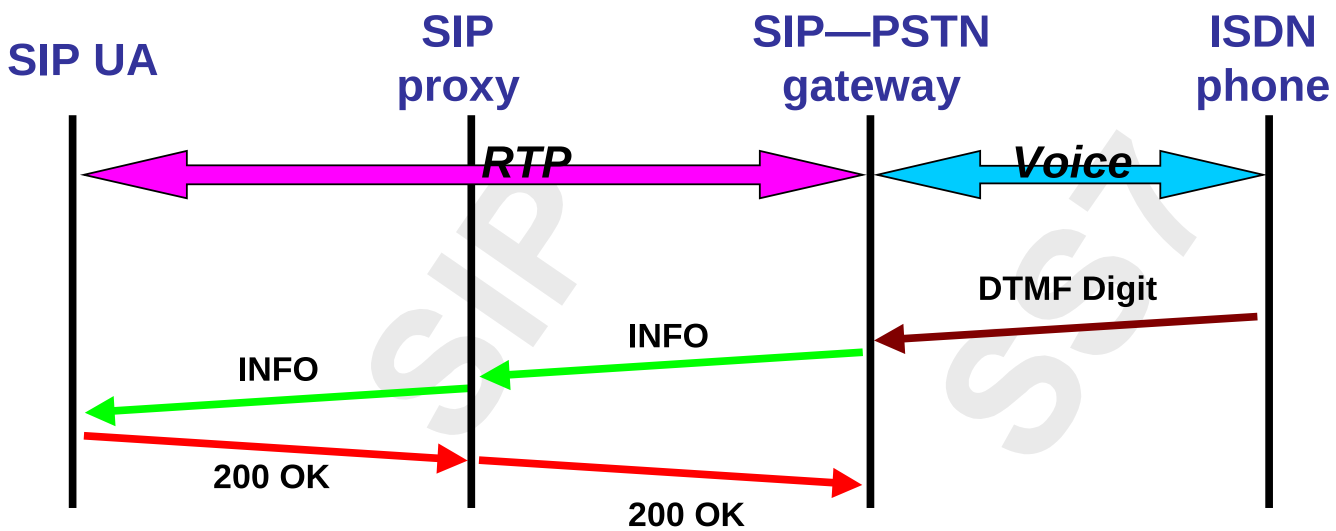


SIP for Telephony (SIP-T)

- **Interface to the PSTN**
- **Preserve feature transparency**
  → transport SS7 information (ISUP MIME type)
  - **Eventually convert between different ISUP versions**
- **Provide enough routing information to find callee**
  → (partially) translate ISUP to SIP
- **Support for tel:-URLs to indicate Called Party Number**
- **Additional information during call**
  → INFO method (RFC 2976)
- **Is tunneling a good thing?**

INFO Method

- **Transmit application-layer information during call**
  - **Use SIP signaling path of current session**
  - **Information is carried in message headers or body**
  - **No change of (SIP-related) call state**



183 Session Progress Message

- INFO not applicable *before* call is established
- ISUP mapping requires inband data prior to final response

