Introduction: Who We Are

“Technologie-Zentrum Informatik”

- Founded 1995
- Non-Profit organization
- Part of the Dept. of Computer Science
  Universität Bremen, Germany
- ~100 Scientists (11 University Professors)
- Main focus is on technology transfer
  → Tech consulting, studies, workshops,
    cooperative projects, …
- 2002 turnover > € 4 Mio
Research Group for Computer Networks

Architectures, Protocols and Interfaces for computer-supported communication and collaboration

◆ Research Activities
  – Multimedia-Conferencing, Media Transcoding
  – Data Distribution (Unicast/Multicast)
  – Content Transformation, XML Technologies
  – Call Signaling, Presence
  – Security in distributed systems
    (focus on secure real-time multimedia)

◆ Standardization Activities
  – IETF: MMUSIC, RoHC, (SIP)
  – TERENA: Task Force “Mobility”, “VoIP Cookbook”
  – Past: ITU-T (H.323)

→ Continuous work on MM Conferencing from early 1990s

Prerequisites

◆ Basic knowledge about computer networks
  – packet switching, circuit switching
  – Some specific networking technologies
    such as Ethernet, serial, ATM

◆ Familiarity with IP networks
  – Internetworking “Architecture”, Internet Principles
  – Network Layer Protocols: IP, ICMP, ...
  – Transport Protocols: UDP, TCP, ...
  – Service Protocols: DNS, DHCP, ...
  – Security Basics

◆ Inclination to ask questions...

◆ Stamina :-)
This Week…

<table>
<thead>
<tr>
<th>Today</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIP Tutorial</td>
<td>Status and Impact of SIP</td>
<td>SIP for small enterprise systems</td>
<td>Instant Messaging &amp; Presence</td>
</tr>
<tr>
<td>SIP Deployments</td>
<td>SIP Security</td>
<td>Public Safety</td>
<td></td>
</tr>
<tr>
<td>The Real-World Impact of SIP</td>
<td>Mobile Networks</td>
<td>Application Building</td>
<td></td>
</tr>
<tr>
<td>SIP – PSTN Interconnection</td>
<td>SIP Services</td>
<td>Charging &amp; Billing</td>
<td></td>
</tr>
</tbody>
</table>

This Tutorial…

◆ Proposed Schedule:

- **1000 – 1100**  
  IP Multimedia, SIP Intro, Basics
- **1100 – 1120**  
  Morning Break
- **1120 – 1230**  
  Basic Call Flows, Registration, Location
- **1230 – 1400**  
  Lunch Break
- **1400 – 1515**  
  Security, SIP Building Blocks
- **1515 – 1545**  
  Afternoon Break
- **1545 – 1700**  
  Telephony, Deployment Issues, (3GPP)

- **1700 – 1730**  
  The Real-World Impact of SIP

◆ We know that there is nothing like a 20min coffee break – but please let’s try anyway…
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

IP Multimedia Applications (1)

- Packet multimedia experiments since 1980s
  - A/V tools + protocols for A/V over IP
  - Conference control protocols
- Internet broadcasting (Mbone)
- 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)

Teleconferences

◆ 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
- Call (and 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)

Traditional IETF Conferencing Concept

- Multicast-based

- Loosely-coupled conferences
  - no membership control
  - inexact information about participants
    - provided on a voluntary basis
  - security by encryption

- Public announcements and invitations
  - Convey session parameters, then get out of the way
    - Session Announcement Protocol (SAP)
    - Session Initiation Protocol (SIP)

- Conference control
  - Some need perceived; limited success over many years
Conference Establishment

Session Description

Workshop
1. Create
Descr.: SIP Tutorial
Orig.: J.Ott jo@tzi.org 327689113
Info: http://www.tzi.org/dmn/
Start: 20.01.2004 / 10.00
End: 20.01.2004 / 17.00
Media: Audio GSM 234.5.6.7/59000
Media: Video H.261 234.5.6.8/58000

2a. Disseminate
Announce Protocol
Netnews
WWW

2b. Invite
E-Mail
Invitation Protocol

3. Join

4. Media streams

IETF Conferencing Architecture

Conference Control
Audio Video
Media Streaming
Shared Apps
Session Direct.
SDP
RSVP RTP / RTCP RTSP SAP SIP HTTP SMTP
UDP SCTP TCP / TLS
IP / IP Multicast
Integrated / Differentiated Services Forwarding
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

◆ 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

◆ Issue: Variable, partially unpredictable delay…
Real-time Media over Packets (2)

1) analog input signal

2) sampled input signal
   (implicit compression)

Real-time Media over Packets (3)

3) Quantization
   (another step of implicit compression)

4) Digital data stream
5) optional further compression yields small discrete frames

6) multiple frames or samples are collected to form packets
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)

---

Jitter vs. Delay

- **Packet Arrival Probability**
- **Transmission Delay**
- **Worst Case Delay**
- **Jitter**
Real-time Media over Packets (6)

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

Media streams (RTP) ─────────────── Control Flows (RTCP)

Real-time Transport Protocol (2)

◆ 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

◆ Standard RTP packet header
  - independent of payload type
  - possibly seconded by payload header
RTP Header

- **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:”)

Payload formats defined for many audio/video encodings

◆ 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)

General principle:
◆ 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, 2431
  – RFC 2435, 2658, 2733, 2793, 2833, 2862, 3016, 3047, 3119, 3189
  – RFC 3190, 3267, 3389, 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

- V: Video
- P: Picture
- X: Additional payload-specific header
- CC: TXCT (Transmission Clock)
- M: Marker
- PT = 0: Payload Type
- Sequence Number
- Timestamp (8 KHz clock)
- Sender SSRC
- PCM Audio

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 (in RTCP)
- 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)

- **V**: Video bit stream
- **P**: Packet number
- **X**: Clock value
- **M**: Marker
- **PT = 0**: Payload type
- **Sequence Number**: Timestamp (8 KHz clock)
- **Sender SSRC**: Header information
- **SBit**, **EBit**, **I**, **V**, **GOBN**, **MBAP**, **QUANT**, **HMVD**, **VMVD**: Various fields

H.261 Bit Stream

Audio Redundancy Coding (1)

- **Audio Packets are small!**
  - have to be because of interactivity
  - avoid large packetization delay
  - packet loss primarily depends on packet rate
  - rather than packet size

- **Payloads for multiple time slots in one packet**
  - send redundant information in packet n
  - to reconstruct packets k, ..., n-1 in packet n
  - redundant information typically sent at lower quality
  - details defined in RFC 2198
  - uses dynamic payload type

- **Format specification, e.g. using SDP**
  - m=audio 20002 RTP/AVP 96 0 0
  - a=rtpmap:96 red/8000/1
Audio Redundancy Coding (2)

Packet #1 ... Packet #2

Packet #2

Primary Encoding: PCM
Secondary Encoding: GSM

Packet #3

Packet #4

Audio Redundancy Coding (3)

<table>
<thead>
<tr>
<th>Redundant 2</th>
<th>Redundant 1</th>
<th>Primary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp = 6400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PT=96</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Block “Redundant 2” (160 Bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Block “Redundant 1” (160 Bytes)</td>
</tr>
<tr>
<td>Data Block “Primary” (160 Bytes)</td>
</tr>
</tbody>
</table>

© 2004 Jörg Ott

TZI Digitale Medien und Netze
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)

<table>
<thead>
<tr>
<th>Event</th>
<th>E</th>
<th>Volume</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modulation</th>
<th>T</th>
<th>Volume</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Packet Format 1: Events

Packet Format 2: Tones
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
    - Audio Redundancy Coding (RFC 2198)

◆ **Feedback loops for senders**
  - based upon generic and specific RTCP messages
  - adapt transmission rate, coding scheme, error control, ...

