Application Layer

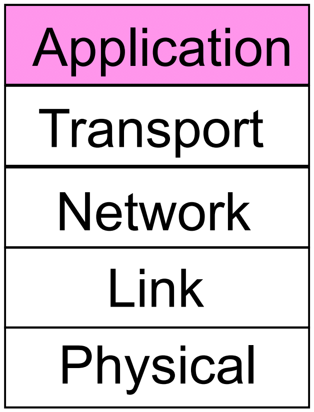
Chapter 7

- DNS – Domain Name System
- Electronic Mail
- The Web
- Streaming Audio and Video
- Content Delivery

Revised: August 2011

The Application Layer

Uses transport services to build distributed applications



DNS – Domain Name System

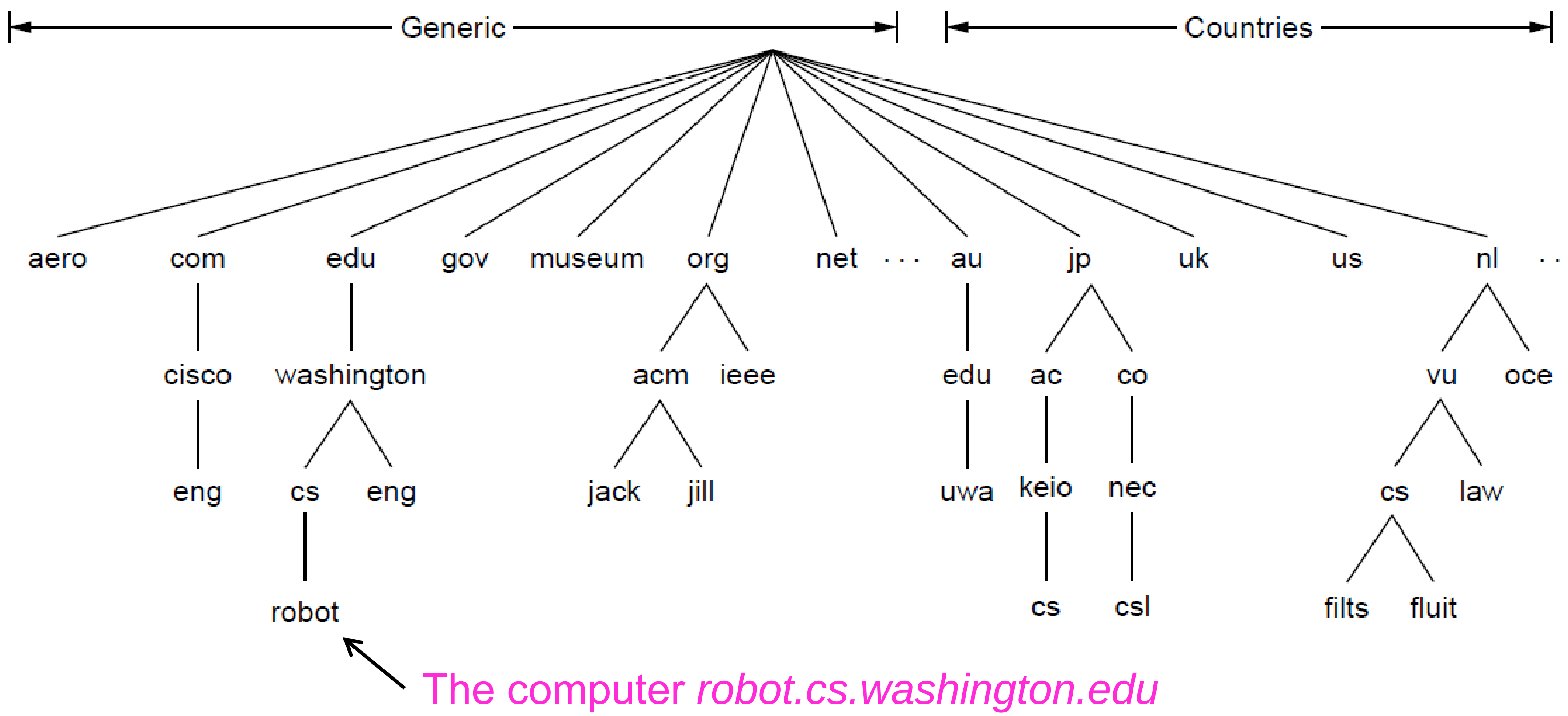
The DNS resolves high-level human readable names for computers to low-level IP addresses

- DNS name space »
- Domain Resource records »
- Name servers »

The DNS Name Space (1)

DNS namespace is hierarchical from the root down

- Different parts delegated to different organizations



The DNS Name Space (2)

Generic top-level domains are controlled by ICANN who appoints registrars to run them

Domain	Intended use	Start date	Restricted?
com	Commercial	1985	No
edu	Educational institutions	1985	Yes
gov	Government	1985	Yes
int	International organizations	1988	Yes
mil	Military	1985	Yes
net	Network providers	1985	No
org	Non-profit organizations	1985	No
aero	Air transport	2001	Yes
biz	Businesses	2001	No
coop	Cooperatives	2001	Yes
info	Informational	2002	No
museum	Museums	2002	Yes
name	People	2002	No
pro	Professionals	2002	Yes
cat	Catalan	2005	Yes
jobs	Employment	2005	Yes
mobi	Mobile devices	2005	Yes
tel	Contact details	2005	Yes
travel	Travel industry	2005	Yes
xxx	Sex industry	2010	No

This one was controversial → (xxx)

Domain Resource Records (1)

The key resource records in the namespace are IP addresses (A/AAAA) and name servers (NS), but there are others too (e.g., MX)

Type	Meaning	Value
SOA	Start of authority	Parameters for this zone
A	IPv4 address of a host	32-Bit integer
AAAA	IPv6 address of a host	128-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
SPF	Sender policy framework	Text encoding of mail sending policy
SRV	Service	Host that provides it
TXT	Text	Descriptive ASCII text

Domain Resource Records (2)

