4: Application Protocols: SMTP and others

Last Modified:
2/3/2003 8:07:08 PM

Electronic Mail

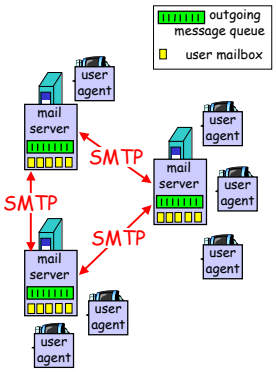
Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: smtp

User Agent

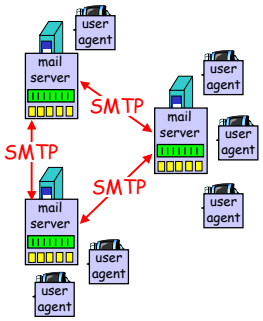
- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Netscape Messenger
- outgoing, incoming messages stored on server



Electronic Mail: mail servers

Mail Servers

- mailbox contains incoming messages (yet to be read) for user
- message queue of outgoing (to be sent) mail messages (if message cannot be delivered will stay in queue)
- smtp protocol between mail servers to send email messages
  - Mail server is an SMTP client when sending mail
  - Mail server is an SMTP server" when receiving mail



Electronic Mail: smtp [RFC 2821]

- Uses tcp to reliably transfer email msg from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
  - Much like HTTP

SMTP History

- SMTP has been around a long time
  - RFC done in 1982
  - In use well before that
- Messages must be in 7-bit ASCII (made sense in text-based early days)
- Requires encoding for binary data (jpegs, etc.) in 7-bit ASCII (yuck!)

Sample smtp interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C:   How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```


try smtp interaction for yourself:

- `telnet servername 25`
- see 220 reply from server
- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands

above lets you send email without using email client (reader)

- How do you know the right server name?

Trace it – does your mail data go in the clear?

What is missing?

- Some commands processed by SMTP protocol mirror mail headers we are used to seeing in our email messages (To, From, …), but are not the same things
- Email headers (To, From, CC, Subject, Date, ..) are considered part of the data by SMTP and are not processed SMTP server at all!
- Email headers are processed by the mail reader software and ignored by SMTP
  - How is Bcc implemented?
- Another example of "protocol" layering (like HTML and HTTP)

Mail message format

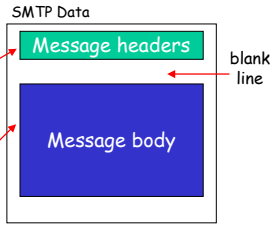
smtp: protocol for exchanging email msgs

RFC 2822: standard for text message format (format of data from smtp perspective)

- header lines, e.g.,
  - To:
  - CC:
  - Subject:

different from SMTP commands!

- body
  - the "message", ASCII characters only



Sample smtp interaction

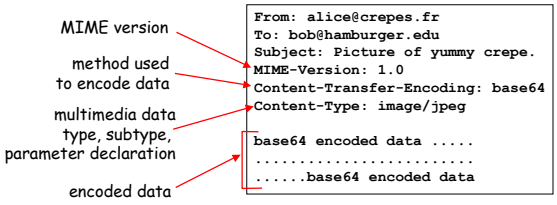
```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: To: bob@hamburger.edu
C: Subject: dinner preferences
C: From: alice@crepes.fr
C:
C: Do you like ketchup?
C:   How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```


SMTP format

- SMTP requires that message (header & body) be in 7-bit ascii
- Certain character strings are not permitted in message (e.g., `CRLF.CRLF`). Thus message has to be encoded (usually into either base-64 or quoted printable)
- SMTP server uses `CRLF.CRLF` to determine end of message

What about sending pictures and other binary data?

- Don't try this by hand ☺
- MIME: multimedia mail extension, RFC 2045, 2056
- additional lines in msg header declare MIME content type



MIME types: Extensible

`Content-Type: type/subtype; parameters`

Text
- example subtypes: `plain`, `html`

Image
- example subtypes: `jpeg`, `gif`

Audio
- example subtypes: `basic` (8-bit mu-law encoded), `32kadpcm` (32 kbps coding)

Video
- example subtypes: `mpeg`, `quicktime`

Application
- other data that must be processed by reader before "viewable"
- example subtypes: `msword`, `octet-stream`

Multipart Type

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=98766789

--98766789
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain

Dear Bob,
Please find a picture of a crepe.
--98766789
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.....base64 encoded data
--98766789--
```