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**
SIP: Session Initiation Protocol

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

History of Mbone conference initiation

- Participant location
- Conference invitation
- Capability negotiation during setup

Session Invitation Protocol
(Handley/Schooler)

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

Simple Conference Invitation Protocol
(Schulzinne)

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

SIP and Conferencing over Time…

◆ Origin: MMUSIC: Multiparty Multimedia Session Control

◆ From Invitation… to initiation, modification, and termination
◆ From Multiparty… to point-to-point-focused
◆ From Multimedia… to voice-centric

Now: Multiparty & multimedia rediscovered

But: Don’t believe in multicast (anymore)!
Timeline: 1996

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

- **22 Feb 1996**: Session Invitation Protocol (SIP) – M. Handley, E. Schooler
- **22 Feb 1996**: Simple Conference Invitation Protocol (SCIP) – H. Schulzrinne

SIP: Setup + Caps Negotiation
SCIP: Setup + Caps
Modify + Terminate

Presentations at 35th IETF, Los Angeles

- **4-8 Mar 1996**: Main Features set: TCP/UDP, Forking, Redirection, addrs
- **4-8 Mar 1996**: INVITE, CAPABILITY
- **4-8 Mar 1996**: From: To: Path:

Timeline: 1997

- **27 Mar 1997**: Draft SIP -02
  - Formal syntax
  - CAPABILITY ➞ OPTIONS
  - Path ➞ Via:
  - Ideas for Alternates:
- **31 Jul 1997**: Draft SIP -03
  - SIP URL: sip://jo@…
  - CONNECTED, BYE, REGISTER
  - Call-ID: Sequence: Allow: Expires:
- **11 Nov 1997**: Draft SIP -04
  - CONNECTED ➞ ACK
  - UNREGISTER
  - Sequence: ➞ CSeq:
  - Call-Disposition: Require:
- **Dec 1997**: IETF Action: Split SIP into base spec and extensions
Timeline: 1998

**SIP -05**
- CANCEL
- UNREGISTER ➔ Ø
- URL sip://jo ➔ sip:jo
- Record-Route:
- IANA assignments
- Security Cons. Sect.

**SIP -06**
- Clarifications & fixes
- Cleaning up the spec
- Call-ID: MUST
tag parameter

**SIP -07**

**SIP -08**

**SIP -09**
- IETF Action:
- Last Call for Proposed

**Timeline: 1998/99**

**SIP -10**
- IETF Action: Approval for Proposed Standard
- No more DNS MX
- URI: RFC 2396

**SIP -11**
- Update on SDP part

**SIP -12**
- DNS Lookup
- Tidying up

**IETF Action:**
- Published as RFC 2543
- SIP WG formed
Timeline: RFC2543bis (2000/2001)

- **13 Jul 00 bis -00**
- **24 Nov 00 bis -01**
- **26 Oct 01 bis -02**
- **20 Jul 01 bis -03**
- **26 Nov 01 bis -04**
- **28 Nov 01 bis -05**
- **Spring 01**
- **26 Oct 01 bis -06**
- **29 May 01 bis -07**
- **6 Aug 00 bis -08**
- **IETF Action: Formation of new SIPPING WG**
- **PGP removed**
- **Complete Rewrite!**


- **28 Jan bis -06**
- **4 Feb bis -07**
- **21 Feb bis -08**
- **27 Feb bis -09**
- **Jun bis -10**
- **until Jan 03**
- **IETF Action: RFC 3261–3266**
- **SIP-related RFC Rallye:**
  - RFC 3361, 3372
  - RFC 3311, 3312
  - RFC 3323–3325, 3329 (Security)
  - RFC 3398, 3420, 3428
  - RFC 3320–3322 (SigComp)
 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 3320: Reason Header
  - RFC 3327: Registering Non-Adjacent Contacts
  - RFC 3428: Instant Messaging
  - RFC 3485, 3486: SIP Compression
  - RFC 3487: Requirements for Resource Priority
  - RFC 3515: SIP REFER Method
  - RFC 3581: Symmetric Message Routing

- **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
  - RFC 3603: Proxy-to-Proxy Extensions

- **Others**
  - RFC 3365, 3666: SIP Call Flows
  - RFC 3312: Integration of Resource Management and SIP
  - RFC 3361: DHCP Option for SIP Servers
  - RFC 3608: Service Route Discovery
  - RFC 3398, 3578: ISUP and SIP Mapping
  - RFC 3420: Internet Media Type message/sipfrag
  - RFC 3427: SIP Change Process
  - RFC 3455: Header Extensions for 3GPP
  - SDP, RTP, ENUM, CMS, AKA, ...

---

**IETF SIP-related Working Groups (1)**

- MMUSIC WG
  - RFC 2543 (Feb 1999)
  - Sep 99

- SIP WG
  - Mar 01

- SIPPING WG

- SIMPLE WG
  - Dec 00

- XCON WG
  - Oct 03
IETF SIP-related 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

XCON WG
- Centralized Conferencing

SIP is not …

- Intended for conference control by itself
  - 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

SIP and the Multimedia Conferencing Architecture

- Audio / Video
- Conference Control
  - SDP
- Call Control
  - SIP
- RSVP
- RTSP
- RTP / RTCP
- SAP
- UDP
- TCP, SCTP, TLS
- IP/IP Multicast
- Integrated Services Forwarding
Base Terminology

- **User Agent Client (UAC):**
  - Endpoint, initiates SIP transactions

- **User Agent Server (UAS):**
  - Handles incoming SIP requests

- **Redirect server:**
  - Retrieves addresses for callee and returns them to caller

- **Proxy (server):**
  - UAS/UAC that autonomously processes requests
    → 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

![Local SIP Architecture Diagram]

- **Administrative Entity (SIP Server):**
  - Registrar
  - Redirect / Proxy Server
  - Location Server

- **Local IP network:**
  - Endpoint (SIP UA)
  - Endpoint (SIP UA)
  - Endpoint (SIP UA)

- **Interconnections:**
  - PSTN
  - ISDN
  - GSM
  - H.323

© 2004 Jörg Ott

TZi Digitale Medien und Netze
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)

Functional Layers

- **Transaction User**
  - session creation, application-specific processing
- **Transaction**
  - Transaction handling
  - request retransmission
- **Transport**
  - send/receive SIP messages
- **Syntax / Encoding**
  - Message parsing
- **Transport Protocol**
  - UDP, TCP, SCTP
  - TLS
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 succeeds
  - No change on error
- Unique dialog identifier

- transaction indicates state transition
  - setup early dialog
  - establish dialog
  - create dialog

- create dialog
- modify dialog
- destroy dialog

---
**Dialog Example: Media Sessions**

- **Prepare media session:** Grün
- **Establish media session:** Streifen
- **Media session in progress:** Lila
- **Terminate media session:** Grün

**SIP Message Syntax: Request**

```plaintext
INVITE sip:user@example.com SIP/2.0
To: John Doe <sip:user@example.com>
From: sip:jo@tzi.uni-bremen.de;tag=4711
Subject: Congratulations!
Content-Length: 117
Content-Type: application/sdp
Call-ID: 234234233@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;
    branch=z9hG4bK776asdhds
```