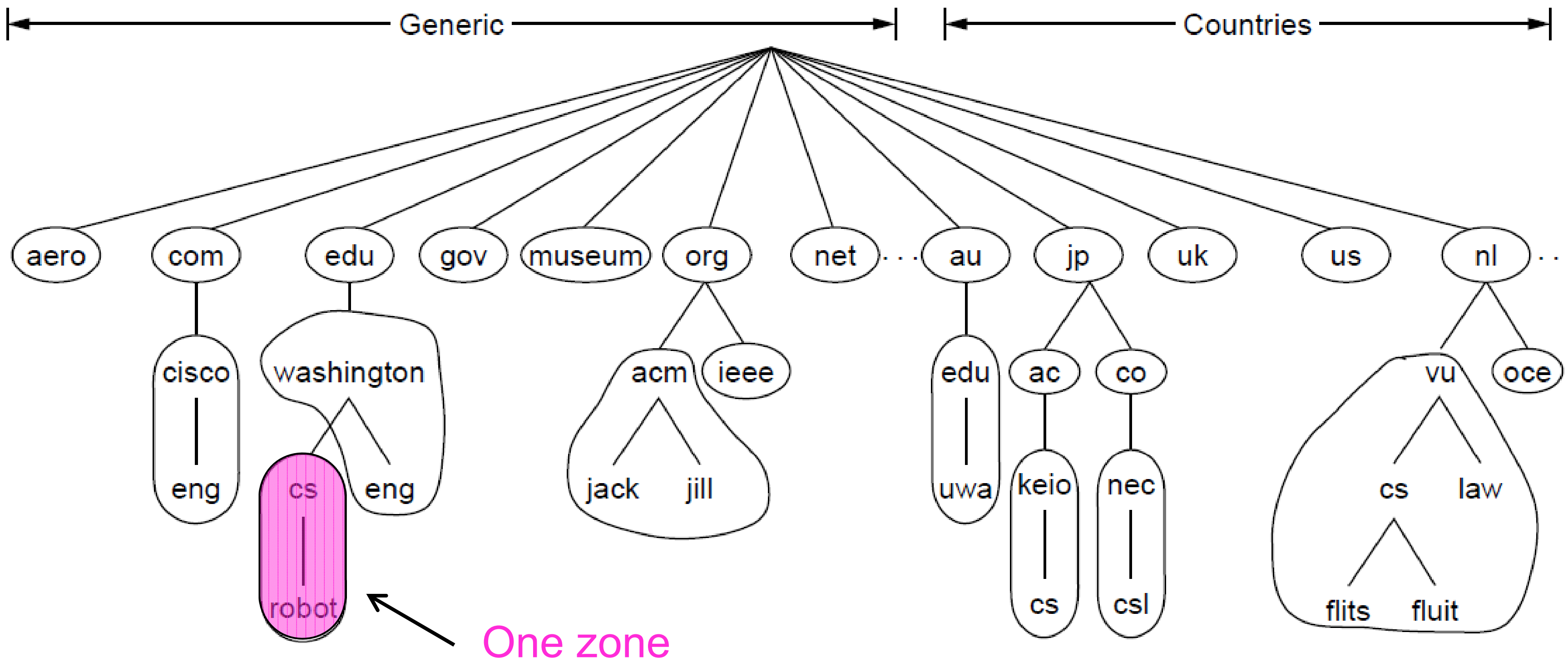
; Authoritative data for cs.vu.nl

cs.vu.nl.	86400	IN	SOA	star boss (9527,7200,7200,241920,86400)	
cs.vu.nl.	86400	IN	MX	1 zephyr	
cs.vu.nl.	86400	IN	MX	2 top	
cs.vu.nl.	86400	IN	NS	star	← Name server
star	86400	IN	A	130.37.56.205	
zephyr	86400	IN	A	130.37.20.10	
top	86400	IN	A	130.37.20.11	← IP addresses of computers
www	86400	IN	CNAME	star.cs.vu.nl	
ftp	86400	IN	CNAME	zephyr.cs.vu.nl	
flits	86400	IN	A	130.37.16.112	
flits	86400	IN	A	192.31.231.165	
flits	86400	IN	MX	1 flits	
flits	86400	IN	MX	2 zephyr	
flits	86400	IN	MX	3 top	
rowboat		IN	A	130.37.56.201	
		IN	MX	1 rowboat	← Mail gateways
		IN	MX	2 zephyr	
little-sister		IN	A	130.37.62.23	
laserjet		IN	A	192.31.231.216	

A portion of a possible DNS database for cs.vu.nl.

Name Servers (1)

Name servers contain data for portions of the name space called zones (circled).



Name Servers (2)

Finding the IP address for a given hostname is called resolution and is done with the DNS protocol.

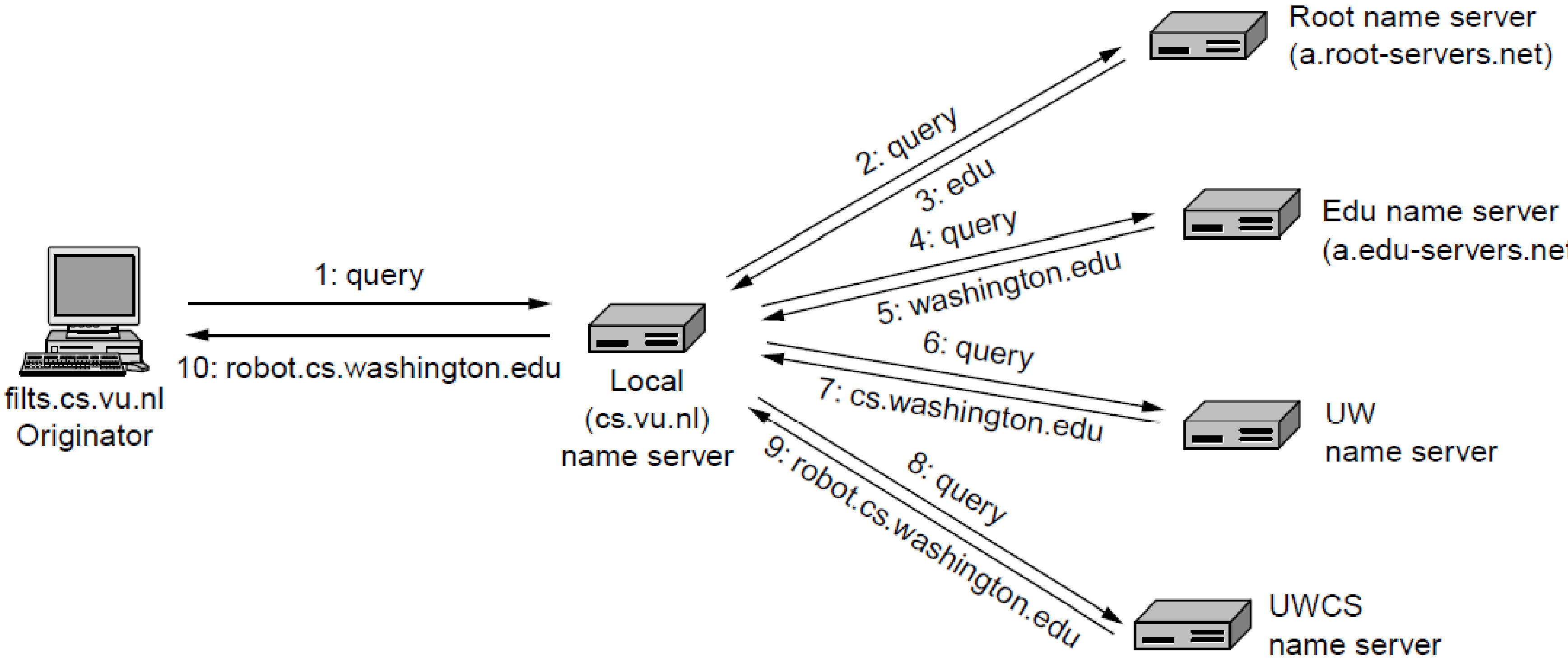
Resolution:

- Computer requests local name server to resolve
- Local name server asks the root name server
- Root returns the name server for a lower zone
- Continue down zones until name server can answer

DNS protocol:

- Runs on UDP port 53, retransmits lost messages
- Caches name server answers for better performance

Name Servers (3)



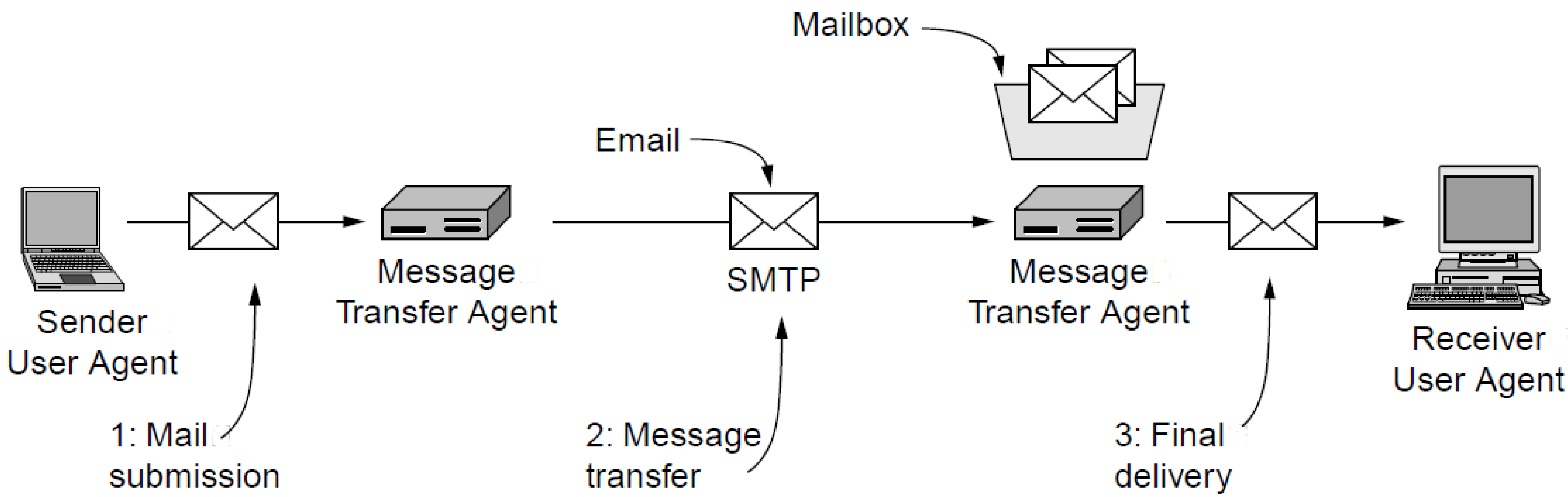
Example of a computer looking up the IP for a name