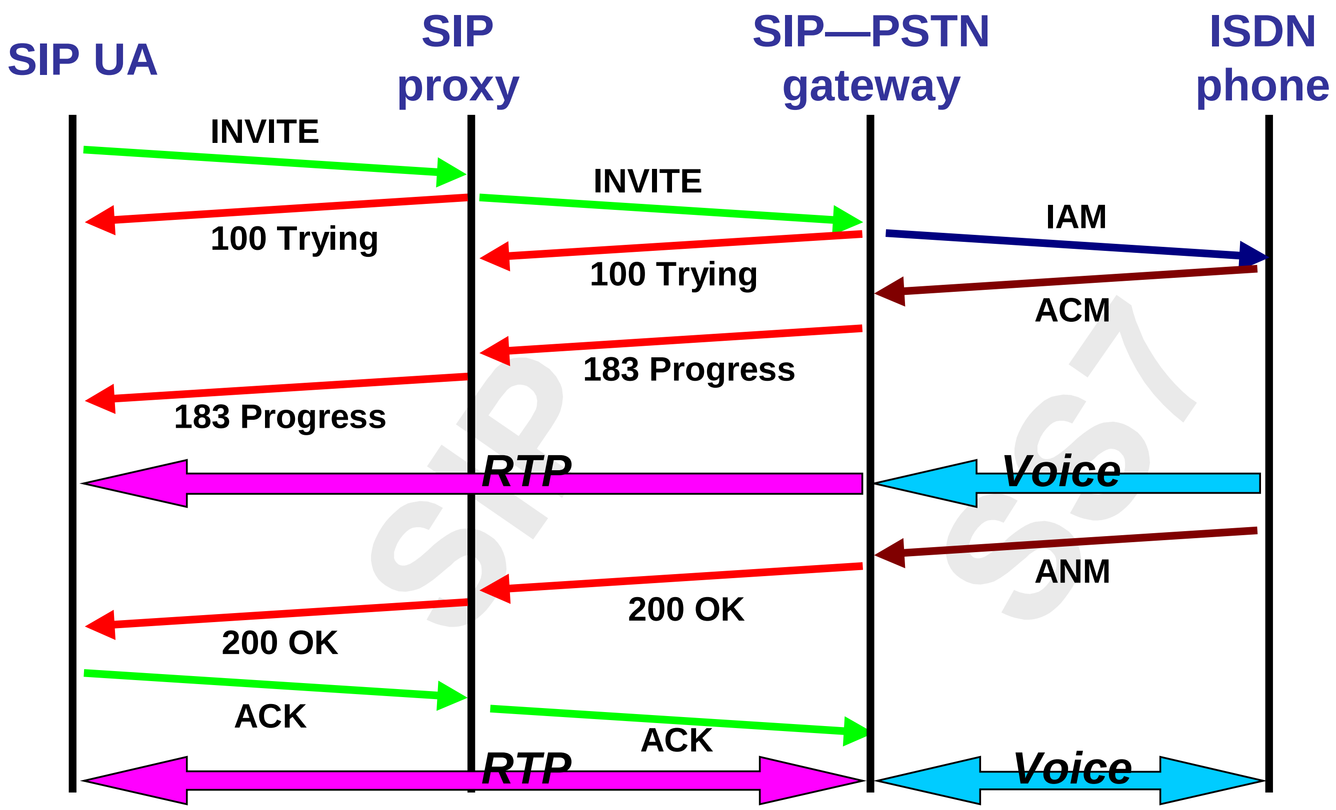
→ provisional response 183

- Additional information in message body
- Session:-header
  - Indicate reason: media, qos, security
- Use 100rel, if need be

Early Media Support

- Early media during call setup (SIP INVITE)
  - One-way transmission to report progress
  - Announcements, specific dial tones, …
  - No charge
  - Inhibit local alerting at calling user agent
- Problem: Media negotiation
  - Send SDP message in provisional response
    → *fast setup*
  - Create SDP from initial INVITE's capability set
    → What if not suitable for early media session?
  - Calling UA will establish *recvonly* media session
    → Cannot decline session or change codecs

SIP—PSTN Call With In-band Alerting



Sample SIP Phone Functionality

- Somewhat resemble a phone
  - More or less futuristic design
  - Two-line to color graphics display
  - Sometimes line power
- Basic SIP functionality
  - Registrations, voice calls (G.711, G.729, G.723, …)
- Expected “supplementary services”
  - Address book, short dials
  - Several “lines”, speaker
  - Call hold, call transfer, conferencing, …
  - N-way conferencing for a few participants (local mixing)
- Some kind of CTI support
  - APIs, 3rd party call control, …
- Autoconfiguration: DHCP, (t)ftp, …
- HTTP server for manual user configuration
- Sometimes web browsers

Some SIP Phones

Tutorial Overview

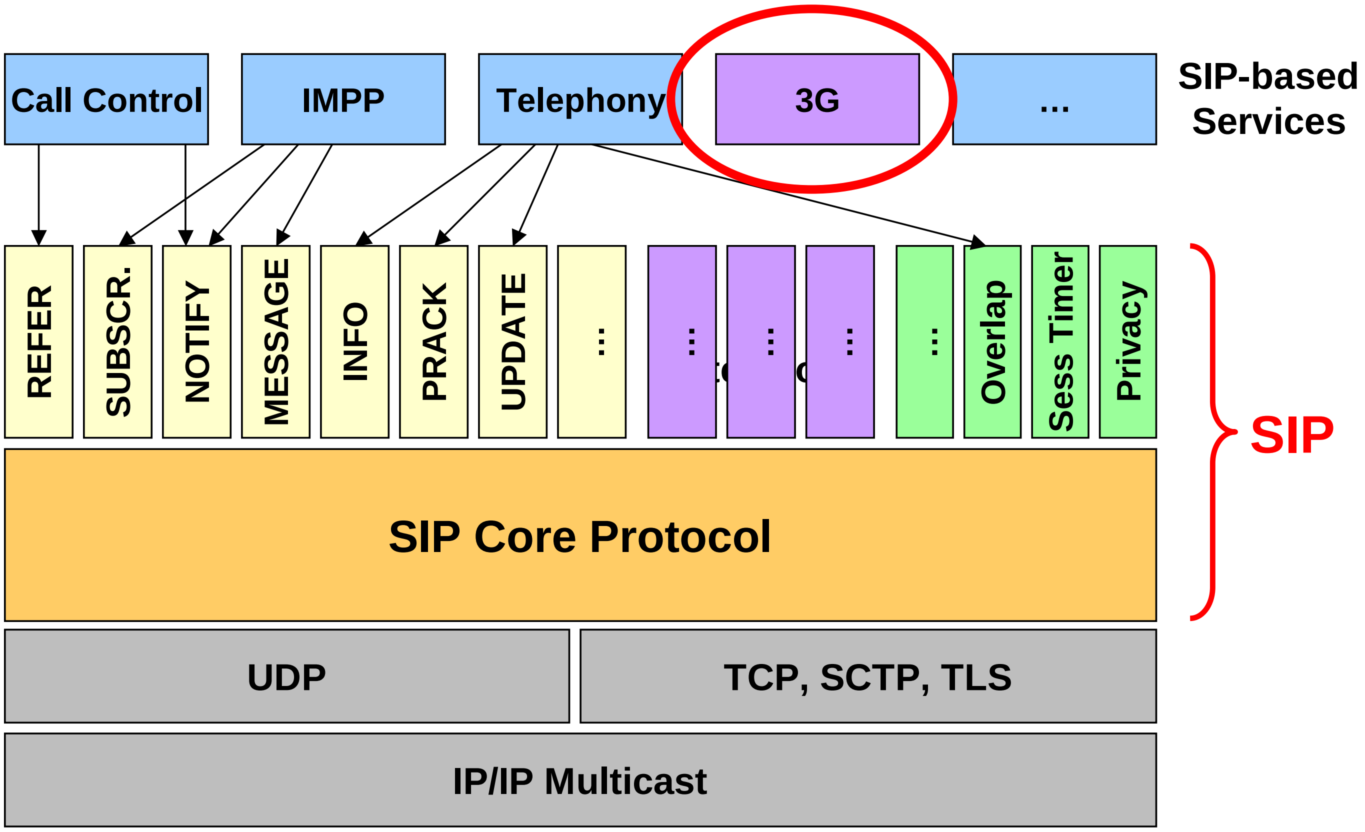
- Internet Multimedia Conferencing Architecture
- Packet A/V Basics + Real-time Transport
- SIP Introduction, History, Architecture
- SIP Basic Functionality, Call Flows
- SIP Security
- SIP Service Creation
- SIP in Telephony
- SIP in 3GPP ← You are here

