Outline

- **Multimedia is different**
- **Real Time Protocol (RTP)**
- **Session Description Protocol (SDP)**
- **Session Initiation Protocol (SIP)**

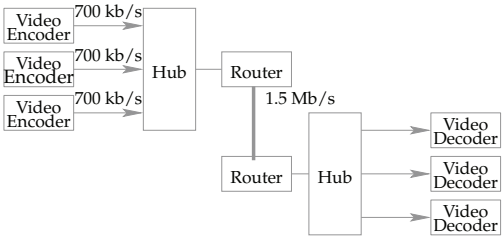
Elastic vs. Inelastic Workloads

- **Some applications adapt to network performance**
  - Examples: file transfer, printing.
  - Will perform the same task regardless of bandwidth
  - Such workloads are *elastic*—they adapt fine to lower performance
- **Other apps have traffic performance requirements:**
  - Example: video requires a certain available bandwidth
  - Workloads that fail if requirements not met are *inelastic*

Why is this an issue?

- **Most data networks provide *best effort delivery***
  - Try their best to deliver your data, but no guarantees
- **Traditional data applications:**
  - Use protocols such as TCP to deal with data loss and unknown network available bandwidth,
  - Tend to generate data in bursts, and
  - Normally do not require any kinds of guarantees from the network
- **Elastic workloads will often swamp inelastic ones**
  - BitTorrent can make Skype sound terrible
- **Traditional networks have not been designed with *Quality Of Service* (QoS) in mind**

Where do things break?



- **Things break when there is a contention for resources**
- **Service may be unacceptable for all**

Example of a Network with QoS: The Phone Network

- **Before you can communicate**
  - Must negotiate a channel with the network
  - If network lacks the resources, your call is not completed
- **If call is completed, you "own" circuit**
  - Yours for the duration of the call
  - Regardless of the additional traffic in the network
- **Circuit has appropriate capacity for voice**

QoS Components

- **Must know traffic characteristics and requirements**
  - Must describe them to the network to reserve bandwidth
- **Need *signaling protocol* betw. nodes & net. mgmt**
- **The Network must implement Admission Control**
  - Reject connections exceeding available bandwidth
- **The Network may also implement Traffic Policing**
  - Make sure traffic emitted by the nodes conform to the agreed parameters

Where to Provide?

- **1990s and early 2000s saw a lot of work on getting QoS into layer 3: Resource ReSerVation Protocol (RSVP)**
  - Fine for walled garden networks, e.g., IP-telephony backbones
  - Consumer applications do not want to assume QoS support
- **There's a lot more to multimedia than simple QoS**
  - Session establishment, media negotiation
- **Higher layers need to be involved**
  - Transport: Real Time Protocol (RTP)
  - Session: Session Description Protocol (SDP)
  - Application: Session Initiation Protocol (SIP)

Outline

- **Multimedia is different**
- **Real Time Protocol (RTP)**
- **Session Description Protocol (SDP)**
- **Session Initiation Protocol (SIP)**

RTP [RFC 3550]

- **Provides end-to-end delivery services for data with real-time characteristics**
  - E.g. interactive audio and video
  - Services include: Source & payload type identification, Sequence numbering, Time-stamping, Delivery monitoring
- **Typically used on top of UDP**
  - Relies on UDP for multiplexing & checksums
- **Work over other network or transport protocols**
  - Lower-level must provide framing & length indication

RTP continued

- **Supports data transfer to multiple destinations**
  - Uses network-level multicast if available
- **Does *not* ensure timely delivery/QoS guarantees**
  - Relies on lower-layer services to do so
- **Does *not* guarantee in-order delivery**
- **Does *not* even guarantee delivery**
  - Underlying network need not be reliable
  - Underlying network need not deliver packets in sequence
  - RTP sequence numbers determine proper location of packet

Two parts to RTP

1. **RTP**
   - Carries data that has real time properties
2. **RTCP**
   - Monitors quality of service
   - Conveys information about participants in a session (for "loosely controlled" sessions)

RTP protocol framework

- **RTP is not a complete protocol**
  - It is a protocol *framework*, deliberately not complete
  - Tailored to applications through modification, not options
- **Each applications needs a *profile specification***
  - defines set of payload type codes
  - defines mapping of payload types to payload formats
- **Also need actual payload format specification**

RTP Session

- **Association among a set of participants communicating with RTP**
- **A session is defined by a particular pair of destination transport addresses**
  - One network address
  - A pair of ports (one for RTP and one for RTCP)
  - May be common to all participants (as in IP multicast)
  - May be different for each (individual network address + common port address)