**SDP content**

```plaintext
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

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: application/sdp
Call-ID: 2342342330134.102.218.1
CSeq: 49581 INVITE
Contact: sip:jdoe@somehost.domain
Via: SIP/2.0/UDP 134.102.218.1;
    z9hG4bK776asdhds

Message headers

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

Message body
(SDP content)

SIP URI Addressing Scheme

◆ Separating names (permanent) and addresses (temporary)
  – Basic mobility support

◆ Two roles reflected in SIP
  – Naming a user; typically sip:user@domain
  – Contact address of a user; typically contains
    host name or IP address, port, transport protocol, ...

◆ URIs may carry additional parameters

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

params ::= ( [ : name [ = value ] ] )
headers ::= field = value [ & headers ]

◆ URIs may also identify services
SIP Addressing Examples

- sip:tzi.org
- sip:192.168.42.1
- sip:john@example.com
- sip:john@example.com:5060
- sip:foo@example.com:12780

URI parameters may carry detailed information on specific URI components:

- sip:john@Example.COM;maddr=10.0.0.1
- sip:+1555123456@tel-gw.myitsp.com;user=phone

Use URI scheme 'sips' to request secure transmission.

Service Description

Encapsulation

- sip:sip%3A%2F%2F192.168.42.5%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%3D323Bto-tag%3D234ab1&Accept-Contact= \ %3Csip%3Ajo%40134.102.218.1%3C%3Bonly%3Dtrue

Separator characters
Further Common URI Schemes

Telephony (RFC 2806)

\[\text{tel:+1-555-12345678} \]
\[\text{tel:7595;phone-context=+49421218} \]

ITU-T H.323 Protocol

\[\text{h323:user@example.com} \]

Instant Messaging

\[\text{im:user@example.com} \]

Presence

\[\text{pres:user@example.com} \]

URIs in Header Fields

URI-parameters vs. header parameters

\[\text{Contact: sip:bob@p2.example.com:55060;methods="\text{NOTIFY}";expires=3600} \]

→ angle brackets:

\[\text{Contact: <sip:bob@p2.example.com:55060;methods="\text{NOTIFY}";} \]
\[\text{expires=3600} \]

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

Application Scenario 1: Direct Call UA–UA

Internet

Alice  Bob

Call signaling
Media streams
Direct Call

- 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

- Local user is contacted by called UA.
- User clicks on “deny”. UA returns error response.
- Calling UA acknowledges the transaction.
**Caller Gives Up**

INVITE
sip:bob@foo.bar.com

180 Ringing

CANCEL

200 OK

487 Request Terminated

ACK

**Caller Gives Up While Call Established**

INVITE
sip:bob@foo.bar.com

180 Ringing

CANCEL

200 OK

ACK

BYE

200 OK

User answers. Transaction state for INVITE is destroyed. Create dialog, establish media session.

Session teardown. Destroy dialog.
Caller Gives Up While Call Established

**tzi.org**
- alice@ruin

**bar.com**
- bob@foo

- Caller hangs up.
- CANCEL

- 487 Transaction does not exist
- No transaction to CANCEL.

- Destroy state for CANCEL transaction

---

Atomic Transaction Processing

Transactions:
- Created by initial (i.e. not retransmitted) requests
- Terminated by final response to initial request
- Identified by Method, Request-URI, To:, From:, Call-ID:, CSeq: and topmost Via:

- INVITE
- 100 Trying (INVITE)

- CANCEL
- 200 OK (CANCEL)
- 487 Terminated (INVITE)
- ACK

- INVITE transaction
- CANCEL transaction
- INVITE transaction

- Only ACKs for final non-2xx responses

Transactions always complete independently!
How to Find The Callee?

- Direct calls require knowledge of callee’s address
- SIP provides abstract naming scheme:
  
  ```html
  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 (manually) configured outbound proxy
  - Outbound proxy may also have been learned upon registration

- If request URI contains IP address and port, message can be sent directly

- 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

1. I am Bob, please redirect calls to my current address ...
2. Call him at address ...
3. Internet
4. Alice

Call signaling
Media streams

INVITE sip:bob@bar.com
100 Trying
302 Moved Temporarily
Contact: sip:bob@foo.bar.com
ACK
INVITE sip:bob@foo.bar.com
200 OK
ACK
Application Scenario 3: Proxied Call

Calling Bob ...

Incoming call from Alice

I am Bob at address ...

Alice

Internet

Bob

Call signaling

Media streams

Proxied Call

tzi.org

alice@ruin

INVITE sip:bob@bar.com

100 Trying

200 OK

ACK

Media Streams

Subsequent requests

sip.bar.com

INVITE sip:bob@foo.bar.com

100 Trying

200 OK

ACK

bar.com

bob@foo
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

A SIP Network
Global SIP Architecture

SIP (Proxy) Server Functionality

- **Stateless vs. stateful**
  - Stateless: efficient and scalable call routing (backbone)
  - Stateful: service provisioning, 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...**

More elaborate functions provided by Back-to-Back User Agents (B2BUAs)
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

Request Routing

◆ Every server determines next hop
◆ Several destination may be tried in parallel
  – Mark outgoing messages with **branch tag**
  – Use z9hG4bK to indicate unique branch tag
◆ Multiple requests can arrive at a single UAS
  – UAS tags To:-header to identify user presence
  – RFC3261: From:-tags mandatory for request merging at UAS
Loops vs. Spirals

- Proxies may collect state for pending transactions:
  - TCP connections
  - Accounting information
  - Parallel search
  - Need response to clean up
- Insert `Via:`-header when forwarding request
- Send response along reverse path of originating request
- Subsequent requests may take different path!
Creation of Via-path

- Send outgoing responses to first Via-entry
- UAS drops responses with different first Via entry
- Add branch-tag to distinguish different search paths

Response Merging

- Proxy gathers incoming responses until
  - No requests are pending, or
  - Request timers have expired
  - A user-initiated final response arrives (6xx or 2xx)
- Best response is returned to caller, others may be discarded or aggregated ("repairable errors")
- OK-responses are returned whenever received
Record-Route

- Proxies might depend on getting every request of established call
  - Update state information for accounting, call distribution, ...
  - Firewalls and NATs

- **Record-Route** and **Route** headers for request routing
  - Prepend globally reachable request URI with proxy’s address to Record-Route entries
  - Receiving UAS copies contents into response
  - **Route** header may be initialized with pre-existing route set

- Subsequent requests for this call will contain *source route* created from initial Record-Route header

Constructing the Route

- UA A sends request for UA B to P1
- P1 and P2 add request URIs to Record-Route header
- UA B stores recorded route for later use

- UA B literally copies record route of request into response
Using a Recorded Route

- UA A creates new request to B with reversed contents of Record-Route from received response in Route-header
- UA A appends B's contact from former response
- Every proxy forwards request according to Route

UA A creates new request to B with reversed contents of Record-Route from received response in Route-header.

UA A appends B's contact from former response.

Every proxy forwards request according to Route.

UA B does not reverse ordering.

Request URIs for proxies are changed to UA A’s contact.

Significant change from RFC 2543.

Cancelling requests

- Cancel a previous request
- Does not affect complete transactions
  - UAS does nothing if final response was already sent
- Used by forking proxies to prune search tree if OK response arrived.
  
(Case P1)

UA A creates new request to B with reversed contents of Record-Route from received response in Route-header.

UA A appends B's contact from former response.

Every proxy forwards request according to Route.

UA B does not reverse ordering.

Request URIs for proxies are changed to UA A’s contact.

Significant change from RFC 2543.

Cancelling requests

- Cancel a previous request
- Does not affect complete transactions
  - UAS does nothing if final response was already sent
- Used by forking proxies to prune search tree if OK response arrived.
  
(Case P1)
Pre-loaded Routes

- Requests may contain pre-existing Route set → default outbound proxy, CFs in 3GPP
- Loose source routing:
  Forwarding proxy may ignore topmost Route entry
- Route entry rewriting:
  In a response Proxy may rewrite his own Route URI
- Deal with forked routes (due to DNS lookups):

![Diagram showing pre-loaded routes with UA A, P1, P2, UA B, and P2' routing](image)

Heterogeneous Error Response

Forking Problem (HERFP) 1

- P1 forks INVITE request
  - UA B stays silent and solicits “401 Unauthorized”
  - UA C begins Ringing (180)
- Provisional response 180 is forwarded upstream
  - UAC sees ringing indication
- No handling for 401 response possible at this time

![Diagram showing heterogeneous error response with P1, UA A, UA B, and UA C](image)
Heterogeneous Error Response Forking Problem (HERFP) 2

- No answer from C hence A hangs up → CANCEL

Heterogeneous Error Response Forking Problem (HERFP) 3

- P1 forwards best response (401) to A
  - Resubmit INVITE with appropriate credentials for B
- Drawbacks:
  - C will ring again
  - large call-setup delay
  - Forked non-INVITE must be idempotent for sequential search

→ Heterogeneous error responses cause problems
User Location

1. SIP call signaling
2. Arbitrary protocol (beyond the scope of SIP)
3. SIP server asks location server where to find callee
4. Location server returns list of contact addresses
5. SIP server proxies or redirects request according to address list

INVITE sip:jo@tzi.org SIP/2.0
To: sip:jo@tzi.org
From: sip:bergmann@tzi.org;tag=4711
Contact: sip:bergmann@134.102.218.57

SIP/2.0 302 Moved Temporarily
Contact: sip:jo@134.102.218.59

134.102.218.57
User Registration

REGISTER sip:tzi.org SIP/2.0
To: sip:jo@tzi.org
From: sip:jo@tzi.uni-bremen.de
Contact: sip:jo@134.102.218.59,
sip:joerg@134.102.218.86
...
User Registration (2)

- Send REGISTER request to registrar
- Request URI sip:domain

→ registrar may refuse requests for foreign domains

- To: canonical 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, capabilities

- 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: *
Extended Registration Functions (1)

- Registrar is the initial point of contact for a UA
- Idea: provide (user-specific) configuration and other information in REGISTER response
  - Allows for centralized control using standard SIP mechanisms