SIP and 3G

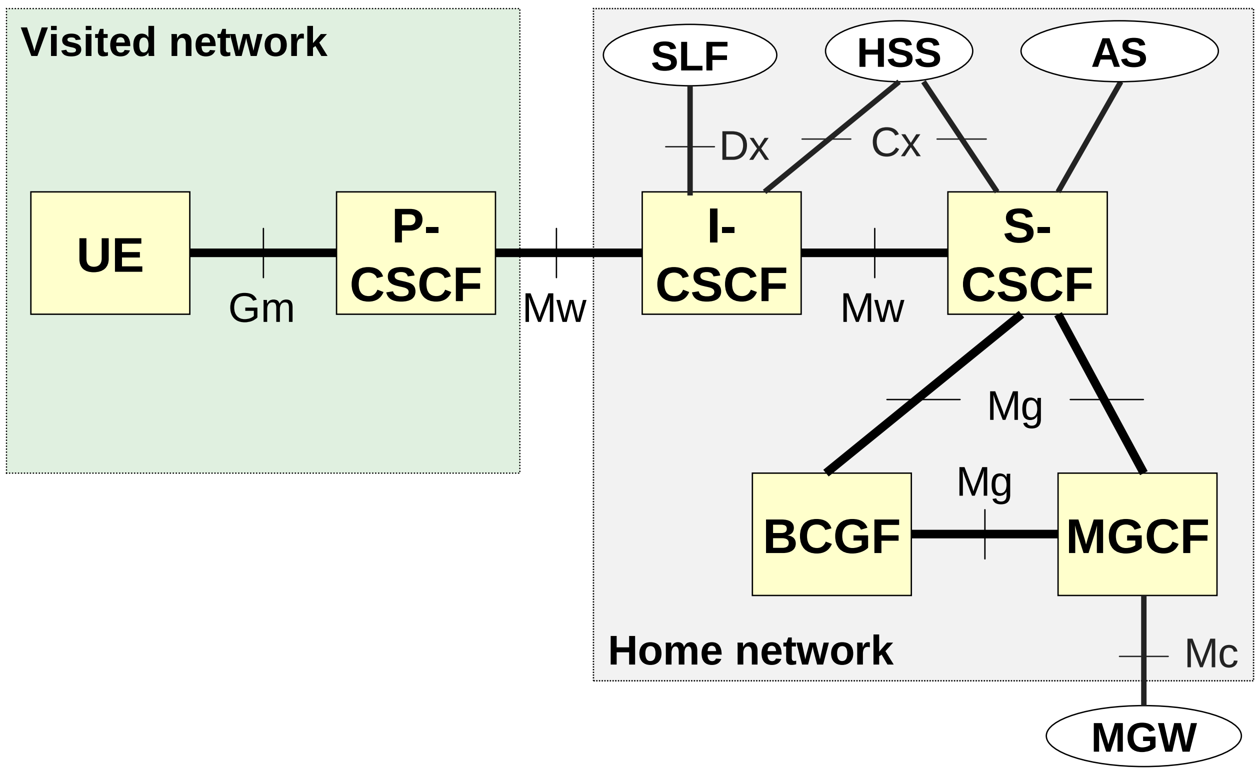
SIP and 3GPP

- 3G networks offer two modes of operation:
  - circuits and packets
- Multimedia functionality based on packets
  - IP Multimedia Subsystem (IMS) in the Core Network (CN)
  - (exception: H.324 used for video telephony)
- IMS uses SIP for signaling
  - one piece out of a number of IETF protocols
- In the long run, all services shall converge to IP
- Release 5: SIP-based multimedia calls
- Release 6: Further SIP services to come (e.g. Presence, IM)