Synchronization Source (SSRC)

- **Source of a stream of RTP packets**
  - Randomly chosen 32-bit numeric identifier
  - Intended to be globally unique
  - Carried in RTP header (not dependent on network address)
- **Timing & sequence number space are per-SSRC**
  - Lets receiver separately handle packets from same source
  - RTCP sender/receiver reports are per SSRC
- **One participant may use multiple SSRCs**
  - Should use one for each stream
  - Different streams may have different media clock rates
  - Or one stream may switch encodings mid-stream
- **Binding of SSRCs is provided through RTCP**
  - E.g., if participant uses multiple cameras in one session

RTP Translators and Mixers

- **Intermediate act as gateways at RTP layer**
  - Allow RTP traffic to pass through firewalls
  - Mix and/or recode data to fit over a low bandwidth link
- ***Translators* leave original SSRC intact**
  - So sources distinct even when all packets from translator
  - Translator may change payload type or combine packets
- ***Mixers* combine streams from multiple sources**
  - Output requires new SSRC with new seq/timestamps
  - Original sources conveyed in each packet using "contributing sources" (CSRC) list

RTP Packet

- **Fixed RTP packet header**
- **List of contributing sources (possibly empty)**
- **RTP Payload**
  - E.g. audio samples, compressed video data, etc. . . .

RTP Data Header

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

RTP Data Header details

- **Padding (P, 1 bit)**
  - Packet ends w. padding octets ending in padding length
  - E.g., useful when encrypting with block cipher
- **Extension (X, 1 bit)**
  - If set, fixed header followed by header extension
- **CSRC Count (CC, 4 bits)**
- **Profile-specific Marker (M, 1 bit)**
  - E.g., might indicate frame boundary

RTP Data Header details continued

- **Payload Type (PT, 7 bits)**
  - RFC 3551 defines some default audio/video types
  - But actual interpretation depends on profile
  - Can even define some types dynamically
- **Sequence Number (16 bits)**
  - Initial value random
  - Increments for each RTP *packet* (not byte) sent
  - Used by receiver to detect packet loss & misordering
- **Time Stamp (32 bits)**
  - Sampling instant of start of payload
  - Resolution depends on data format, must be sufficient for synchronization & measuring jitter

RTP Control Protocol (RTCP)

- **Uses separate port from data**
  - Save monitoring tools from sorting through data packets
  - For UDP, use even port n for RTP, and $n + 1$ for RTCP (So good idea for NATs to preserve even/odd port parity)
- **Multiple RTCP messages sent in *compound packets***
  - Reduces packet rate, saves bandwidth & processing cost
  - Figure out # segments based on packet length
- **Compound packets start w. *reception report***
  - Losses in multicast distribution can be quickly isolated
  - Senders can adapt to current network conditions

RTCP Bandwidth Allocation

- **Application decides bandwidth needed for session**
  - Included in session announcement or inferred by scope
- **Control b/w should be fixed fraction of total**
  - RTP currently recommends 5% of total session bandwidth
  - Profile can specify some other fraction
- **RTCP b/w kept constant by varying report interval**
  - Track number of active senders and receivers
  - Senders get 25% of total RTCP b/w, receivers the rest
  - Minimum report interval 5 seconds to avoid bursts on small sessions
  - Actual interval randomized to avoid synchronization

RTCP Receiver Report (RR)

V=2	P	RC	PT=RR=201	LENGTH
SSRC OF SENDER				
SSRC_1 (SSRC OF FIRST SOURCE)				
FRACTION LOST			CUMULATIVE NUMBER OF PACKETS LOST	
EXTENDED HIGHEST SEQUENCE NUMBER RECEIVED				
INTERARRIVAL JITTER				
TIME OF LAST SR (LSR)				
DELAY SINCE LAST SR (DLSR)				
. . . ADDITIONAL RECEPTION REPORTS. . .				
. . . PROFILE-SPECIFIC EXTENSIONS. . .				

RTCP Sender Report (SR)

V=2	P	RC	PT=SR=200	LENGTH
SSRC OF SENDER				
NTP TIMESTAMP, MOST SIGNIFICANT WORD				
NTP TIMESTAMP, LEAST SIGNIFICANT WORD				
RTP TIMESTAMP				
SENDER S PACKET COUNT				
SENDER S OCTET COUNT				
. . . RECEPTION REPORTS . . .				
. . . PROFILE-SPECIFIC EXTENSIONS. . .				