Electronic Mail

- Architecture and services »
- The user agent »
- Message formats »
- Message transfer »
- Final delivery »

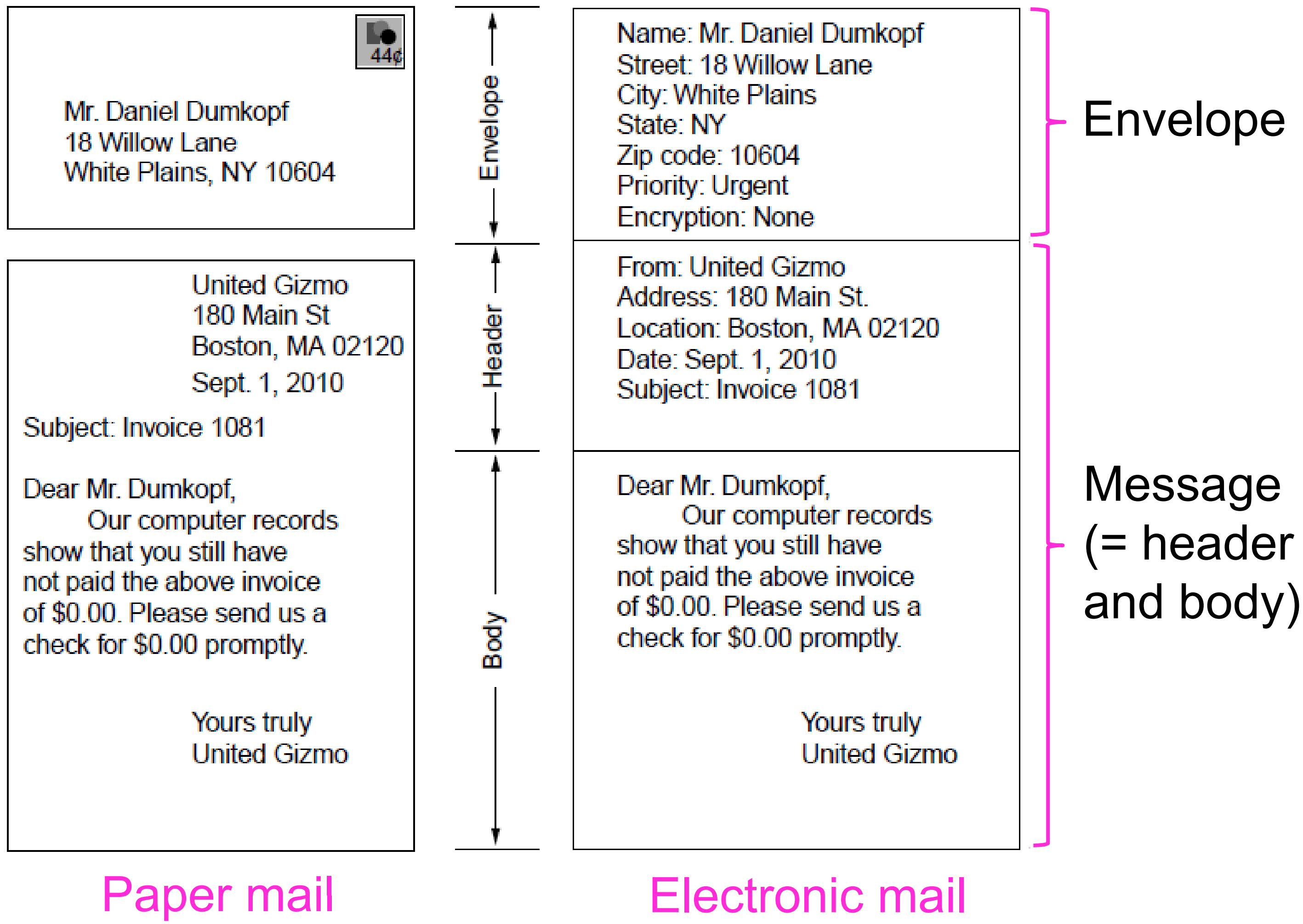
Architecture and Services (1)

The key components and steps (numbered) to send email



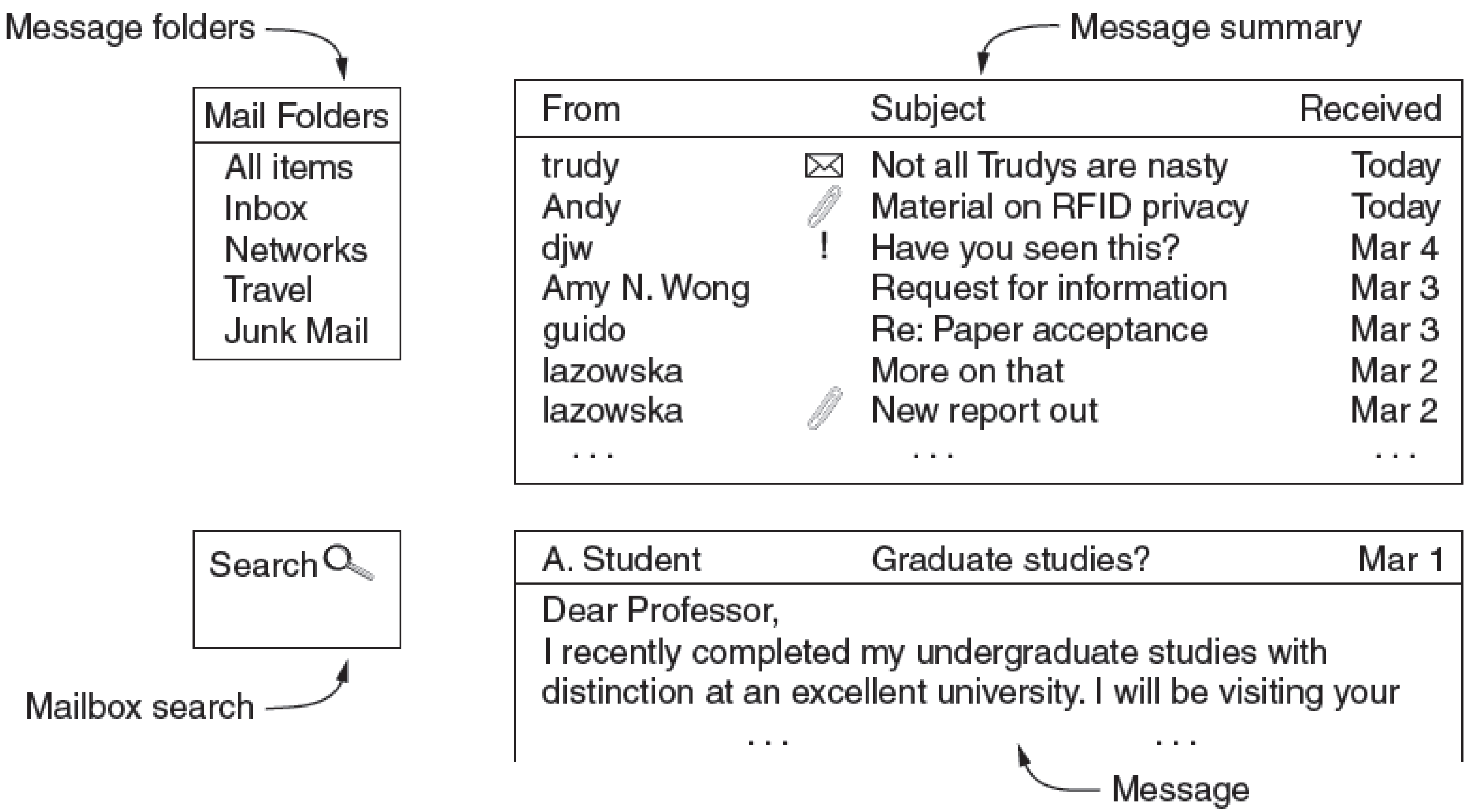
Architecture of the email system

Architecture and Services (2)