SIP and 3G



SIP-related Components in 3GPP



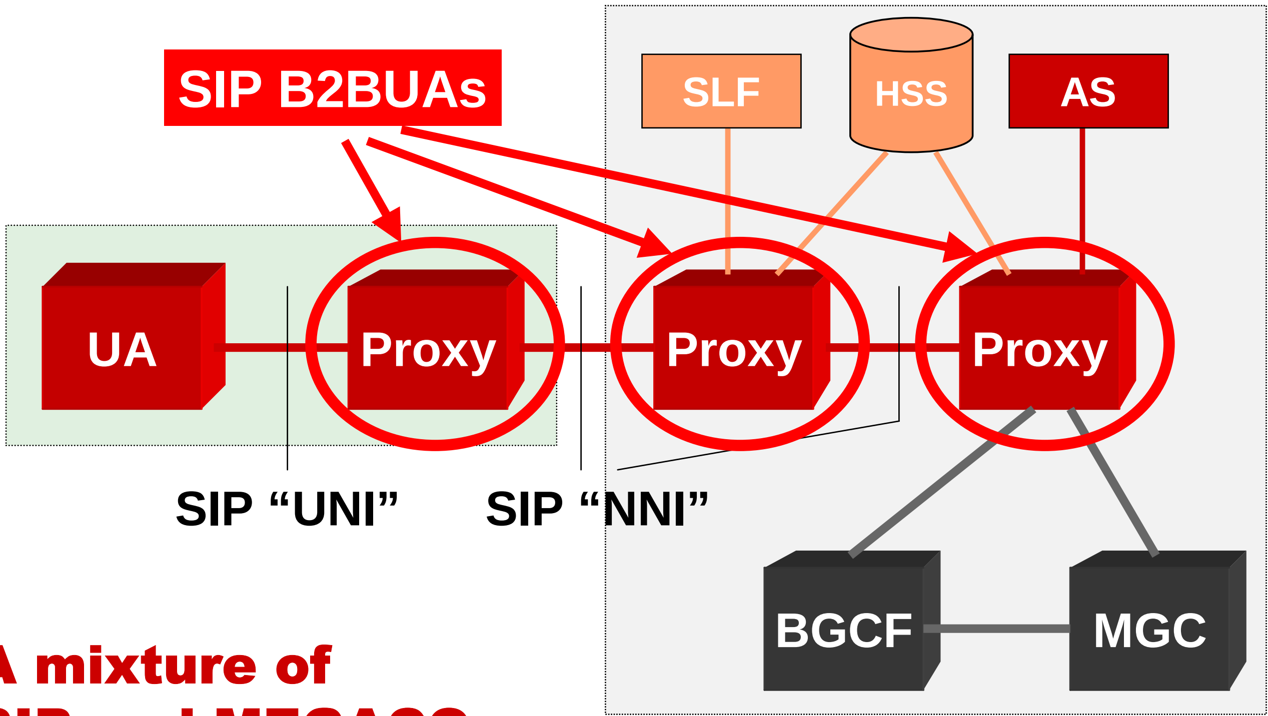
A Collection of Acronyms

- UE User Equipment

- P-CSFC Proxy Call Session Control Function
- I-CSFC Interrogating Call Session Control Function
- S-CSFC Serving Call Session Control Function
- HSS Home Subscriber Server
- AS Application Server
- SLF Subscription Locator Function

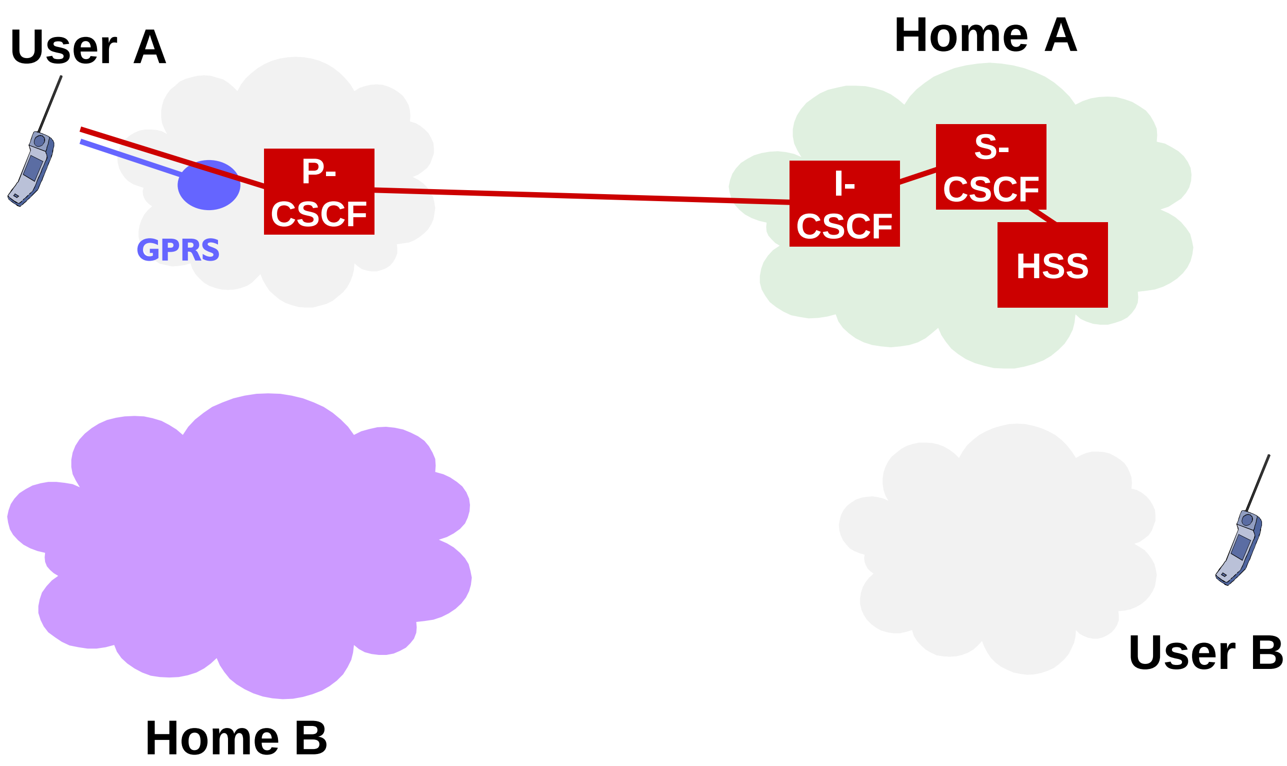
- BGCF Breakout Gateway Control Function
- MGCF Media Gateway Control Function
- MGW Media Gateway