- **Superset of RR, used when sender has sent RTP packets**

RTCP Source Description (SDES)

V=2	P	SC	PT=SDES=202	LENGTH
SSRC / CSRC_1				
SDES ITEMS FOR SSRC / CSRC_1				
. . . ADDITIONAL SDES CHUNKS . . .				

	LENGTH	USER AND DOMAIN NAME . . .
	LENGTH	COMMON NAME OF SOURCE. . .
	LENGTH	EMAIL ADDRESS OF SOURCE. . .

Canonical End-Point Identifiers

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    CNAME=1    |     length    | user and domain name        ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **SDES that must appear in every compound packet**
- **CNAME identifies each participant uniquely**
  - Stays constant even if SSRC changes (by conflict or restart)
  - Binds streams from multiple media tools to same source
  - For monitoring, intended to be human-readable as well
  - Translators should translate RFC 1918 addresses to ensure global uniqueness

Analyzing RTCP Reports

- **Cumulative counts for long- and short-term analysis**
  - Subtract any two reports to get activity over interval
  - NTP timestamps in reports allow you to compute rates
  - Monitoring tools needn't understand data encodings
- **Sender reports give utilization information**
  - Average packet rate and average data rate over any interval
  - Monitoring tools can compute this without seeing the data
- **Receiver reports give loss and round-trip information**
  - Extended sequence number conveys # packets expected
  - Packets lost and packets expected give long term loss rate
  - Fraction lost field gives short-term loss rate
  - LSR and DLSR give senders ability to compute round-trip time

RTP Profiles

- **Provide interpretation of generic fields**
  - Mapping from payload type number to encodings
  - Use of marker bits
  - Frequency of timestamp counter
- **Other items which may be specified in a profile**
  - RTP header extensions
  - Additional RTCP packet types
  - RTCP report interval
  - Use of security/encryption
- **No support for parameter negotiation, membership control**
  - Protocols such as SDP and SIP handle this

Outline

- **Multimedia is different**
- **Session Description Protocol (SDP)**
- **Real Time Protocol (RTP)**
- **Session Initiation Protocol (SIP)**

SDP [RFC 4566]

- **Originally designed for multimedia multicast**
  - Session directory tool advertises multimedia conferences
  - Must communicate the conference addresses
  - Must communicate app-specific information necessary for participation.
- **SDP is designed to convey such information**
- **Can use multiple transport protocols including:**
  - SAP (Session Announcement Protocol)
  - Email using MIME extensions
  - HTTP

Session Information conveyed by SDP

- **Session name and purpose**
- **Time(s) the session is active**
  - Arbitrary list of start/stop times
  - Repeat times (e.g., every Tuesday at 2:45pm)
- **The media that constitute the session**
  - Type (audio, video), format, protocol
  - Network (possibly multicast) address, protocol, port
- **Other useful info:**
  - Bandwidth required
  - Contact information for a responsible person
  - etc.

Session Descriptions

- **Compact, but entirely text-based**
  - Lines of the form: $type = \langle value \rangle$
  - Facilitates embedding in various transport methods
- **Begins with session-level section**
  - Starts with line "texttv=0" (version 0)
  - Contains lines that apply to all media streams
  - Ends at first "`m=`..." line
- **Followed by zero or more media-level descriptions**
  - Starts with "`m=`...", ends before next "`m=`..." line
  - Can contain directives overriding session-level section
- **With some transports, can concatenate sessions**
  - Each "`v=0`" starts a new session description

Session Description Syntax

- **Session description (* = optional)**
  - `v=` protocol version
  - `o=` owner/creator and session number
  - `s=` session name
  - `i=*` session information
  - `u=*` URI of description
  - `e=*` email address
  - `p=*` phone number
  - `c=*` connection information—optional if in all media
  - `b=*` bandwidth information
  - >> one or more time descriptions
  - `z=*` time zone adjustments
  - `k=*` encryption key
  - `a=*` zero or more session attribute lines
  - >> zero more media descriptions

Session Description Syntax continued

- **Time description**
  - `t=` time the session is active
  - `r=*` zero or more repeat times
- **Media description**
  - `m=` media name and transport address
  - `i=*` media title
  - `c=*` connection info.—optional defined at session level
  - `b=*` bandwidth information
  - `k=*` encryption key
  - `a=*` zero or more session attribute lines
- **The type set is small, not extensible**
  - SDP parsers must ignore unknown announcement types
- **Use attributes for media-specific details**