Spam/forged mail

- "Received:" and "MessageID" headers are part of the data
  - Accurate and helpful from legitimate servers and user agents
- Start with a legitimate server you trust
  - Don't relay messages from a site outside your domain to another host outside your domain
  - Verify the Mail From field (resolvable domain and matching IP address)
  - Refuse traffic from known spammers

Sample Spam

From dogboyseven@aol.com Sat Sep 4 16:55:41 1999
Received: from cs2.CS.Berkeley.EDU (cs2.CS.Berkeley.EDU [169.229.60.56])
by mammoth.CS.Berkeley.EDU (8.9.1a/) with ESMTP id QAA20836 for <jnm@mailspool.CS.Berkeley.EDU>;
Sat, 4 Sep 1999 16:55:38 -0700 (PDT)

Received: from mail.everfaster.com (mail.everfaster.com [197.46.220.4])
by cs2.CS.Berkeley.EDU (8.9.1a/8.6.6.Beta11) with ESMTP id LAA18735 for <jnm@cs.berkeley.edu>; Sat, 4
Sep 1999 16:55:04 -0700 (PDT)

Received: from **gate.hypermoon.com** (pool37.qs4w.longlink.net [217.6.1.7])
by mail.everfaster.com (8.8.7/8.8.7) with SMTP id PAA20074; Sat, 4 Sep 1999 19:54:21 -0400 (EDT)

Received: from fritz.hotdogcity.com (fritz.hotdogcity.com [221.88.9.16])
*by **server.big-hello.com** (8.8.8/8.8.8) with SMTP id RAA04617; Sat, 4 Sep 1999 19:53:33 -0400 (EDT)*

Received: by fritz.hotdogcity.com with Internet Mail Service (5.5.248.0)
id Q196494F; Sat, 4 Sep 1999 19:53:25 -0400 (EDT) Date: Sat, 4 Sep 1999 19:53:23 -0400 (EDT)
From: Charles Lewis <clewis@hotmail.com>
To: jnm@cs.berkeley.edu
Subject: You'll never believe this!
Message-ID: <19990904195323.H8159@fritz.hotdogcity.com>
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii

You won't believe this, but some company just paid me to surf the web! Check out...

Tracking and Reporting Spam

- Record IP address of sender and time and date of message transfer
- Spamcop uses a combination of tools like dig, nslookup and finger to cross-check all the information in an email header and find the email address of the system administrator responsible for the network from which the mail was sent
- postmaster@domain or abuse@domain

Multiple recipients

- When you send mail to your outgoing mail server, transfer one copy of message regardless of how many recipients
  - Great for spammers ☹
- Mail servers could play the same trick
  - Look at RCPT to list
  - If more than one recipient per destination mail server then transfer just one mail
- Could also send one copy per recipient
  - Recommended configuration?

Email viruses

- Often attachments which once opened run with the users full privileges and corrupt the system on which mail is read
- Viruses tend to target Windows as it is the platform used by the majority of people

SMTP vs HTTP

- Smtp: persistent connections like HTTP 1.1
- Both have ASCII command/response interaction, status codes
- http: each object is encapsulated in its own response message
- smtp: multiple objects message sent in a multipart message
- http: pull; smtp: push

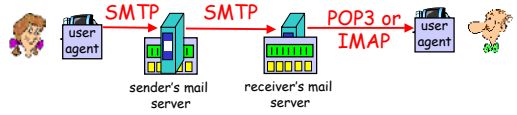
Outgoing Mail Server?