SIP Components in 3GPP

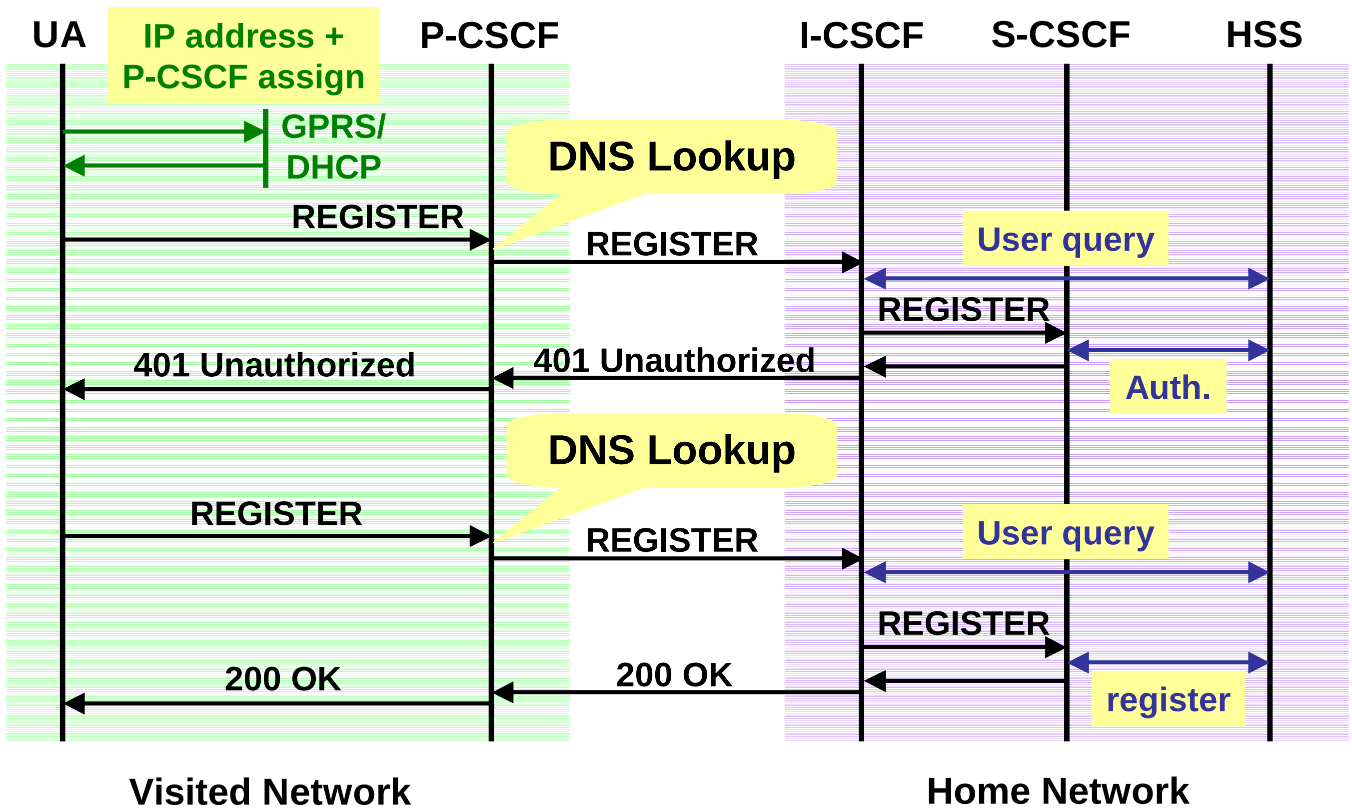


A mixture of SIP and MEGACO…

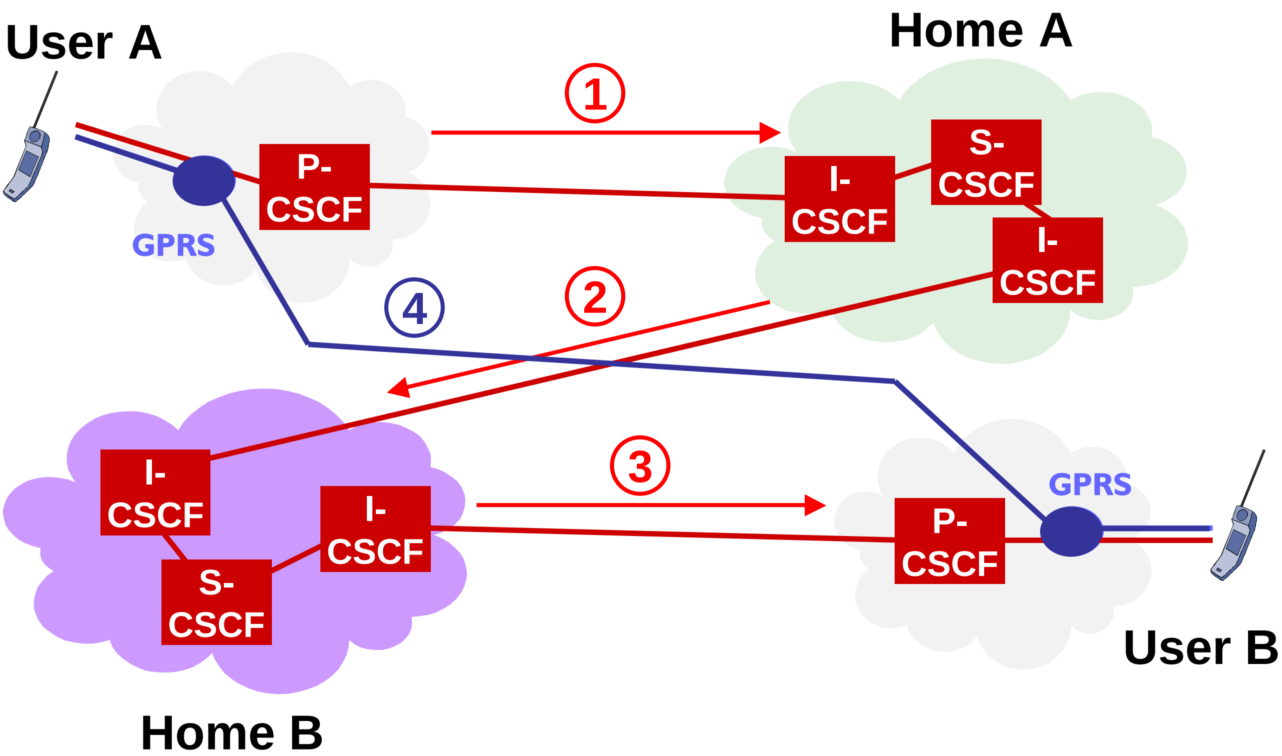
3G Roaming Scenario: Registration



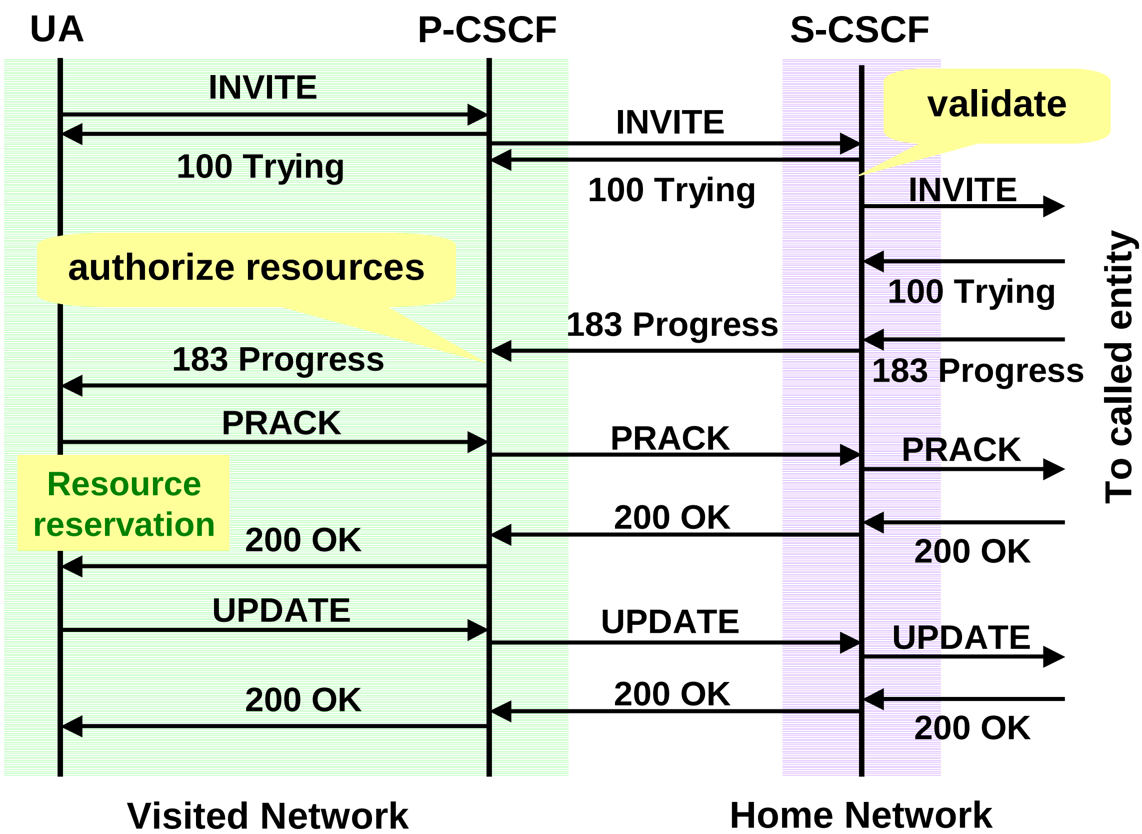
SIP Registration of a Mobile Node



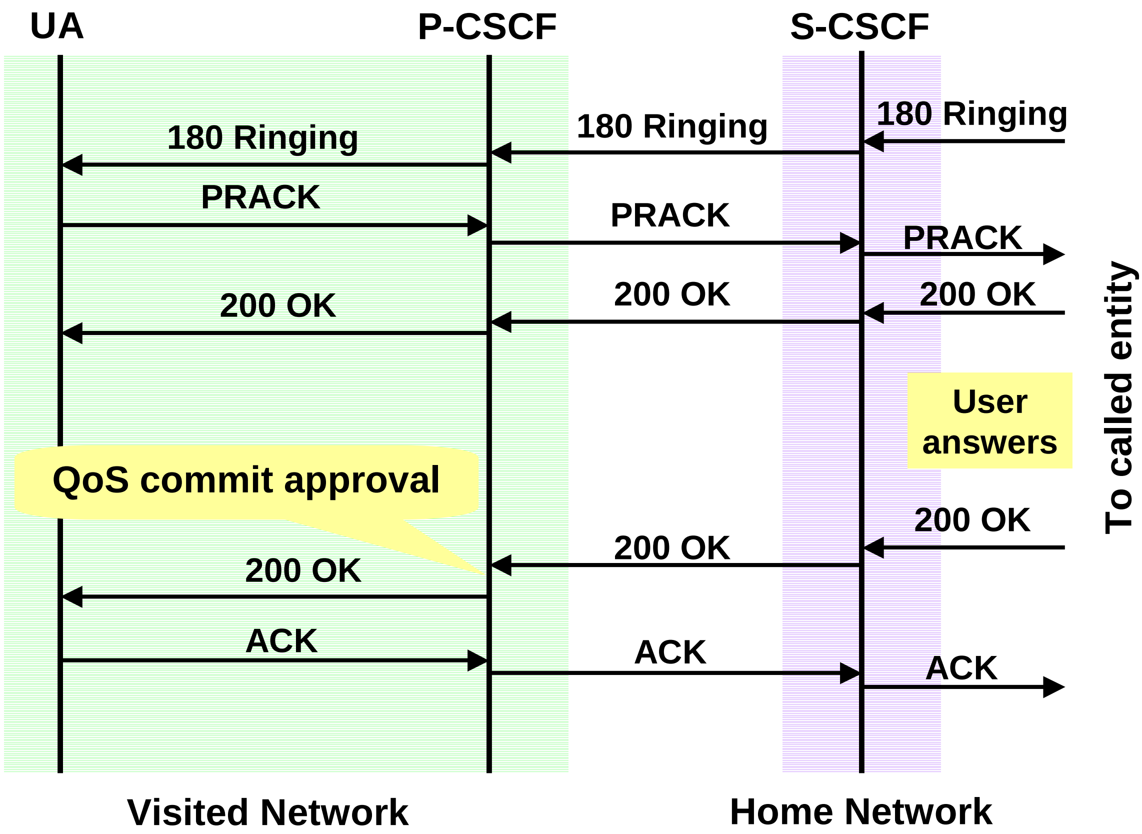
3G Roaming Scenario: Simple Call



Simple SIP Call: Caller Side (1)



Simple SIP Call: Caller Side (2)



Some further 3G Signaling

- Called side: pretty much the same
- Call termination by either endpoint: straightforward
  - Simple two-way handshake: BYE – 200 OK
  - Release associated GRPS resources
  - Accounting in S-CSCF/HSS: e.g. generate Call Detail Record (CDR)