1. Service Route Discovery

- Dynamically configure “home service proxies” for UAs
  - Proxies to be traversed first for outbound requests
  - Routing of incoming requests can be enforced by infrastructure
  - Routing of responses implied from Via: headers
- Simple extension header returned by registrar
  - Syntax similar to Route: header
- May be used by UA subsequently for loose source routing
  - Included in Route: header

Extended Registration Functions (2)

2. Globally Routable UA URI (GRUU)

- Users may be registered at multiple UAs
- Incoming SIP requests may be forked to all UAs
- May need to direct a request to a particular UA
  - Examples: Call Transfer, Conferencing, Presence

- Additional parameter in Contact: header from registrar
  - Valid for the lifetime of this UA’s registration
  - Local proxies ensure proper routing to the right UA
- UA may place GRUU in Contact: header in later requests
  - UA uses grid parameter to distinguish individual uses of a GRUU
- Remote UAs will use GRUU to send messages
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 server:
Store registration
Return OK response

User agent:
Silently drop
REGISTER request

User agent:
Silently cache
contact address

→ registrar is 134.102.218.99
Media Session Setup Between Endpoints

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

“Normal” operation

 UA A

 INVITE
 Caps(A)

 200 OK
 Caps(A) ∩ Caps(B)

 ACK

 UA B

 Delayed description

 UA A

 INVITE

 200 OK
 Caps(B)

 ACK
 Caps(A) ∩ Caps(B)

 UA B

 Use of UPDATE in Offer/Answer Model

 A

 INVITE Caps(A)

 183 Session Progress
 Caps(A) ∩ Caps(B)

 PRACK

 200 OK

 UPDATE Caps(A')

 200 OK
 Caps(A') ∩ Caps(B')

 UPDATE Caps(B'')

 200 OK
 Caps(A'') ∩ Caps(B'')

 ACK

 B

 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

SDP Version

Originator + version

© 2004 Jörg Ott
SDP Example

Media Name and Transport Address

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

Connection Information
(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
SDP in SIP Applications

Alice @ 192.168.1.1
Bob @ 192.168.1.2

INVITE sip:bob@192.168.1.2

Audio session
with a set of alternatives

Additional media session

v=0
o=alice 7595 J IN IP4 192.168.1.1
s=Hello again
e=alice@example.com
t=0 0
c=IN IP4 192.168.1.1
m=audio 46000 RTP/AVP 96 97 98
a=rtpmap:96 G729/8000
a=rtpmap:97 GSM-EFR/8000
a=rtpmap:98 PCMU/8000
m=audio 46002 RTP/AVP 99
a=rtpmap:99 telephone-events

200 OK

v=0
o=alice 5160 J IN IP4 192.168.1.1
s=Hello again
e=alice@example.com
t=0 0
c=IN IP4 192.168.1.1
m=audio 46000 RTP/AVP 96 97 98
a=rtpmap:96 G729/8000
a=rtpmap:97 GSM-EFR/8000
a=rtpmap:98 PCMU/8000
m=audio 46002 RTP/AVP 99
a=rtpmap:99 telephone-events

Only one audio format supported
Second media session not supported
SDP in SIP Applications

alice@192.168.1.1  

bob@192.168.1.2

INVITE sip:bob@192.168.1.2
v=0
o=alice 7595 1 IN IP4 192.168.1.1
s=Hello again
e=alice@example.com
t=0 0
c=IN IP4 192.168.1.1
m=audio 46000 RTP/AVP 96 97 98
a=rtpmap:96 G729/8000
a=rtpmap:97 GSM-EFR/8000
a=rtpmap:98 PCMU/8000
m=audio 46002 RTP/AVP 99
a=rtpmap:99 telephone-events

200 OK
v=0
o=bob 5160 1 IN IP4 192.168.1.2
s=Hello again
e=bob@example.com
t=0 0
c=IN IP4 192.168.1.2
m=audio 34000 RTP/AVP 98
a=rtpmap:98 PCMU/8000
m=audio 0 RTP/AVP 99
a=rtpmap:99 telephone-events

ACK

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:

jo@ruin 134.102.218.1  

audio data  
L8/8000  

49823

(cab@)  

134.102.218.46

(no video)
Send/Receive Only

◆ Media streams may be unidirectional
  – Indicated by \texttt{a=sendonly, a=recvonly}
◆ Attributes are interpreted from sender’s view
◆ \texttt{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
◆ \texttt{recvonly}
  – Sender lists supported codecs
  – Receiver chooses the subset he intends to use
  – Multicast session: recipient listens on specified address
◆ \texttt{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 (RFC 3311)

- 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
- SIP in Telephony
- SIP in 3GPP

You are here

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)

Some countermeasures:

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

Typical threats for distributed applications using the public Internet

Reuse existing security mechanisms.
Why SIP Security?

- Ensure privacy
  (media encryption, anonymous calls, personalized services, ...)

- Billing and accounting
  (probably pay for services, assured bandwidth, etc.)

- Regulatory requirements
  - Call id blocking
  - Prosecute abuse (call tracing facility)
  - Emergency call service
  - Multi-Level Priorization and Preemption
Hop-by-hop Encryption of SIP Messages

◆ Lower layer mechanisms
  – Applicability depends 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
  – e.g. in trusted networks

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 builds 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**
- **Use SIP message bodies to carry confidential information**
  - Can be used to achieve end-to-end encryption
    - E.g., when the network cannot be trusted or other mechanisms are not available
    - Does only protect the message body, not the header fields
- **S/MIME**
  - Allows to encapsulate arbitrary content for security purpose
  - Encryption and digital signature

**Issue: Key distribution problem prior to call**
- No global PKI
- Certificates for SIP proxies, typically not for users
- Typically no shared secrets
- Validation of end user certificates will have to rely on public key infrastructures (or pre-configured certificates)
SIP and S/MIME for End-to-End Encryption

◆ Carry (partial) SIP messages in S/MIME body
  – Content-Type: message/sipfrag (or: message/sip)
  – Apply MIME security to message body
  – Use multiparty/mixed if necessary
    – e.g. to also carry SDP bodies

◆ Include confidential headers in message body
  – E.g. From:, Contact:, Subject:

◆ Keep headers required for routing “outside” in the clear
  – Some outer header fields remain visible
    – To, From, Call-ID, Cseq, …
  – Some mandatory headers may be populated with dummy values
  – Some outer header fields may also be changed in transit
    – Request-URI, Via, Record-Route, …

◆ May also be used for end-to-end authentication

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

◆ Original SDP allows only for one per media key field (“k=“)

◆ SDP extensions for better keying support
  – Requires encrypted SDP in SIP message body
  – Requires protected communication path

◆ Further SDP extensions for secure media keying
  – MIKEY allows for end-to-end negotiation of keys
  – Protection of the exchanged information within SDP
# Simple SIP Authentication

- **HTTP digest authentication**
  - basic authentication deprecated

- **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**

## Authentication: Example Call Flow

**INVITE**

sip:jo@tzi.org

401 Unauthorized

**ACK**

**INVITE**

sip:jo@tzi.org

200 OK

**ACK**

**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

Multiple Authentication Steps


Proxy: INVITE: Proxy-Authenticate: ...

UAS: INVITE: 401 Unauthorized: INVITE: WWW-Authenticate: ...

Proxy: INVITE: Proxy-Authorization: ...

UAS: 180 Ringing: 200 OK: ACK
Transitive Trust Relationships

- UAC establishes TLS connection
- Proxy provides certificate
- UAC verifies cert
- Password-based authentication of calling user
- P1 adds P-Asserted-Identity, signs message and routes to P2
- ITSP B trusts signed AI
- P4 and UAS have authenticated → UAS trusts AI

Privacy of Calling Party Information

- Option-tag privacy to enforce treatment at proxies
- Extension header Privacy:
  - Initiator can specify a required level of privacy
  - Typically, intermediary has to obscure certain header fields → B2BUA
  - Different types of privacy information (header, session, user)

- Use asserted identity mechanism if authentication required → need trust relationships between ITSPs
SIP and S/MIME for End-to-End Authentication

- Header fields are included in a signed S/MIME message body
  - Idea similar to encryption: message/sip, message/sipfrag
  - Include enough headers to prevent replay
    - From:, Date:, Contact: (To:, Call-ID:, CSeq:)
- UA may sign the message itself: straightforward
  - Workable -- but receiver has to obtain UA’s certificate
  - Work in progress to locate Certificate servers per domain
- UA may have an authentication server sign on its behalf
  - Requires (local) trust relationship between UA and auth server
  - Requires only auth server’s certificate to be known
- Authenticated Identity Body (AIB)
  - Third party ensures receiver about sender identity
  - UA prepares and includes the above message body
  - Asks auth server to sign
  - Indicated by: Content-Disposition: message/aib

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
Basic 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

- 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
- SIP for presence
- ...