- Why not just SMTP server on local machine?
- "Push not pull" means your PC must be constantly on to accept "push"

SMTP = outgoing

- Notice we didn't see any SMTP commands to "get" or "retrieve" mail
- SMTP is for outgoing mail only
- How do we get mail?
  - Early days: log on to server and read mail from a mailbox = file on server
  - How many people still read mail that way? (I do ☺)
  - Today many people read mail on their PC
  - How do they get their mail from the mail server?

Incoming mail?



- Mailbox file
- POP: Post Office Protocol [RFC 1939] authorization (agent <-->server) and download
- IMAP: Internet Mail Access Protocol [RFC 1730] more features (more complex) manipulation of stored messages on server
- HTTP: Hotmail , Yahoo! Mail, etc.
  - Why not use HTTP to transfer random things like email?
  - Convenient – don't need mail reader just the ubiquitous web browser
- Other?

POP3 protocol

authorization phase
- client commands:
  - `user:` declare username
  - `pass:` password
- server responses
  - +OK
  - -ERR

transaction phase, client:
- `list:` list message numbers
- `retr:` retrieve message by number
- `dele:` delete
- `Quit`

```
S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK user successfully logged on

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```


try POP interaction for yourself:

- `telnet servername 110`
- see "OK POP3 server ready" reply from server
- enter user, pass, list, retr, dele commands

above lets you send get you own email without using email client (reader)

Trace it – do your password *and mail data* go in the clear?

Do you configure your mail reader to pop mail every X minutes? Same as announcing your password regularly!

IMAP

- Allows user to set up and maintain multiple folders (for sorting mail) on the remote server
- Can get headers for and manipulate messages without downloading them (can even download individual MIME attachments)
  - Don't pay cost to download over slow link
  - Don't leave them on insecure computers
- Stateful protocol - stores per user information about folders and the status of the messages in them
  - Folder information, actual messages
  - Seen, Deleted, Answered flags per message

IMAP con't

- During an IMAP connection, the server transitions between multiple states
  - Initially non-authenticated
  - Authenticated
  - Selected – folder selected and operations on messages permitted
  - Finally, Logout state

Authentication in IMAP

- Client requests a certain AUTHENTICATION method
  C: A001 AUTHENTICATE KERBEROS_V4
- If server implements that authentication mechanism then it will authenticate via that method
  S: + AmFYig==
  C: BAcAQU5EUkVXLkNNVS5FRFUAOCAsho84kL
  N3/IJmrMG+25a4DT+nZImJjnTNHJUtxAA+o0KPKfH
  EcAFs9a3CL5Oebe/ydHJUwYFd
  S: + or//EoAADZI=
  C: DlAF5A4gA+oOIALuBkAAmw==
  S: A001 OK Kerberos V4 authentication successful
- Sever can respond with NO if it does not support that authentication mechanism
  S: A001 NO authenticate failure

Authentication in IMAP (cont)