- Call termination by “network”
  - S-CSCF generates SIP BYE requests to called and calling UA
- Terminal de-registration by user
  - Simple two-way SIP handshake: REGISTER – 200 OK
- Terminal de-registration by “network”
  - Uses SIP NOTIFY mechanism to inform the UA

SIP Features used in 3G

- SIP Base Spec (RFC 3261)
- Reliable Provisional Response (PRACK) (RFC 3262)
- Session Description Protocol (RFC 2327)
- SDP Offer/Answer Scheme (RFC 3264)
- SUBSCRIBE / NOTIFY method (RFC 3265)
- SDP extensions for IPv6 (RFC 3266)
- UPDATE method (RFC 3311)

- Resource reservation extensions (“manyfolks”)
- Authentication and privacy

… among others … + … a number of extensions …

Service Routing Extensions in 3G

- P-CSCF and S-CSCF perform central functions for UAs
  - Calls need to pass (at least) through both of them
  - to ensure all services are available
- Configure P-CSCF as outbound proxy
  - Address obtained during link layer initialization
- Record proxy chain to be traversed during registration
  - SIP extension for recording proxies to be traversed en-route
  - Path: header
  - Used together with loose source routing from S-CSCF to UA
- Learn about S-CSCF in REGISTER response
  - SIP extension for service routing
  - Service-Route: header
  - Used together with loose source routing (as defined in RFC 3261)