The User Agent

What users see – interface elements of a typical user agent



Message Formats (1)

Header fields related to message transport; headers are readable ASCII text

Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Message Formats (2)

Other header fields useful for user agents

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	Email address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

Message Formats (3)

MIME header fields used to describe what content is in the body of the message

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Message Formats (4)

Common MIME content types and subtypes

Type	Example subtypes	Description
text	plain, html, xml, css	Text in various formats
image	gif, jpeg, tiff	Pictures
audio	basic, mpeg, mp4	Sounds
video	mpeg, mp4, quicktime	Movies
model	vrml	3D model
application	octet-stream, pdf, javascript, zip	Data produced by applications
message	http, rfc822	Encapsulated message
multipart	mixed, alternative, parallel, digest	Combination of multiple types

Message Formats (5)

Putting it all together: a multipart message containing HTML and audio alternatives.

```
From: alice@cs.washington.edu
To: bob@ee.uwa.edu.au
MIME-Version: 1.0
Message-Id: <0704760941.AA00747@cs.washington.edu>
Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
Subject: Earth orbits sun integral number of times

This is the preamble. The user agent ignores it. Have a nice day.

--qwertyuiopasdfghjklzxcvbnm
Content-Type: text/html

<p>Happy birthday to you<br>
Happy birthday to you<br>
Happy birthday dear <b> Bob </b><br>
Happy birthday to you</p>

--qwertyuiopasdfghjklzxcvbnm
Content-Type: message/external-body;
        access-type="anon-ftp";
        site="bicycle.cs.washington.edu";
        directory="pub";
        name="birthday.snd"

content-type: audio/basic
content-transfer-encoding: base64
--qwertyuiopasdfghjklzxcvbnm--
```

One part (HTML)

Another (audio)

Message Transfer (1)

Messages are transferred with SMTP (Simple Mail Transfer Protocol)

- Readable text commands
- Submission from user agent to MTA on port 587
- One MTA to the next MTA on port 25
- Other protocols for final delivery (IMAP, POP3)

Message Transfer (2)

Sending a message:

- From Alice to Bob
- SMTP commands are marked [pink]

```
                    S: 220 ee.uwa.edu.au SMTP service ready
C: HELO abcd.com
                    S: 250 cs.washington.edu says hello to ee.uwa.edu.au
C: MAIL FROM: <alice@cs.washington.edu>
                    S: 250 sender ok
C: RCPT TO: <bob@ee.uwa.edu.au>
                    S: 250 recipient ok
C: DATA
                    S: 354 Send mail; end with "." on a line by itself
C: From: alice@cs.washington.edu
C: To: bob@ee.uwa.edu.au
C: MIME-Version: 1.0
C: Message-Id: <0704760941.AA00747@ee.uwa.edu.au>
C: Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
C: Subject: Earth orbits sun integral number of times
C:
C: This is the preamble. The user agent ignores it. Have a nice day.
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: text/html
C:
C: <p>Happy birthday to you
C: Happy birthday to you
  . . . (rest of message) . . .
C: --qwertyuiopasdfghjklzxcvbnm
C: .
                    S: 250 message accepted
C: QUIT
                    S: 221 ee.uwa.edu.au closing connection
```

Message Transfer (3)

Common SMTP extensions (not in simple example)

Keyword	Description
AUTH	Client authentication
BINARYMIME	Server accepts binary messages
CHUNKING	Server accepts large messages in chunks
SIZE	Check message size before trying to send
STARTTLS	Switch to secure transport (TLS; see Chap. 8)
UTF8SMTP	Internationalized addresses

Final Delivery (1)

User agent uses protocol like IMAP for final delivery

- Has commands to manipulate folders / messages [right]

Alternatively, a Web interface (with proprietary protocol) might be used

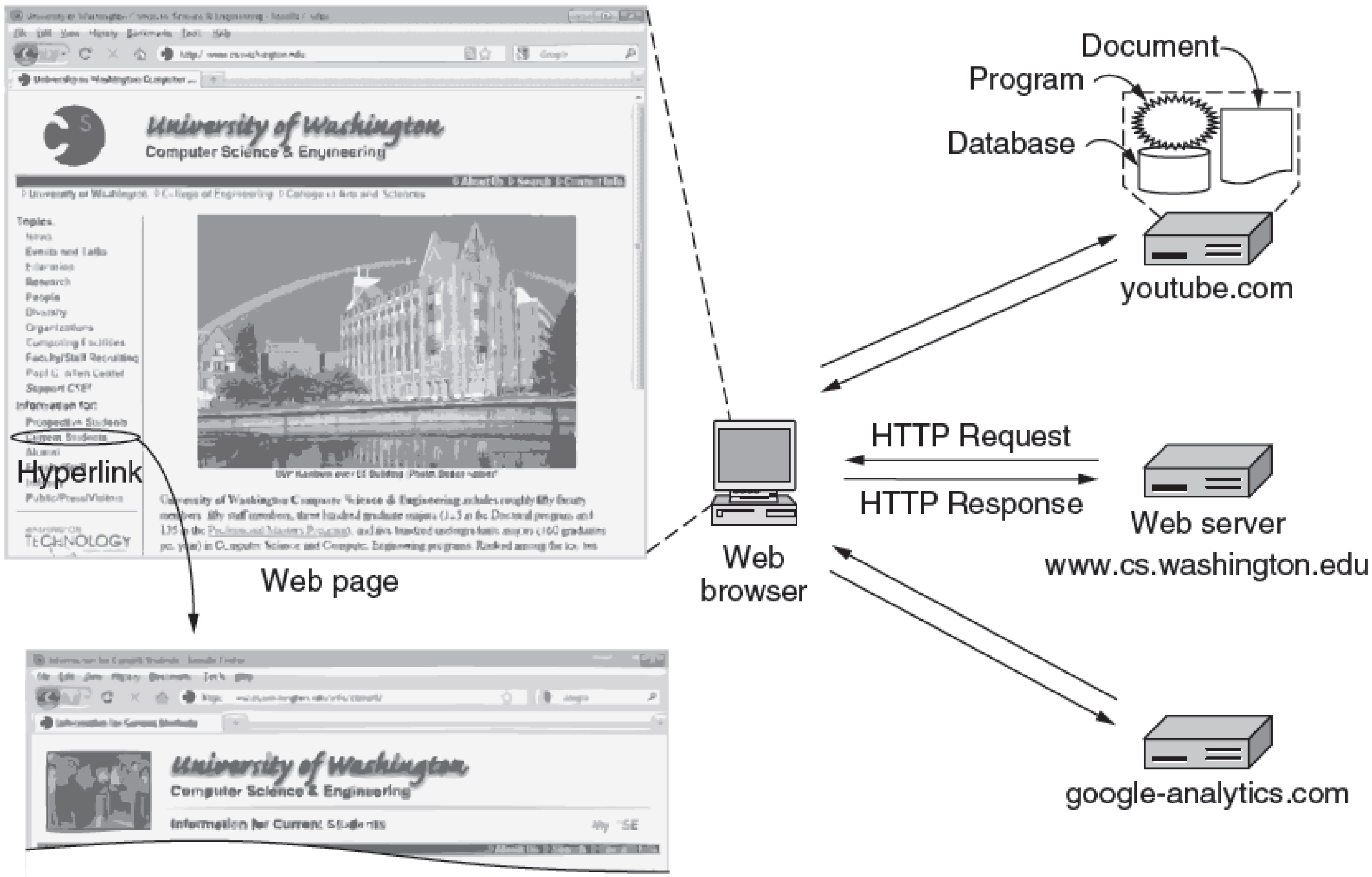
Command	Description
CAPABILITY	List server capabilities
STARTTLS	Start secure transport (TLS; see Chap. 8)
LOGIN	Log on to server
AUTHENTICATE	Log on with other method
SELECT	Select a folder
EXAMINE	Select a read-only folder
CREATE	Create a folder
DELETE	Delete a folder
RENAME	Rename a folder
SUBSCRIBE	Add folder to active set
LIST	List the available folders
LSUB	List the active folders
STATUS	Get the status of a folder
APPEND	Add a message to a folder
CHECK	Get a checkpoint of a folder
FETCH	Get messages from a folder
SEARCH	Find messages in a folder
STORE	Alter message flags
COPY	Make a copy of a message in a folder
EXPUNGE	Remove messages flagged for deletion
UID	Issue commands using unique identifiers
NOOP	Do nothing
CLOSE	Remove flagged messages and close folder
LOGOUT	Log out and close connection