- Addressed later today where appropriate...
Reliable Provisional Responses

- Signaling Gateways, ACD systems etc. may depend on provisional responses
  - Such as 180 Ringing

  → 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

A starts sending a reliable 182 response

A receives 182 response

PRACK for previous 182 response arrived, transmit OK response

B receives 1xx with RSeq:-header

B sends PRACK

B receives 200 OK

B stops PRACK retransmissions

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/UPDATE arrives 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

- A
  - INVITE
    - Supported: timer
    - Session-Expires: 20
  - 422 Session Timer Too Small
    - Min-SE: 30
  - ACK
  - INVITE
    - Supported: timer
    - Session-Expires: 40
    - Min-SE: 30
  - 200 OK
    - Session-Expires: 30 ;refresher=uas
    - Min-SE: 30
  - ACK

- Proxy

- B
  - INVITE
    - Supported: timer
    - Session-Expires: 40
    - Min-SE: 30
    - ACK
  - 200 OK
    - Session-Expires: 30 ;refresher=uas
    - Min-SE: 30
    - ACK
Event Notifications

◆ Need for flexible event notification
  – Enable presence information
  – Better support for mobile SIP applications
  – 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
  – Used with PUBLISH method to update presence state
  – 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 register interest in a piece of state
  – Event: identifies the state information to be retrieved
    ♦ 489 Bad Event
    ♦ Syntax reflects package concept:
      
      \[
      \text{package ( 'template' }^* \rightarrow \text{ e.g. 'Event: presence.winfo'}
      \]
  – 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 lengthened by server
  – Expires: 0
    ♦ Poll the state information once; do not establish a subscription
    ♦ Terminate a subscription
  – Body may indicate desired notification policy (filters)

◆ 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

◆ Accepting / rejecting subscription requests
  – Automated e.g. through configured black / white list
  – Ask user if acceptance cannot be determined automatically

◆ Watcher Info package used to monitor one’s own presence
  – Generates notifications for subscription requests
  – User then needs to authorize subscription
    ♦ Through arbitrary means (e.g. web page)
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

◆ Accepting / rejecting subscription requests
  – Automated e.g. through configured black / white list
  – Ask user if acceptance cannot be determined automatically

◆ Watcher Info package used to monitor one’s own presence
  – Generates notifications for subscription requests

◆ 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
  – Event throttling to be specified in event packages
Example: Successful SUBSCRIBE

SUBSCRIBE
Event: voice-mail
Expires: 3600
Accept: text/plain

200 OK
Event: voice-mail
Expires: 1800

NOTIFY
Event: voice-mail
Expires: 1759
Subscr-State: active

200 OK

Example: Unsuccessful SUBSCRIBE

SUBSCRIBE
Event: voice-mail
Expires: 3600
Accept: text/plain

∅

489 Bad Event
Allow-Events: zompel

202 Accepted
Event: voice-mail

NOTIFY
Event: voice-mail
Expires: 0
Subscr-State: terminated
:reason=rejected

∅

200 OK
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

Referrer A | Referee B | Refer target C
---|---|---
Referrer B sends REFER method to Referee B
- refers to Refer target C
Referred party sends 202 Accepted
Referred party sends NOTIFY
- Content-Type: message/sipfrag
- Content-Length: 6
Originator receives NOTIFY
- 200 OK
Originator sends INVITE
- Referred-By: a@tzi.org
Originator receives 100 Trying
Originator receives 200 OK
Originator sends ACK
Originator receives 200 OK
Originator sends BYE
Originator receives 200 OK
Securing REFER

◆ Refer Target C receives a request from Referee B
  – May be indistinguishable from any other request from B
  – May be acceptable to C only if A is known as initiator
  – Need to provide authentication information in the request

◆ Referred-By: header provides first indication
  – But may be faked by B

◆ Extension to securely pass information from A to C
  – Include signed S/MIME message body in REFER request
    ♦ Content-ID: <sip-message-id>
    – To be copied by Referee into target request
  – Optional cid= parameter of Referred-By: points to signed body
    ♦ Referred-By: <sip:referrer@example.com>;cid="<sip-message-id>"

◆ Refer target may refuse message without proper Referred-By
  – 429 Provide Referrer Identity

REFER Semantics?

◆ Refer-To: supports arbitrary parameters
  – Including method= and response=
  – May populate various headers of a message
  – Enables remote control of a SIP UA

◆ Default: INVITE method for SIP URIs
  – Inside dialog: call transfer
  – Outside dialog: click-to-dial

◆ But generalized command: Apply <method> to <URI>
  – “Force” a remote UA to do something
  – In principle broad applicability but only little semantics “defined”
    ♦ Fetch the content at http://…
    ♦ Subscribe to a certain resource
    ♦ Send a certain reply to a request
    – Handle with care…
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
    - Parameters: require, explicit
  - Reject-Contact: Reject particular URIs
  - Implied predicates from SIP method, events
  - Additional parameters for Contact: headers in REGISTER

Preference/Capability Matching

1. REGISTER sip:tzi.org SIP/2.0
   To: <sip:jo@tzi.org>
   Contact: <sip:+49171...@dtag.de>
   ;audio ;mobility="mobile"
   ;methods=“INVITE,MESSAGE,...”
   Contact: <sip:jo@134.102.218.1>
   ;audio;video
   ;mobility="fixed"

2. SELECT * FROM ...
   WHERE...

3. INVITE sip:jo@134.102.218.1

4. INVITE sip:jo@tzi.org SIP/2.0
   Reject-Contact: * ;mobility="mobile" ;media="!audio"
   Request-Disposition: proxy, parallel
**SIP Service Creation Model**

- **SIP Core Protocol**
  - UDP
  - TCP, SCTP, TLS
  - IP/IP Multicast

**SIP-based Services**
- Call Control
- IMPP
- Telephony
- 3G
- ...

**SIP Extensions**
- REFER
- SUBSCR.
- NOTIFY
- PUBLISH
- MESSAGE
- INFO
- PRACK
- UPDATE
- ...

**Call Processing Language (CPL)**

- **Per-user configuration of server behavior**
  - Idea of flexible service configuration (as on a PBX)
  - 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; upload

- **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

call

addr-switch “destination”

tzi.org

no answer

proxy to +49421218...

busy

Proxy to +49171...

otherwise

Proxy to Voicemail@..

busy

no answer
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

SIP for Presence and Instant Messaging

(a somewhat brief introduction)
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

◆ Define a new method for Instant Messaging
  – MESSAGE
  – Define basic message content format (text/plain)
IMPP Presence Model

- Watcher UA
- Presence UA

- Watcher
  - Fetcher (Poller)
  - Subscriber

- Presentity
  - S
  - S
  - S
  - S

Presence Service

SIP Entities for Presence (1)

- Notifier
  - Event State: ob@tzi.org

- Notifier
  - Event State: jo@tzi.org

- Subscriber
  - Event State: ob@tzi.org

- Subscriber
  - Event State: jo@tzi.org

SUBSCRIBE / NOTIFY

Presence UAs

Presence UA
Example: SIP for Presence

```
SUBSCRIBE
Event: presence
Expires: 3600
Accept: text/plain
```

```
200 OK
Event: presence
Expires: 1800
```

```
NOTIFY
Event: presence
Expires: 1759
Subscr-State: active
```

```
200 OK
```

```
SUBSCRIBE
Event: presence
Expires: 3600
Accept: text/plain
```

```
202 Accepted
Event: presence
```

```
NOTIFY
Event: presence
Expires: 1759
Subscr-State: active
```

```
200 OK
```

SIP Presence Notification (PIDF)

```
<presence ...
entity="pres:JoE@Tzi.Org">
  <tuple id="mobile-im">
    <status>
      <basic>open</basic>
    </status>
    <contact priority="0.8">im:Jo@smsg.WTzi.Org</contact>
    <note xml:lang="en">Don't Disturb Please!</note>
    <note xml:lang="fr">Ne dérangez pas, s'il vous plait</note>
    <timestamp>2003-01-14T10:49:29Z</timestamp>
  </tuple>
  <tuple id="interactive-mm">
    <status>
      <basic>closed</basic>
    </status>
    <contact priority="1.0">sip:JoE@Tzi.Org</contact>
  </tuple>
  <note>I'll be in Paris next week</note>
</presence>
```
Rich Presence Information Data Format (RPID)

- Introduces additional (more descriptive) elements
  - Ideally to be derived automatically from user activity
  - Organized in tuples
  - May be grouped into classes

Elements and value examples:

- Activity: on-the-phone, away, meeting, travel, vacation, ...
- Contact-Type: device, in-person, services, ...
- Idle: inactivity time of a user (e.g. not typing)
- Place: home, office, public, street, train, ...
- Privacy: public, private, quiet
Example: PUBLISHing State

**Install State**

- **EPA**
  - PUBLISH
  - Event: presence
  - Expires: 3600
  - Content-Length: ...