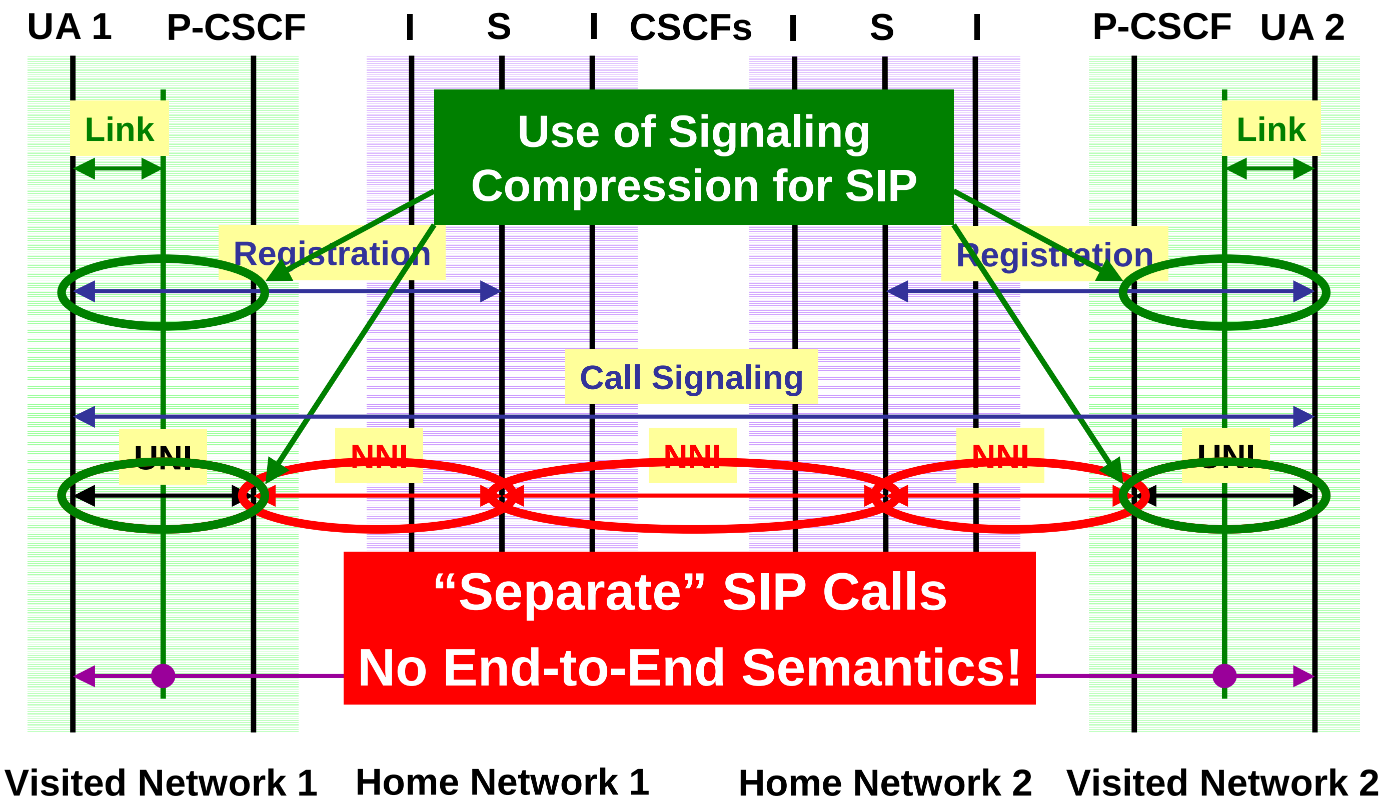
Some SIP Security in 3G

- **Basic Approach: closed network infrastructure**
  - Transitive trust (and secure communications) within the network
- **SIP Digest Authentication for registration**
- **Stateful network infrastructure**
  - Only registered users can place calls
  - Ingress nodes (P-CSCFs) validate UA requests
  - S-CSCF additionally check validity per request
- **P-Asserted-Identity: header**
  - May contain network-generated (anonymized) identifier
- **Privacy: header to select CLIP/CLIR**
  - User id may be removed per user request when leaving the network

Further 3G Extension Headers

- **P-Associated-URI:**
  - URIs allocated by service provider returned in REGISTER response
- **P-Called-Party-ID:**
  - To indicate original request URI to the called party
- **P-Visited-Network-ID:**
  - Conveys the identifier of the currently visited network
- **P-Access-Network-Info:**
  - Reports information about the currently used access radio link
- **P-Charging-Function-Addresses:**
  - Indicates where to report charging information (e.g. CDRs) to
- **P-Charging-Vector:**
  - Enable correlation of charging information across the network

Summary: Signaling and Media Flows in 3G



Service Creation in 3GPP

- Standardized functional building blocks in the network
  - services to be created on top of those
  - Partially standardized services
- "APIs" for service providers
- SIP for interfacing to Application Servers
- Service creation in the network
  - "closed" environment
  - potential for limited outside access
- BUT: No end-to-end signaling
  - Header fields may be removed by P-CSCF
  - New methods may not be passed through
- No / limited end user or third party innovation possible

Conclusion: SIP and 3G

3G is partially built the *old* way – the telephony way…

- Breaks the (S)IP end-to-end model
  - built-in limitation of innovation
- Enforces net-centric service creation
  - built-in protection against newcomers
- Closed architectural model
  - built-in gateways even to external SIP devices

(obvious economic motivations for the above)

- Many boxes, many lines, many interfaces
  - High degree of complexity? (! KISS)

Further Information

http://www.ietf.org/html.charters/sip-charter.html
http://www.ietf.org/html.charters/sipping-charter.html
http://www.ietf.org/html.charters/simple-charter.html
http://www.ietf.org/html.charters/impp-charter.html

http://www.softarmor.com/sipwg
http://www.softarmor.com/sipping

http://www.cs.columbia.edu/~hgs/sip

http://www.3gpp.org/
ftp://ftp.3gpp.org/