SDP Example

- **Example session description:**

```
v=0
o=cs144-staff 2890844526 2890842807 IN IPv4 171.16.64.4
s=SDP Lecture
i=A Leecture on the session description protocol
u=http://cs144.scs.stanford.edu/notes/l13.pdf
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 3456 RTP/AVP 0
m=video 2232 RTP/AVP 31
m=whiteboard 32416 udp wb
a=orient:portrait
```

Outline

- **Multimedia is different**
- **Session Description Protocol (SDP)**
- **Real Time Protocol (RTP)**
- **Session Initiation Protocol (SIP)**

Session Initiation Protocol [RFC 3261]

- **SIP is a protocol designed to enable the invitation of users to participate in multimedia session**
  - Not tied to a specific conference control scheme
  - Supports loosely or tightly controlled sessions
  - Enables user mobility by relaying and redirecting invitations to a user's current location
- **Communication is between *users*, not hosts**
  - User identifiers define control path (whom to ask about user)
  - Data path (actual media) can be completely decoupled

Session Initiation Protocol (2)

- **SIP is a *control* protocol for creating/modifying/ terminating sessions w. one or more participants**
- **Examples of sessions are:**
  - Internet telephone calls
  - Internet multimedia conferences
  - Internet multimedia distribution
- **Communication among members in a session may be:**
  - Via multicast,
  - Via a mesh of unicast "relations"
  - Or via a combination of both

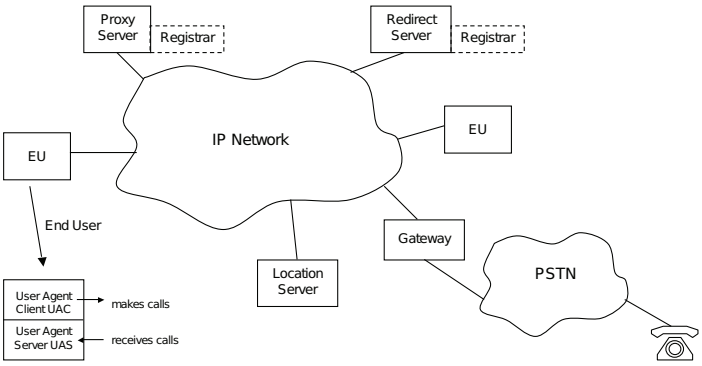
Functional Features

- **Allows participants to agree on a set of compatible media types through SDP**
- **Supports user mobility by proxying and redirecting requests to the user's current location.**
- **Can be extended with additional capabilities.**
- **It is not tied to any particular conference control protocol**
- **It is independent of the lower layer transport protocol**

Call Setup

- **Initial phase**
  - Client tries to find address at which to contact remote user
- **Subsequent phases**
  - Implement request-response protocol
  - Session description is sent with an invitation to join
- **Status code responses**
  - Informational – request received, continuing process
  - Success – action received, understood, and accepted
  - Redirection – client must undertake further action to complete request
  - Client error – request contains bad syntax
  - Server error – server couldn't complete valid request

Big Picture



Addressing

- **SIP addresses are URLs: user@host**
  - user: user name or telephone number
  - host: domain name or numeric IP address
- **Examples:**
  - `sip:dm@scs.stanford.edu`
  - `sip:16505555555@sip.future-nine.com`
- **To send a message, a SIP client can send it to a pre-configured proxy, or use DNS:**
  - Check for DNS SRV records
  - Then check for MX records
  - Finally, use an A record

Components (1)

- **Clients**
  - End systems
  - Send SIP requests
  - Usually also contain SIP user agent server (UAS), which listens for call requests, prompts user or executes program to determine response
- **Proxy server**
  - Proxies request to another server
  - Possibly translates and rewrites request
  - Can "fork" request to multiple servers, creating search tree

Components (2)

- **Redirect Server**
  - Redirects users to try another server (user agent may act as redirect server)
- **Location Server (or service)**
  - Used by SIP redirect or proxy server to obtain information about a user's possible location(s)
  - May be co-located with a SIP server but the manner in which a SIP server requests location services is beyond the scope of SIP
  - May be anything (LDAP, whois, local file, result of program execution)

Components (3)

- **Registrar**
  - A server that accepts REGISTER requests
  - Typically co-located with a proxy or redirect server
  - Allows a client to let the proxy or redirect server know at which address(es) it can be reached

Methods (1)

- **There are 6 methods in SIP**
- **Invite**
  - Invites a participant to a conference
  - Conference can be unicast, multicast, new or in existence
- **Bye**
  - Ends a client's participation in a call
- **Cancel**
  - Terminates a search
- **Options**
  - Queries a participant about their media capabilities, and finds them, but doesn't invite