- **ESC**
  - 200 OK
  - Expires: 1800
  - SIP-ETag: 116xyz42

**Refresh State**

- **EPA**
  - PUBLISH
  - Event: presence
  - Expires: 1800
  - SIP-If-Match: 116xyz42
  - Content-Length: 0

- **ESC**
  - 200 OK
  - Expires: 1800
  - SIP-ETag: 891abc4190

**Modify State**

- **EPA**
  - PUBLISH
  - Event: presence
  - Expires: 1800
  - SIP-If-Match: 891abc4190
  - Content-Length: ...

- **ESC**
  - 200 OK
  - Expires: 1800
  - SIP-ETag: 336hoh1875

**Delete State**

- **EPA**
  - PUBLISH
  - Event: presence
  - Expires: 0
  - SIP-If-Match: 336hoh1875
  - Content-Length: 0

- **ESC**
  - 200 OK
  - Expires: 0

---

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

---

**REGISTER**

Accept: text/plain
Contact: jo@tzi.org;:methods="MESSAGE"

200 OK

**MESSAGE**

Accept: text/plain
Content-Type: text/plain
Content-Length: 12

Hello, world

200 OK
Presence & IM 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**

---

Issue: Congestion Control

- **PUBLISH, NOTIFY: Throttling of events**
  - Keep event rate under control: one PUBLISH / NOTIFY per RTT
  - To be defined on a per-package basis

- **MESSAGE: Message frequency**
  - Only one outstanding message: one MESSAGE per RTT
  - But: messages are stand-alone; no dialog context to check against

- **MESSAGE: Large messages**
  - UDP is an acceptable transport for SIP: no congestion control
  - Endpoints can’t see beyond next hop
    - Artificial limit on message size?
    - Request end-to-end congestion control?
    - Content indirection: convey only pointers (URLs) to contents in message?

- **Approach: Message Session Relay Protocol (MSRP)**
Message Session Relay Protocol (MSRP)

- Protocol for Messaging Sessions
  - Uses TCP or another reliable and congestion controlled transport
  - Message encoding similar to SIP and HTTP

- Just another media protocol
  - Messaging sessions require explicit setup and teardown
  - MSRP URLs to identify sessions
  - SDP to describe sessions (m=message)
  - Uses SDP Offer/Answer to convey parameters
    - e.g. in SIP INVITE / 200 OK / ACK
  - VISIT method to establish session
  - SEND method to convey messages
  - Session termination out of band (e.g. SIP BYE)

- Two modes of operation
  - Direct communication between peers (simple case)
  - Communication via relays (NATs, firewalls, policy)
    - BIND method to work with relays

Direction Communication between Peers

A

Choose URL

SIP INVITE

VISIT

SEND

SIP BYE

B

200 OK

200 OK

200 OK

200 OK

m=message <port> msrp/tcp *
c=IN IP4 somenametogohere
a=accept-types:text/plain, text/html
a=direction:both
a=session:msrp://a.domain.org/some-id
Communication via a Relay

A

BIND

Choose URL

B

TD-ID: 314
Expires: 600
S-URL:msrp://r.org/

TD-ID: 314
Expires: 300
S-URL:msrp://r.org:9999/35

S-URL:msrp://r.org:9999/35

200 OK

SIP INVITE

VISIT

200 OK

SEND

ACK

SEND

ACK

200 OK

BIND

200 OK

SIP BYE

200 OK

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 send 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
◆ Message waiting
  – Specific event package: message-summary
◆ Conferencing
  – Various extensions; supports base and enhanced SIP UAs
Third-Party Call Control (3PCC)

- Examples: “Click-to-dial”, conference bridge control, …
- Several approaches with different advantages / drawbacks
- Simplest call flow:

```
Step 1

1. INVITE no SDP
2. OK SDP(A)
3. INVITE SDP(A)

UA A          Controller          UA B

Step 2

4. OK SDP(B)
5. ACK
6. RTP media
7. ACK

UA A          Controller          UA B
```

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

A
Active SIP dialog

On-hold (INV/200 OK/ACK)

REFER
Refer-To: sip:c@tzi.org

202 Accepted

NOTIFY
Content-Type: message/sipfrag
Content-Length: 6

200 OK

B

INVITE
Referred-By: a@tzi.org

100 Trying

200 OK

C

ACK

200 OK

200 OK

200 OK

BYE

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)
SIP Conferencing

- **Motivators:** n-way calling, video conferencing, ...
  - Tightly coupled conferences
- **Different conferencing models preserved**
  - Except for “fully meshed” conference: complexity just not worth it!
- **Terminology**
  - Trying to avoid already overloaded terms as much as possible
- **Functional entities in a “system model”**
  - Plus implementations examples
- **Set of protocols**
  - Definition of basic building blocks (SIP and other)
  - Sample combinations to implement conferencing services

Conference Participants

- **MUST work with basic SIP support only**
  - No awareness of conference
  - Just a point-to-point call with minimal means for control
  - Possibly augmented by out-of-band control (e.g. HTTP)
- **Member types (SIP UA)**
  - Conference-unaware: plain-old SIP device
  - Conference-aware: supports conferencing features
  - Focus: (one) center of a conference
  - Anonymous: Visible but unidentified participant
  - Invisible: Participant whose presence is not known
Conferences

- **Conference types**
  - Basic: just plain SIP, no further means for control
  - Complex: some conferencing features provided
  - Cascaded: several foci concatenated in a conference
  - Sidebar: conference as (logical) part of another

- **Focus**
  Signaling center of a conference

- **Conference URI**
  Identifies a focus
  *(isFocus parameter may indicate this)*

- **Factory URI**
  for automated conference creation
  - Yields a dynamically generated conference URI in return

Terminology and Model
Terminology and Model

Conference Policy Control Protocol (CPCP)

Focus

Conference Policy Server

Member-Ship & Media Policy

SIP Subscription

Focus

Conference Notif. Service

Mixer

e.g. RTP Sessions

SIP Dialog

Participant

SIP Signaling Relationships

SIP UA (Ann) -> SIP UA (Carol) -> SIP UA (Dave)

SIP UA (Dave) -> SIP UA (Ann) -> SIP UA (Carol)

SIP UA (Carol) -> SIP UA (Dave) -> SIP UA (Ann)
Conferencing Scenarios (1)

- **Simple conferencing scenarios**
  - Plain SIP only (RFC 3261, 3264)

- **Extend point-to-point call**
  - Works only with local focus; otherwise, new call required

- **Ad-hoc conference**
  - Automated creation at focus
  - IVR / DTMF for control
  - Audio for information about the conference and its members

- **Reserved conference**
  - Same as ad-hoc
  - Use external means for reservation and configuration (e.g. web)
Conferencing Scenarios (2)

- Advanced conferencing scenarios:
  - Support for Call Transfer
  - Support means to communicate information from focus to UA
  - Optional: means to manipulate conference and media policy

- Extend point-to-point call
- Join / create a conference based upon an existing dialog
- Ad-hoc conference
- Reserved conference

- Make use of additional conferencing features

Sample Conferencing Features

- Invite participants (dial-in, dial-out), expel participants
- Authenticate new participants by members
- Obtain conference and media policy information
- Manipulate membership policy
  - Participant privileges, participant management (black list, white list)
  - Floor control
- Explicit media control (media policy)
  - Configure media distribution
  - Add / remove media sessions
- Create, control, and terminate sidebars
  - Separate conference vs. media policy
- …
Membership Policy

Define, retrieve/notify, modify, and act upon...