The World Wide Web

- Architectural overview »
- Static Web pages »
- Dynamic pages and Web applications »
- HTTP – HyperText Transfer Protocol »
- The mobile Web »
- Web search »

Architectural Overview (1)

HTTP transfers pages from servers to browsers



Architectural Overview (2)

Pages are named with URLs (Uniform Resource Locators)

- Example: http://www.phdcomics.com/comics.php

Protocol — http
Server — www.phdcomics.com
Page on server — comics.php

Our focus → http

Name	Used for	Example
http	Hypertext (HTML)	http://www.ee.uwa.edu/~rob/
https	Hypertext with security	https://www.bank.com/accounts/
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
file	Local file	file:///usr/suzanne/prog.c
mailto	Sending email	mailto:JohnUser@acm.org
rtsp	Streaming media	rtsp://youtube.com/montypython.mpg
sip	Multimedia calls	sip:eve@adversary.com
about	Browser information	about:plugins

Common URL protocols

Architectural Overview (3)

Steps a client (browser) takes to follow a hyperlink:

- Determine the protocol (HTTP)
- Ask DNS for the IP address of server
- Make a TCP connection to server
- Send request for the page; server sends it back
- Fetch other URLs as needed to display the page
- Close idle TCP connections

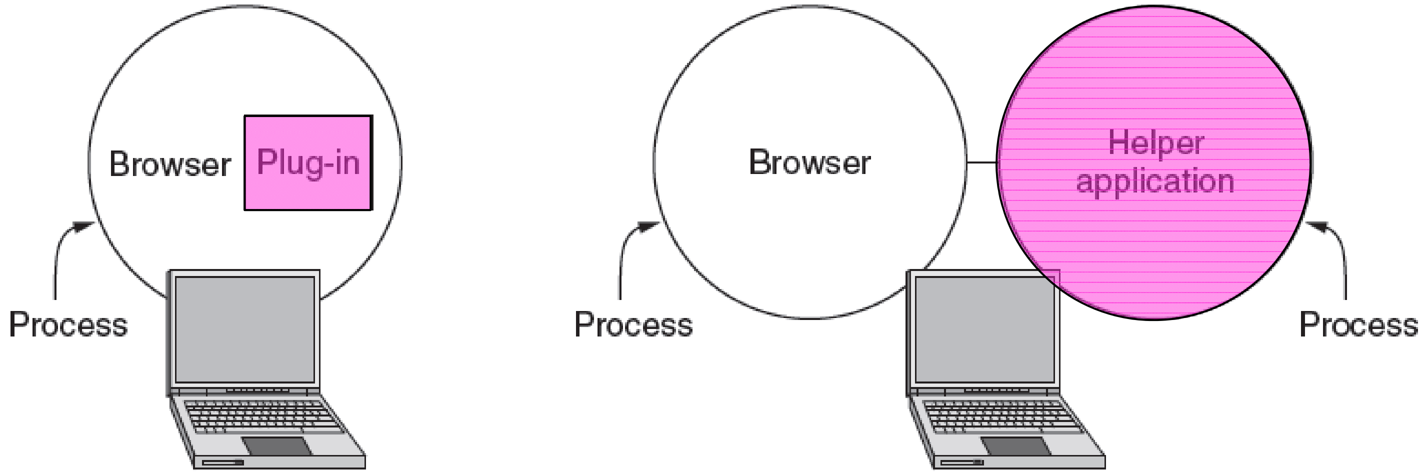
Steps a server takes to serve pages:

- Accept a TCP connection from client
- Get page request and map it to a resource (e.g., file name)
- Get the resource (e.g., file from disk)
- Send contents of the resource to the client.
- Release idle TCP connections

Architectural Overview (4)

Content type is identified by MIME types

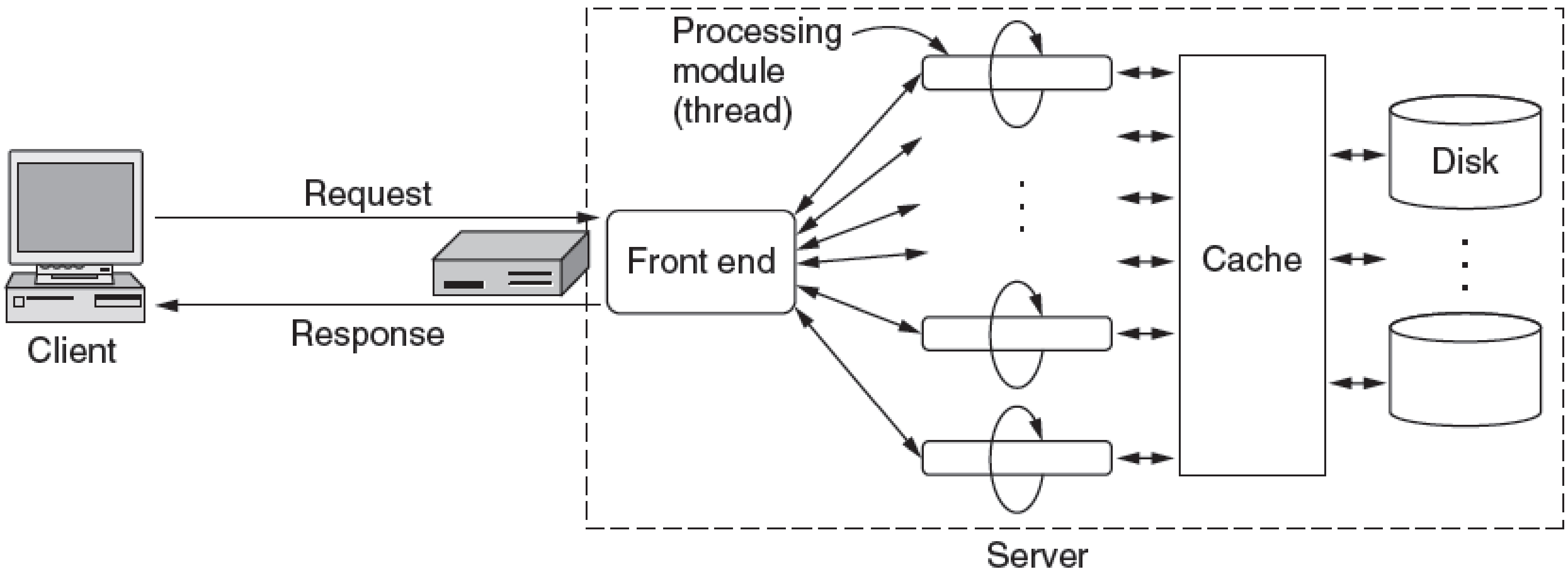
- Browser takes the appropriate action to display
- Plug-ins / helper apps extend browser for new types