Methods (2)

- **Ack**
  - Call acceptance
  - Reliability
- **Register**
  - Informs a SIP server about the location of a user

Responses (1)

- **1xx – Informational**
  - Request received, continuing to process request
  - Examples: 100 trying, 180 ringing, 181 call is being forwarded, 182 queued
- **2xx – Success**
  - Action successfully received, understood, and accepted
  - Example: 200 OK
- **3xx – Redirection**
  - Further action must be taken in order to complete the request
  - Examples: 300 multiple choices, 301 moved permanently, 302 moved temporarily, 305 use proxy

Responses (2)

- **4xx – Client error**
  - Request contains bad syntax
  - Examples: 400 bad request, 401 unauthorized, 402 payment required
- **5xx – Server error**
  - Server failed to fulfill an apparently valid request
  - Examples: 500 internal server error, 501 not implemented, 502 bad gateway
- **6xx – Global failure**
  - The request cannot be fulfilled at any server
  - Examples: 600 busy everywhere, 604 does not exist everywhere, 606 not acceptable

Message Syntax

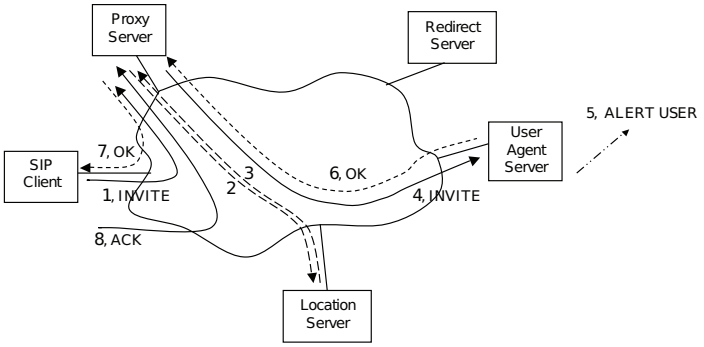
- **Text based**
- **Many header fields from http**
- **Some new ones**
  - Via
- **Payload may contain media description**
  - typically uses SDP, Session Description Protocol

```
INVITE sip:dm@scs.stanford.edu SIP/2.0
From: sip:pal@scs.stanford.edu
To: dm@scs.stanford.edu
Call-ID: 19990321@scs.stanford.edu
Cseq: 10 INVITE
v=0
o=user1 12345 6789 IN IP4 171.0.1.2
s=Multimedia Networks
i=Presentation Multimedia Networks and
Communication
e=pal@scs.stanford.edu
c=IN IP4 224.2.0.1/127
t=0 0
m=audio 3456 RTP/AVP 0
```

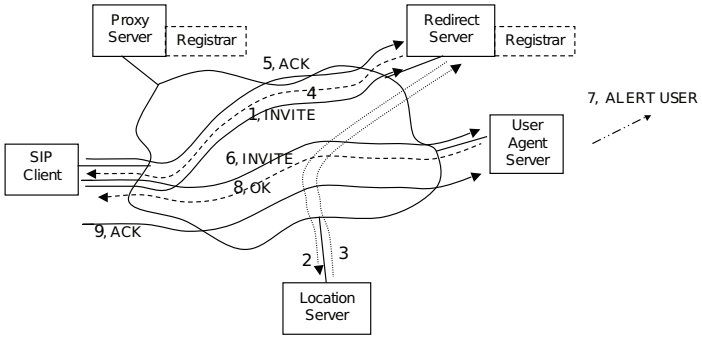
Basic Operation

- **Client sends req. to locally configured proxy**
  - Or obtains domain server IP address (using DNS)
- **Call initiator contacts SIP server for domain**
- **Location server locates receiver**
- **Call is established**
  - Initiator sends an INVITE request
  - Invited party answers (agrees)
  - Initiator receives OK indication
  - Initiator sends an ACK request

Proxy Example



Redirect Example



Other Features

- **Multiple call acceptances**
  - Client control
  - Client selection
  - Multiparty conferencing
- **Security**
  - Encryption and authentication end-to-end
  - Uses existing mechanisms
- **Messaging**
  - SIP is behind AIM and many other IM systems
  - SIMPLE: SIP for Instant Messaging and Presence Leveraging Extensions (simple)

Services

- **Call forwarding**
- **Hold**
- **Blind call transfer**
- **Transfer with notification**
- **Operator assisted transfer**
- **Full mesh unicast conferences**
- **Multicast conferences**
- **Authenticated Caller ID**
- ***66**
- **Local services: call waiting, mute, *69**