- Client can try various authentication mechanisms in decreasing order of preference looking for one the server supports
- In the worst case, a client may authenticate with plain text login
  C: a001 LOGIN SMITH SESAME
  S: a001 OK LOGIN completed

Once authenticated, client can:

- SELECT a mailbox
  - C: A142 SELECT INBOX
  - S: * 172 EXISTS S: * 1 RECENT
  - S: * OK [UNSEEN 12] Message 12 is first unseen
  - S: * OK [UIDVALIDITY 3857529045] UIDs valid
  - S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
  - S: * OK [PERMANENTFLAGS (\Deleted \Seen *)] Limited
  - S: A142 OK [READ-WRITE] SELECT completed
- CREATE, RENAME or DELETE mailboxes
- FETCH messages from a mailbox
- SEARCH through messages
- APPEND messages to a mailbox

Pop vs IMAP

- Similarities
  - Mail delivered to a shared, constanly connected server
  - New mail accessible anywhere in network on a variety of platforms
  - For access only, Need SMTP to send mail
- Differences
  - POP simpler and more established (more clients and servers that support it)
  - IMAP is stateful protocol with more features; POP uses less server resources
  - IMAP = prioritize download time; POP = shorter overall connection time

Network News

Thanks to Jeffrey Vinocur (NNTP presentation, Spring 2002)

What is Usenet?

- Reading/posting to Usenet newsgroups
- Conceptually: a semi-organized collection of forums ("newsgroups") for public discussion
- Technically: a system for distributing email-like messages

Usenet Messages

- Format: like email, but a bit stricter and with some extra headers (e.g., Newsgroups) – we don't care about this today, except for two important headers
- Message-ID: unlike email, every message truly needs to have a globally unique identifier
- Path: we'll see this header later

Path: news.litech.org!lnsnews.lns.cornell.edu!paradoxa.ogoense.net!not-for-meow
From: meowbot@meowing.net (A Meowbot)
Newsgroups: alt.dev.null
Subject: Why?
Date: Sun, 27 Jan 2002 23:25:52 +0000 (UTC)
Organization: a tyranny of meowing fascist censor cabalists
Lines: 4
Approved: nope.
Message-ID: <mW.3C548C72.8BC5@K0deZ.scriptkiddie.net>
X-Trace: paradoxa.ogoense.net 1012173952 6565 141.154.205.147 (27 Jan 2002 23:25:52 GMT)
X-Complaints-To: abuse@ogoense.net
X-Meow: Wouf
Mail-Copies-To: nobody
X-No-Repost: yes
Xref: news.litech.org alt.dev.null:492

Because we like you.

--
Meow

Network Topology

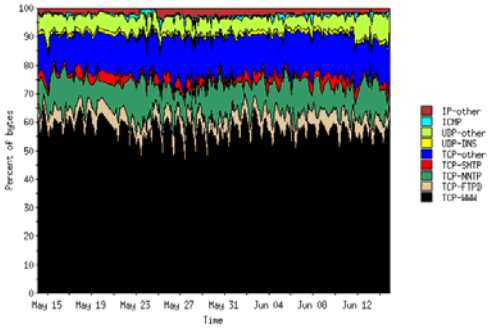
- Users connect to a local site
- Each site may have several servers for better throughput
- Sites are connected by (manually-requested and -configured) peering links to other sites
- Major sites have hundreds of peers

So I post…then what?

- The goal is for every article to make it to every server in the world – the "floodfill" model
- This can be as fast as a few seconds or as long as a few days (normally a few hours)

Serious bandwidth

Credit:
CAIDA (1999)



An article arrives…

This can be either a new post from a user or an article being "fed" from a peering server.

1. The server's "name" added to the Path header (history of where the article has been)