Architectural Overview (5)

To scale performance, Web servers can use:

- Caching, multiple threads, and a front end



Architectural Overview (6)

Server steps, revisited:

- Resolve name of Web page requested
- Perform access control on the Web page
- Check the cache
- Fetch requested page from disk or run program
- Determine the rest of the response
- Return the response to the client
- Make an entry in the server log

Architectural Overview (7)

Cookies support stateful client/server interactions

- Server sends cookies (state) with page response
- Client stores cookies across page fetches
- Client sends cookies back to server with requests

Domain	Path	Content	Expires	Secure
toms-casino.com	/	CustomerID=297793521	15-10-10 17:00	Yes
jills-store.com	/	Cart=1-00501;1-07031;2-13721	11-1-11 14:22	No
aportal.com	/	Prefs=Stk:CSCO+ORCL;Spt:Jets	31-12-20 23:59	No
sneaky.com	/	UserID=4627239101	31-12-19 23:59	No

Examples of cookies

Static Web Pages (1)

Static Web pages are simply files

- Have the same contents for each viewing

Can be visually rich and interactive nonetheless:

- HTML that mixes text and images
- Forms that gather user input
- Style sheets that tailor presentation
- Vector graphics, videos, and more (over) . . .

Static Web Pages (2)

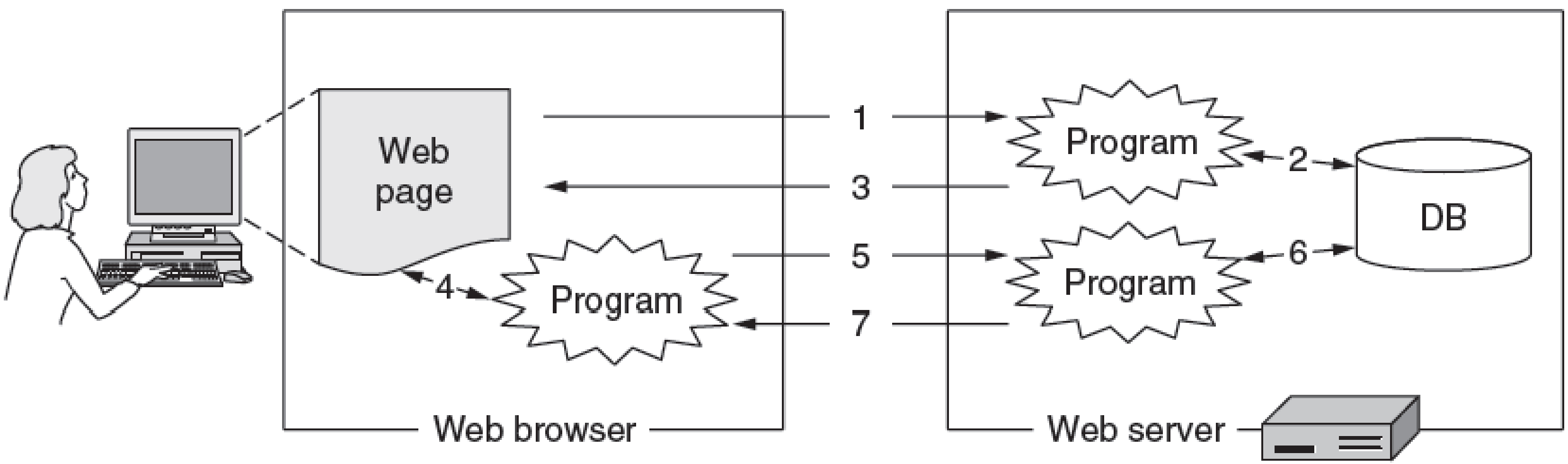
Progression of features through HTML 5.0

Item	HTML 1.0	HTML 2.0	HTML 3.0	HTML 4.0	HTML 5.0
Hyperlinks	x	x	x	x	x
Images	x	x	x	x	x
Lists	x	x	x	x	x
Active maps & images		x	x	x	x
Forms		x	x	x	x
Equations			x	x	x
Toolbars			x	x	x
Tables			x	x	x
Accessibility features				x	x
Object embedding				x	x
Style sheets				x	x
Scripting				x	x
Video and audio					x
Inline vector graphics					x
XML representation					x
Background threads					x
Browser storage					x
Drawing canvas					x

Dynamic Pages & Web Applications (1)

Dynamic pages are generated by programs running at the server (with a database) and the client

- E.g., PHP at server, JavaScript at client
- Pages vary each time like using an application



Dynamic Pages & Web Applications (2)

Web page that gets form input and calls a server program

```
<html>
<body>
<form action="action.php" method="post">
<p> Please enter your name: <input type="text" name="name"> </p>
<p> Please enter your age: <input type="text" name="age"> </p>
<input type="submit">
</form>
</body>
</html>
```

PHP server program that creates a custom Web page

```
<html>
<body>
<h1> Reply: </h1>
Hello <?php echo $name; ?>.
Prediction: next year you will be <?php echo $age + 1; ?>
</body>
</html>
```

PHP calls

Resulting Web page (for inputs “Barbara” and “32”)

```
<html>
<body>
<h1> Reply: </h1>
Hello Barbara.
Prediction: next year you will be 33
</body>
</html>
```

Dynamic Pages & Web Applications (3)

JavaScript program produces result page in the browser

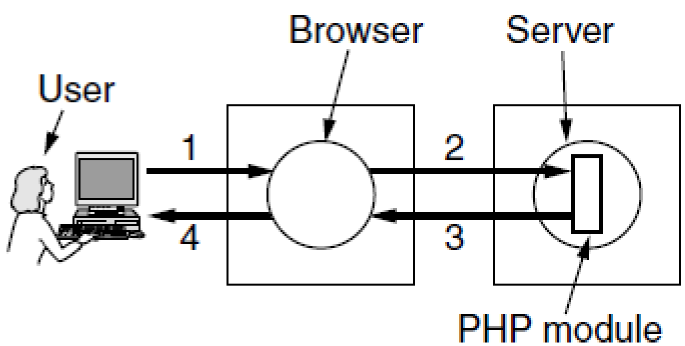
```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
   var person = test_form.name.value;
   var years = eval(test_form.age.value) + 1;
   document.open();
   document.writeln("<html> <body>");
   document.writeln("Hello " + person + ".<br>");
   document.writeln("Prediction: next year you will be " + years + ".");
   document.writeln("</body> </html>");
   document.close();
}
</script>
</head>
```