- Formal rules for the conference
  - Conference creation, termination, (policy) modification
  - Access control: black list, white list, rules for authentication
  - Privileges of individual participants
  - Visibility of the conference and its members
  - Access to floor and media policy (defined separately)

- General conference attributes

- Participant management
  - Invite, expel

- ...

Media Policy

- Mixer model
  - Input switch: collecting & selecting input streams from participants
    - Possibly transcoding and other per-media functions
  - Mixing topologies describing mixing policies
  - Output switch: selecting & distributing output streams to participants
    - Possibly transcoding and other per-media functions

- Mixer may be centralized or not

- Media policy defines how incoming streams are processed, combined, and then distributed
  - Individual mixing functions may be defined per participant
  - Common mixing functions may be defined for the conference
  - Mixing function may take into account “events” from other components
SIP Signaling Building Blocks

- **Membership control**
  - Initiation of conferences: INVITE
  - Inviting / adding to conferences: INVITE, REFER
  - Leaving a conference: BYE
  - Expelling from a conference: REFER (method="BYE")

- **Conference control**
  - State change notifications: SUBSCRIBE / NOTIFY
    - Dialog package, conference package, ...
  - Conference / media policy control
    - Might use data manipulation framework discussed in SIMPLE context

- **Other**
  - Determine focus URI OPTIONS
  - ...

- **Augmented by other protocols (CPCP for conference control)**

---

Signaling Building Blocks using other Protocols

- **Conference Policy Control Protocol (CPCP)**
- **Floor control protocol**

- **Common baseline:**
  - Request-response protocol (RPC-style)
  - May be strictly client – server
  - Asynchronous notifications provided by SIP
  - HTTP/HTML access to a web page for human interaction
  - SOAP RPCs e.g. over HTTP
Call Flow Example: Conference Creation

Alice  Focus  Bob  Carol
INVITE sip:conf-factory
180 Ringing
200 OK Contact:conf-id
ACK
Media sessions
SUBSCRIBE sip:conf-id
200 OK
NOTIFY
200 OK

Call Flow Example: User Joining

Alice  Focus  Bob  Carol
Media sessions
INVITE sip:conf-id
180 Ringing
200 OK Contact:conf-id
ACK
Media sessions
SUBSCRIBE sip:conf-id
200 OK
NOTIFY
200 OK
Call Flow Example: Adding a User

Alice | Media sessions | Focus | Media sessions | Bob | Media sessions | Carol
---|---|---|---|---|---|---
REFER sip:conf-id
Refer-To: carol
202 Accepted
NOTIFY
200 OK
NOTIFY
200 OK
Media sessions
INVITE Contact:conf-id
180 Ringing
200 OK
ACK
Media sessions
SUBSCRIBE sip:conf-id 200 OK
NOTIFY 200 OK
NOTIFY 200 OK

Call Flow Example: Removing a User

Alice | Media sessions | Focus | Media sessions | Bob | Media sessions | Carol
---|---|---|---|---|---|---
REFER sip:conf-id
Refer-To: carol?m=BYE
202 Accepted
NOTIFY
200 OK
Media sessions
BYE sip:carol
200 OK
NOTIFY
200 OK
NOTIFY
200 OK
Call Flow Example: User Leaving

Alice → Focus → Bob

Media sessions → Media sessions

NOTIFY → 200 OK → ACK

BYE sip:conf-id

Media sessions

Terminate conference e.g. by non-SIP means

Interfacing to the PSTN

1. PSTN — PSTN

2. SIP — PSTN

IP based network

SIP Proxy

Border element maps SS7 to SIP and vice versa

SIP

SS7
Interfacing to the PSTN

- 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
  - Address resolution using ENUM or TRIP
- Convey additional information during call
  - PSTN – SIP – PSTN case
    - INFO method (RFC 2976)
- SIP to PSTN mapping
  - Call flows for basic interoperation (RFC 3666)
  - Support for Overlap Sending

Example for SIP and Overlap Sending (1)

- Approach 1: Collect digits in gateway
  - Use timeout
  - collect minimal number of digits
Example for SIP and Overlap Sending (2)

- **Approach 2: Send multiple INVITEs**
  - Use timeout, possibly collect minimal number of digits
  - Wait for feedback from the SIP network
  - CANCEL obsolete INVITEs

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

- Optionally made reliable (100rel)
  - PRACK message to confirm receipt of 183

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 *early* media session → Cannot decline session or change codecs
**Quality of Service**

- Best-effort quality potentially too poor for IP telephony
  - need support for resource reservation in SIP

- Some applications need multi-phase call-setup
  - Resource reservations to assure certain QoS
  - Establish particular security relationships
    - Extend security services to media channels
    - Avoid fraud and theft-of-service

- Establish call only if **preconditions** are met
  - No charge for defect calls
  - No inconsistent state of involved endpoints
    (alert user only if resource reservation succeeded, ...)
  - Fallback options: e.g. use codecs with lower bandwidth
SIP Preconditions (RFC 3312)

◆ SDP extension handles negotiation of QoS parameters
  – SDP attributes describe preconditions to be met
  – Current status (curr:) vs. desired status (des:)
  – Either party may request confirmation on current state (conf:)

'a=curr:' type status direction
'a=des:' type status strength direction
'a=conf:' type status direction

◆ Offer/answer to create common view on both UAs state
  – Typically INVITE, UPDATE, PRACK
  – Option tag precondition
  – New response code 580 Precondition Failure
  – Mark failed and unknown preconditions in SDP answer

QoS Parameter Negotiation (Overview)

Caller sends state description
Update desired state. Need caller to reserve in send direction
→ confirmation request

Acknowledgment answer
Independent reservations made

Confirmation from caller after successful resource reservation
Callee signals successful reservation. User is being alerted.
Example Negotiation of QoS Parameters

**Confirmation request**

```
v=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
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
m=video 59485 RTP/AVP 31
a=rtpmap:31 H261/90000
a=curr:qos local send
a=curr:qos remote none
a=des:qos optional local sendrecv
```

**Confirmation response**

```
v=0
c=IN IP4 134.102.218.46
m=audio 50239 RTP/AVP 98 99
a=rtpmap:98 L8/8000
a=rtpmap:99 L16/8000
a=curr:qos e2e none
a=des:qos mandatory e2e sendrecv
a=conf:qos e2e recv
m=video 56112 RTP/AVP 31
a=rtpmap:31 H261/90000
a=curr:qos local none
a=curr:qos remote send
a=des:qos failure local sendrecv
a=des:qos optional remote sendrecv
```

**Successful Reservation**

- INVITE indicates support for reliable provisional responses
- Turn off local user alerting
- Choose appropriate subset of preconditions offered by sender
- Include confirmation request
- Reliable provisional response must be acknowledged
- Initiator of confirmation request awaits UPDATE message or CANCEL
- Continue call setup
Failed Reservation

- INVITE contains preconditions with confirmation request
- Indicate support for reliable provisional responses

- Turn off local user alerting
- No reservation results

- Reliable provisional response must be acknowledged

- Callee’s resource reservation fails

- Send error response to caller
- Failed call setup complete

Interaction with Application Services

- Traditional PSTN / PBX services often controlled via DTMF
  - Stimulus signaling with semantics defined by the application
  - Assumes that signaling and media paths are identical

- Does not apply to SIP
  - Media and signaling may go separate paths
  - Application servers may be on neither path

- SIP defines Application Interaction Framework
  - Taxonomy for interactions between users and application servers
  - Focus on stimulus-based signaling

- KPML Approach
  - SUBSCRIBE to register interest of an application server in user input
  - XML document to describe input expected from user
  - NOTIFY to inform server about key events
Operational Issues

- Regulatory Stuff ➔ Friday
- Deployments ➔ Wednesday, Thursday
- Accounting / Billing ➔ Friday
- NATs / Firewalls ➔ Thursday
- Devices ➔ …