2. The server stores the article so users can read it
3. For each of the server's peers, determine if the peer has seen the article already (first check for peer's name in Path header, then ask the peer about the Message-ID)
4. Send the article to peers who do not have it

Path headers and Message-IDs

- Let's trace an article. The initial component (at the end!) of the Path header marks the original posting server; then the originating server adds its name:

Path: paradoxa.ogoense.net!not-for-meow

- Then this article gets fed to a another server and then add their hostname:

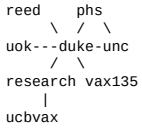
Path: lnsnews.lns.cornell.edu!paradoxa.ogoense.net!not-for-meow

- And then it gets fed to another server…

Path: news.litech.org!lnsnews.lns.cornell.edu!paradoxa.ogoense.net!not-for-meow

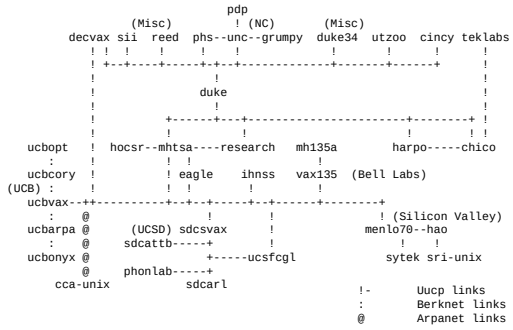
Usenet, 1980

Credit:
Mark Horton



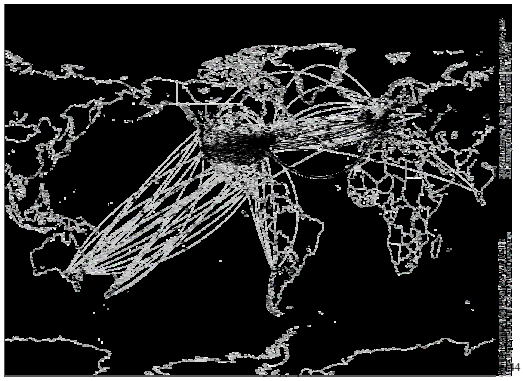
Usenet, 1981

Credit:
ucbvax!mark



Usenet, 1993

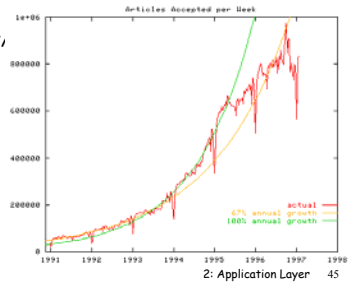
Credit:
Brian Reid



Usenet today

Credit:
Karl L. Swartz

- 1.4 million articles daily
- ~ 360 GB daily
- Over a 100 Mbit/ link is > 8 hours!

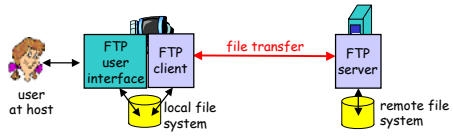


Usenet is like a herd of performing elephants with diarrhea – massive, difficult to redirect, awe-inspiring, entertaining, and a source of mind-boggling amounts of excrement when you least expect it.

– Professor Gene Spafford, Purdue University

FTP

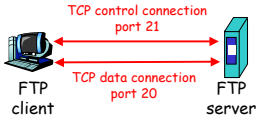
ftp: the file transfer protocol



- transfer file to/from remote host
- client/server model
  - *client:* side that initiates transfer (either to/from remote)
  - *server:* remote host
- ftp: RFC 959
- ftp server: port 21

ftp: separate control, data connections

- ftp client contacts ftp server at port 21, specifying TCP as transport protocol
- two parallel TCP connections opened (both full duplex):
  - control: exchange commands, responses between client, server. "out of band control"
  - data: file data to/from server, can be used in either direction, need not always exist
- ftp server maintains "state": current directory, earlier authentication



ftp commands, responses

Sample commands:

- sent as ASCII text over control channel
- `USER username`
- `PASS password (sent in clear text!)`
- `LIST` return list of file in current directory
- `RETR filename` retrieves (gets) file
- `STOR filename` stores (puts) file onto remote host

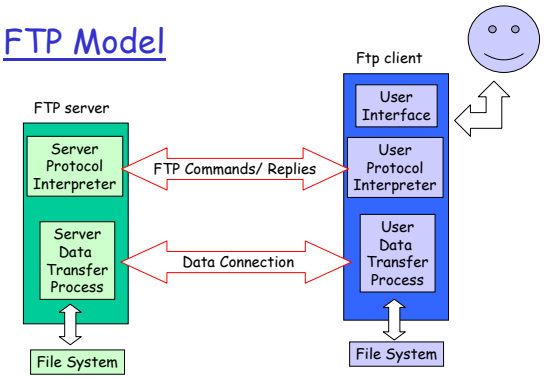
Sample return codes

- status code and phrase (as in http)
- `331 Username OK, password required`
- `125 data connection already open; transfer starting`
- `425 Can't open data connection`
- `452 Error writing file`

FTP Connection

- Client connects to port 21 on server; this established the control channel
- Over the control channel, the client specifies the characteristics including IP address and port number for data connection (note: needn't be on the same machine as the client)
- Can ask server to set up a passive connection for the data port as well (good for getting around firewalls)