First page with form, gets input and calls program above

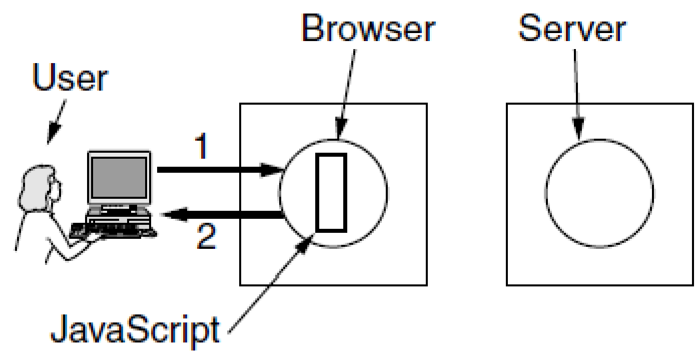
```
<body>
<form>
Please enter your name: <input type="text" name="name">
<p>
Please enter your age: <input type="text" name="age">
<p>
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

Dynamic Pages & Web Applications (4)

The difference between server and client programs



Server-side scripting with PHP



Client-side scripting with JavaScript

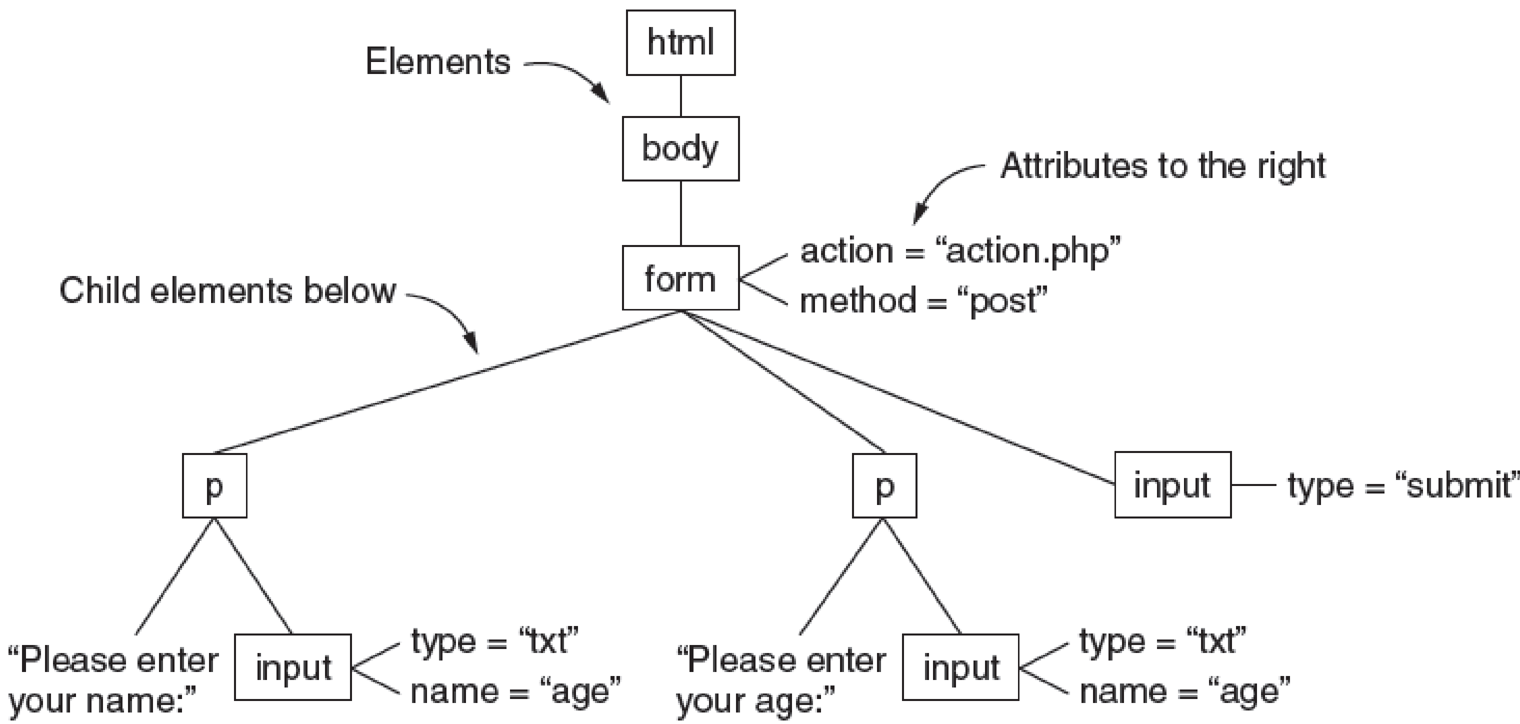
Dynamic Pages & Web Applications (5)

Web applications use a set of technologies that work together, e.g. AJAX:

- HTML: present information as pages.
- DOM: change parts of pages while they are viewed.
- XML: let programs exchange data with the server.
- Asynchronous way to send and retrieve XML data.
- JavaScript as a language to bind all this together.

Dynamic Pages & Web Applications (6)

The DOM (Document Object Model) tree represents Web pages as a structure that programs can alter



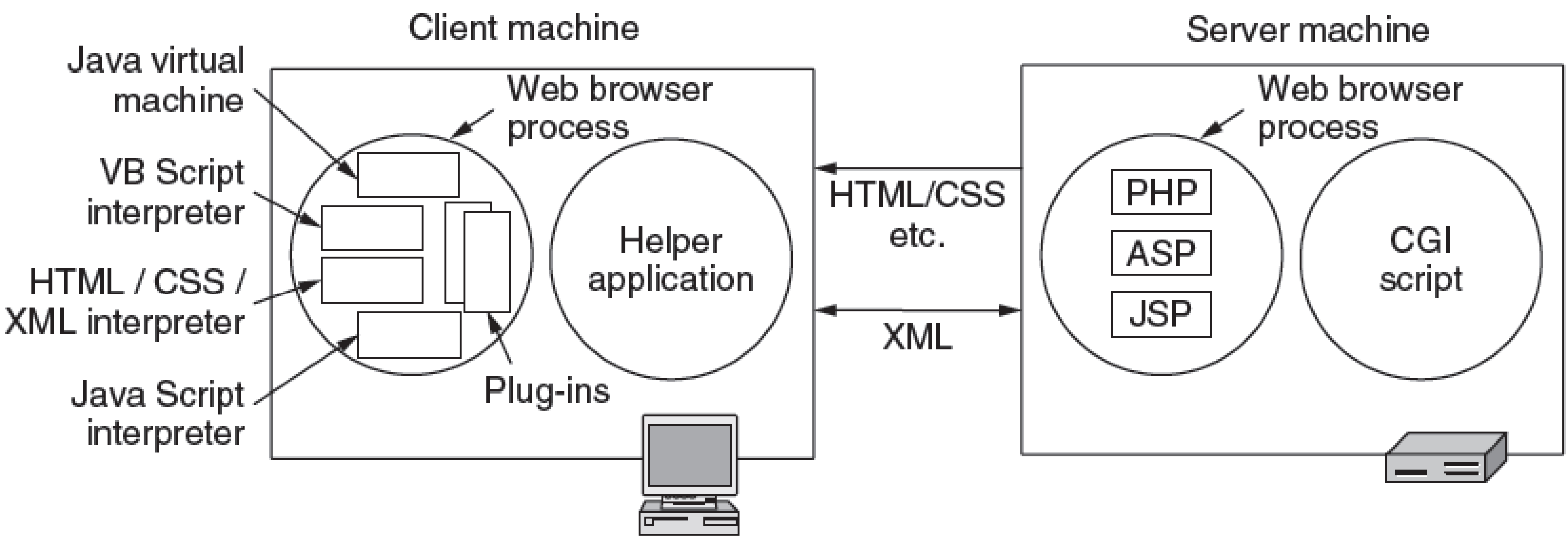
Dynamic Pages & Web Applications (7)

XML captures document structure, not presentation like HTML. Ex:

```
<?xml version="1.0" ?>

<book_list>

<book>
   <title> Human Behavior and the Principle of Least Effort </title>
   <author> George Zipf </author>
   <year> 1949 </year>
</book>

<book>
   <title> The Mathematical Theory of Communication </title>
   <author> Claude E. Shannon </author>
   <author> Warren Weaver </author>
   <year> 1949 </year>
</book>

<book>
   <title> Nineteen Eighty-Four </title>
   <author> George Orwell </author>
   <year> 1949 </year>
</book>