Regulatory Issues

- Public telephony networks must fulfill certain functions
  - Legal interception
  - Prohibit abuse or malicious use
  - Availability for disaster recovery
  - ➔ Need feature transparency in hybrid networks

- Multilevel Priorization (MLP)
  - Increase probability for call-setup on overloaded networks
  - ISUP IAM precedence parameters and calling party category
  - ➔ Resource-Priority: header in SIP
  - Avoid abuse
  - ➔ authentication and authorization

- Preemption
  - Make resources available for high-priority traffic
    (release existing resource reservations if necessary)
  - Legal issues in many countries
Regulatory Issues

- Public telephony networks must fulfill certain functions
  - Legal interception
  - Prohibit abuse or malicious use
  - Availability for disaster recovery
  - Need feature transparency in hybrid networks

- Multilevel Priorization (MLP)
  - Increase probability for call-setup on overloaded networks
    - ISUP IAM precedence parameters and calling party category
    - Resource-Priority: header in SIP
  - Avoid abuse
    - authentication and authorization

- Preemption
  - Make resources available for high-priority traffic
    (release existing resource reservations if necessary)
  - Legal issues in many countries

Multilevel Priorization (MLP) e.g.: E.106, GETS, GTPS

SOS Calls

- Emergency calls for individuals
  - No calling party categorization
  - Geographic location of caller
  - Prohibit abuse: authentication, no CLIR possible
  - Need to respect privacy

- Access network may be far away from endpoint’s location
  - Find geographically nearest emergency center
  - Map symbolic SOS-identifier to suitable emergency code
  - Add geographic info

ACD of local emergency center

PSTN

moblie endpoint
tel:112
Firewalls, Proxies, NATs

- Intermediate devices in the Internet (aka Middle Boxes)
  - Can be transparent or non-transparent to hosts

- Different functions:
  - Packet-filters (firewalls)
  - Application layer gateways (proxies)
  - Network address translators (NATs)

- Middlebox communication

Firewalls

- Packet filters, enforcing packet filtering/forwarding policies
- Usually transparent to hosts and applications
- Different motivations
  - Security: protect a network from attacks
  - Resource management: restrict usage of the network
- Different filtering criteria
  - Source and Destination IP address
  - Protocols: ICMP, TCP, UDP etc.
  - TCP and UDP port numbers (applications)
- Filter specification
  - Usually statically configured
  - Most configurations disallows packets for “non-standard ports”

- Corporate Intranet
- Internet

Packet filter spec
- Source, destination IP addr.
- Protocol (UDP, TCP, ICMP)
- Source, destination port
- ...
- May be dynamically configured.
Firewalls and SIP

◆ Default configuration of many firewalls:
  – Allow outbound TCP and outbound UDP for DNS
  – Allow inbound TCP for well-known TCP ports only (e.g. HTTP)
◆ SIP communication disallowed in most cases
  – Port 5060 (UDP, TCP)
  – Relatively easy to fix: re-think filter firewall policy and update filter
    spec. to allow port 5060
  ♦ Still a problem with SIP communication on other ports…
◆ The real problem:
  – Media sessions
  – Transport parameters are signalled by SIP messages
  ♦ Negotiable, not known in advance
  ♦ Different transport mechanisms for different applications
◆ Solution approaches
  – Have firewalls be controlled by ALGs
  – Adapt SIP to make it more firewall-friendly
  – Define firewall-control protocols to open firewalls dynamically
  ♦ Issues: Authentication and authorization

Network Address Translators

◆ Intermediate systems that can translate addresses (and
  port numbers) in IP packets
  – Often used to map global addresses to address/port number
    combination of hosts in a corporate network
◆ Different motivations
  – Security
    ♦ Make internal host inaccessible from the public Internet
  – Efficient usage of address space
    ♦ Share one globally unique address
◆ NATs create address/port number mappings
  – Mappings are usually created dynamically, e.g. on connection setup
  – Static configurations also possible
  – Works best with connection oriented communication
  – Most common case: TCP connection from client-server sessions
    ♦ Client in private address space, server in public Internet
  – NATs have to keep state for mappings that are tied to connections
Network Address Translators

Corporate Intranet

Translation table

H1: ftp
H2: ftp
H2: telnet
H3: http

10.0.42.1:16587
10.0.42.2:17202
10.0.42.2:19001
10.0.42.3:32006

10.0.42.16
10.0.42.2:17202
10.0.42.2:19001
10.0.42.3:32006

130.3.18.39
130.3.18.39:20001
130.3.18.39:20002
130.3.18.39:20003
130.3.18.39:20004

ftp server
ftp server
telnet
web server

NAT Traversal for SIP

Network Address and Port Translators
- Temporary address and port mappings
- Private IP addresses for SOHOs
  (e.g. multiplexing single dialup-connection)
- First line of defense in common firewall configurations

Problems
- Violation of the Internet’s end-to-end-principle
- NATs break many existing protocols

Solutions
- Midcom: control protocol for middle boxes (long term)
- ALG-functionality in NATs (unlikely for most protocols)

→ Solution: ICE
SIP Devices

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, …
◆ HTTP server for manual user configuration
◆ Sometimes web browsers
◆ Autoconfiguration: DHCP, (t)ftp, …
Autoconfiguration / Administration

◆ Key to large scale SIP deployments in enterprises
◆ Key to real plug & play products & broad deployment

◆ Aspects include
  – IP layer management
  – User and Device Profile Management
  – Software updates

◆ SIP Device Config Framework
  – Locate profile provisioning server
  – Register (authenticate) and retrieve user / device profiles
  – Receive notification of updates to profiles

◆ SIP Device Requirements
  – Collects expected functionality and behavior
  – Users, service providers, network admins, manufacturers, integrators
  – Representation of configuration data

Some SIP Phones from our Lab
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 Support for Mobility

- Personal Mobility
  - SIP URIs
  - SIP registration and authentication
  - SIP call routing
  - SIP routing to services

- Terminal Mobility
  - Ubiquitous IP connectivity (+ roaming)
  - Mobile IP (orthogonal to SIP)
  - Dynamic SIP session re-routing
  - 3G link layer roaming mechanisms

- Service Mobility
  - Dynamic SIP session re-routing
  - 3G Hand-off procedures

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 Core Protocol

- UDP
- TCP, SCTP, TLS
- IP/IP Multicast

SIP-related Components in 3GPP

- Visited network
- Home network

- UE
- P-CSCF
- I-CSCF
- S-CSCF
- SLF
- HSS
- AS
- BCGF
- MGCF
- MGW
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

User A

P-CSCF

GPRS

Home A

I-CSCF

S-CSCF

HSS

User B

Home B

SIP Registration of a Mobile Node

UA

P-CSCF

I-CSCF

S-CSCF

HSS

IP address + P-CSCF assign

GPRS/DHCP

REGISTER

DNS Lookup

401 Unauthorized

401 Unauthorized

DNS Lookup

REGISTER

200 OK

200 OK

Visited Network

Home Network

User query

Auth.

User query

register
3G Roaming Scenario: Simple Call

User A

1. User A sends an INVITE message to the P-CSCF in Home A.

2. The P-CSCF forwards the INVITE message to the I-CSCF in Home A.

3. The I-CSCF forwards the INVITE message to the S-CSCF in Home B.

4. The S-CSCF forwards the INVITE message to the I-CSCF in Home B.

5. The I-CSCF forwards the INVITE message to the P-CSCF in Home A.

Simple SIP Call: Caller Side (1)

UA

- INVITE
  - 100 Trying
  - authorize resources
  - 183 Progress
  - PRACK
  - Resource reservation
  - 200 OK

- UPDATE
  - 200 OK

- UPDATE
  - 200 OK

P-CSCF

- INVITE
  - 100 Trying
  - authorize resources
  - 183 Progress
  - PRACK

S-CSCF

- INVITE
  - 100 Trying
  - authorize resources
  - 183 Progress
  - PRACK

Visited Network

- 200 OK

Home Network

- 200 OK

validate

100 Trying

183 Progress

PRACK

200 OK

183 Progress

PRACK

200 OK

To called entity
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

Use of Signaling Compression for SIP

“Separate” SIP Calls
No End-to-End Semantics!

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 real 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.softarmor.com/sipwg/
http://www.softarmor.com/sippingwg/
http://www.softarmor.com/simple/

http://www.cs.columbia.edu/sip/
http://www.cs.columbia.edu/sip/sipit/