- Server tries to connect to data port as specified by client
- Once established data connection can be used in both directions

FTP Model



Multimedia Applications

Multimedia Applications

- Audio/video conferencing, streaming audio, etc.
  - On-demand playback: could download before beginning playback; could support rewind, fast forward etc.; start-up time and RTT not very important
  - Live transmission: usually broadcast from one source like TV or radio; much like on demand; no rewind or fast forward; more sensitive to delay (how close to live?)
  - Conferencing: interactive, start-up time and RTT matter alot
- Examples: vic (video conferencing), vat (audio conferencing), RealAudio, Quicktime, WindowsMedia

Requirements of multimedia

- Several methods for compressing and encoding voice/video; sender and receiver negotiate
- Ability to display stream (at degraded quality) with lost packets
- Ability to specify the timing requirements between packets of related data for smooth playback
- Frame boundary indication
- Synchronization of related audio and video streams
- No retransmission of lost packets

Real-time Transport Protocol (RTP)

- TCP overhead to high; UDP not good enough
- Initially, each application had its own protocol, implementing only those parts of TCP it really needed on top of UDP
- RTP offers generalized real time transport services
  - Thin protocol; Runs on top of UDP
  - Implements functionality commonly needed by multimedia applications - timing reconstruction, loss detection, security and content identification
  - RFC 1889

Realtime Transport (?) Protocol

- Is this an application level protocol or a transport protocol?
  - Done at application level
  - If TCP implemented at application level (good project ☺), does that make it an application level protocol or a transport level protocol?
- Where is the right place to put these features?

Real-time Streaming Protocol (RTSP)

- Network "Remote Control"
  - Like FTP has data channel and control channel; RTSP is the control channel for streaming audio/video
  - Not used to deliver data; often uses RTP for the data portion
- Establishes and controls audio and video delivery
  - Single or multiple audio/video streams (time synchronization if desired)
  - Live feeds or stored clips
- Industry consortium announced in 1996 – since then?
  - Mostly development continued on proprietary versions: Real Network's (originally Progressive Networks) RealMedia, RealAudio and RealPlayer , Quicktime, WindowsMedia???

RTSP Requests

- DESCRIBE – description of presentation
- OPTIONS - get supported methods; capability announcements
- SETUP – establish a new session
- PLAY – start playback/streaming; reposition
- ANNOUNCE – change description of presentation
- RECORD – start recording
- REDIRECT – redirect client to a new server; for load balancing
- PAUSE –stop delivery but keep state
- TEARDOWN – stop delivery, remove state

Trying RTSP

- `telnet servername 554`

C: **DESCRIBE rtsp://streamserver/rafile.rm RTSP/1.0\n\n**

S: **RTSP/1.0 200**

Trying RTSP (2)

```
C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
Transport: rtp/udp; compression; port=3056; mode=PLAY
S: RTSP/1.0 200 1 OK
Session 4231
C: PLAY rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=0-
C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi
   RTSP/1.0
Session: 4231
Range: npt=37
C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi
   RTSP/1.0
Session: 4231
S: 200 3 OK
```


RTSP vs HTTP

- RTSP actually derived from HTTP
  - Avoid mistakes (like always specify full URI)
  - More methods of course
- RTSP server needs to maintain state from SETUP to control PLAY command; HTTP server is stateless (uses cookies to trick client into remembering it)
- Data can be delivered in or out of band with RTSP; HTTP data delivered in band
- RTSP is a symmetric protocol (client and server can both isssue requests); HTTP client issues requests
  - Ex. server can announce new available streams (audio from a new participant in a conference)

Session Description Formats

- Format for describing the number and sources for all streams in a presentation
- May offer alternatives
  - Different audio channels in various languages
  - Different quality of audio/video for various BW connections
- Specify timing requirements between various streams
- Examples: SDF, SDP

SDP example

```
session (v 0)(o mhandley 2890844526 2890842807 IN IP4
   126.16.64.4)
(s Sd seminar)(i A seminar on the session description protocol)
(u http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.01.ps)
 (e M.Handley@cs.ucl.ac.uk (Mark Handley))
(c IN IP4 224.2.17.12/127)(t 2873397496 2873404696)
(a recvonly) (all (media (m audio 3456 VAT PCMU))
 (media (m video 2232 RTP H261))
(media (m whiteboard 32416 UDP WB)(orient portrait)) ))
```

From:
http://www.cs.columbia.edu/~hgs/rtsp/sdf.html

From URL in web page to streaming audio/video

<EMBED SRC="http://server/foo.sdf" TYPE = "application/x-audio">

- HTTP gets session or presentation description file (not part of RTSP) from a web server
- Presentation Description indicates RTSP server to contact
  - Note: RTSP is presentation description format neutral
- RTSP sets up a stream to control delivery
- RTSP used to indicate server that will actually stream the data and by what protocol
  - Ex. specify an RTP server to deliver the data
- Note: possibly 3 servers involved!

Alternative: HTTP Streaming

- Many sites simply send audio and video over HTTP
- When object arrives will be opened by appropriate application just like Doc files or PDF files
- Estimate when it is safe to begin playback without the playback outpacing the download
- Download mode and a limited streaming mode can be supported this way
  - Rewind? Fast forward?
  - Can support full streaming if delays ok

Audio and Video on the Internet

- Quicktime
  - HTTP streaming or RTP and RTSP
- RealServer
  - one control channel: RTSP over TCP
  - one data channel: PNA (Progressive Networks Audio) over UDP (?)
  - Also can use RTSP to interleave data and control onto one TCP channel (common configuration)
- WindowsMedia
  - Similar to RealPlayer: control channel and data channel
  - Harder to find details of protocols (surprise, surprise)
  - But formats are not compatible (surprise, surprise)

More Application Level Protocols?

- Telnet, Rlogin, SNMP (Simple Network Management Protocol), Instant Messenger (AIM), DHCP (BOOTP) , RPC, NFS, X,Finger, Whois,IDENT……..
- You now know how to investigate any of these on your own
  - RFCs for open protocols, Run apps and trace them, Get client/server source,…
- It would be a lot more fun to learn more than application level protocols though, right?

Roadmap

- We've looked at a bunch of application level protocols (HTTP, DNS, SMTP, POP, IMAP, NNTP, RTP, ..) – Lessons?
  - Many were human readable – why?
  - High level examples of protocol layering (SMTP, HTTP)
  - Some ran on TCP, some on UDP, one on both – why?
  - Used telnet/nslookup to interact with these protocols more directly
  - Traced them (What went in clear text?!)
- Food-for-thought: Design a "Telephone Protocol"
- Next.. How would we implement an application level protocol ourselves?
  - Socket API
- After that down to transport layer

Outtakes

telnet source

- We've been using telnet to examine various application protocols
- telnet basically opens a TCP connection to the specified port
- Getting the telnet source and examining it would be a good exercise

Real Time Control Protocol (RTCP)

- Real-time conferencing of groups of any size within an internet.
- Provides source identification, quality-of-service feedback from receivers to the multicast group, synchronization of different media streams

ReSerVation Protocol (RSVP)

- Host can use to request specific quality of service from the network for a specific flow of data
- Must be processed and honored at each router to be meaningful
  - Works much like dynamic routing protocols; messages processed by applications at user level
  - If a flow is "admitted" then resource reservation decisions will be made in form of packet classifier and schedulers that will prioritize the use of resources
- Cisco's take on RSVP
  - http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/rsvp.htm