</book_list>
```

Dynamic Pages & Web Applications (8)

Web applications use a set of technologies, revisited:



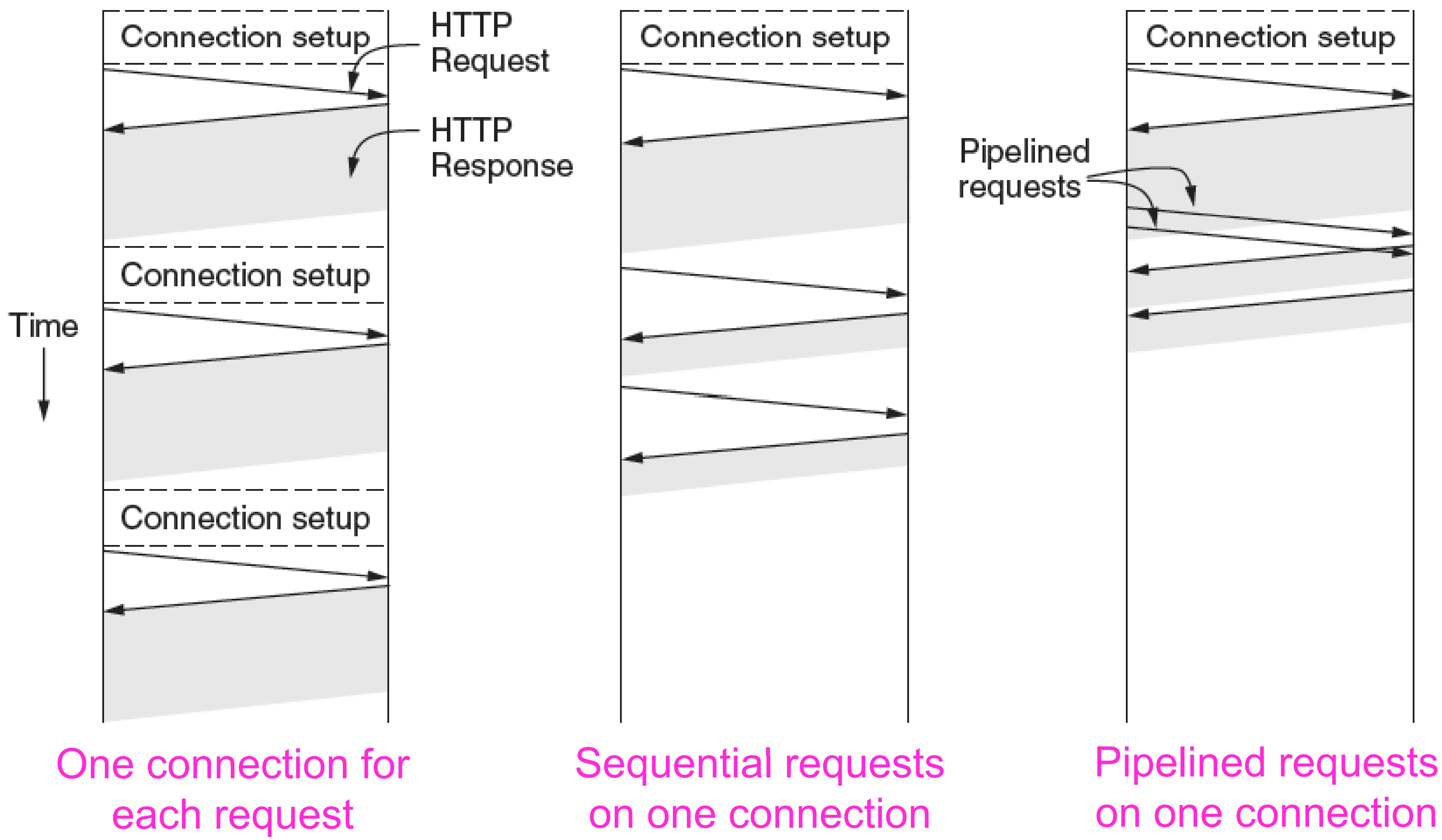
HTTP (1)

HTTP (HyperText Transfer Protocol) is a request-response protocol that runs on top of TCP

- Fetches pages from server to client
- Server usually runs on port 80
- Headers are given in readable ASCII
- Content is described with MIME types
- Protocol has support for pipelining requests
- Protocol has support for caching

HTTP (2)

HTTP uses persistent connections to improve performance



HTTP (3)

HTTP has several request methods.

Fetch a page → GET

Used to send input data to a server program → POST

Method	Description
GET	Read a Web page
HEAD	Read a Web page's header
POST	Append to a Web page
PUT	Store a Web page
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Connect through a proxy
OPTIONS	Query options for a page

HTTP (4)

Response codes tell the client how the request fared:

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

HTTP (5)

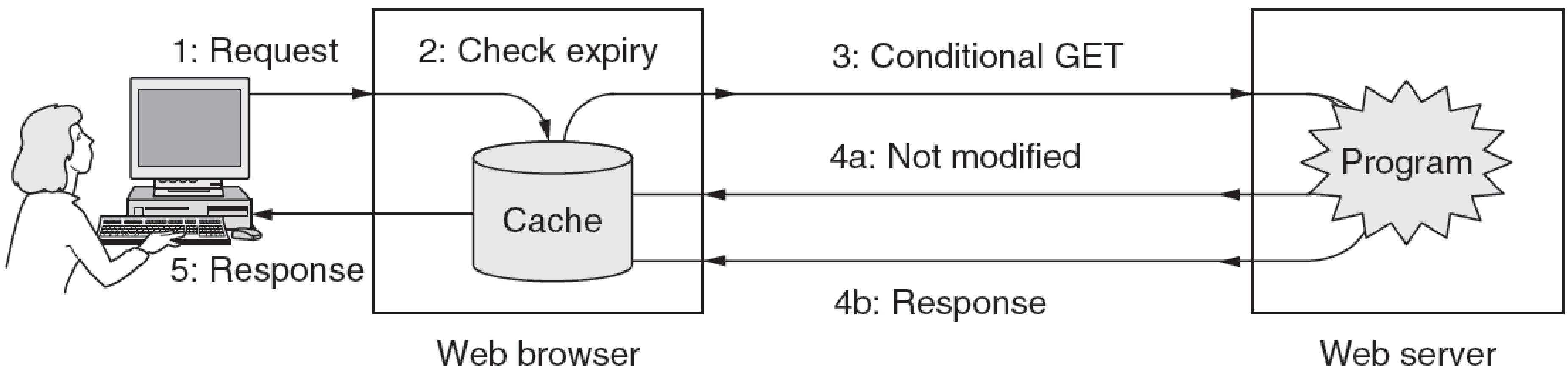
Many headers carry key information:

Function	**Example Headers**
Browser capabilities (client → server)	User-Agent, Accept, Accept-Charset, Accept-Encoding, Accept-Language
Caching related (mixed directions)	If-Modified-Since, If-None-Match, Date, Last-Modified, Expires, Cache-Control, ETag
Browser context (client → server)	Cookie, Referer, Authorization, Host
Content delivery (server → client)	Content-Encoding, Content-Length, Content-Type, Content-Language, Content-Range, Set-Cookie

HTTP (6)

HTTP caching checks to see if the browser has a known fresh copy, and if not if the server has updated the page

- Uses a collection of headers for the checks
- Can include further levels of caching (e.g., proxy)



The Mobile Web

Mobiles (phones, tablets) are challenging as clients:

- Relatively small screens
- Limited input capabilities, lengthy input.
- Network bandwidth is limited
- Connectivity may be intermittent.
- Computing power is limited

Strategies to handle them:

- Content: servers provide mobile-friendly versions; transcoding can also be used
- Protocols: no real need for specialized protocols; HTTP with header compression sufficient

Web Search

Search has proved hugely popular, in tandem with advertising that has proved hugely profitable

- A simple interface for users to navigate the Web

Search engine requires:

- Content from all sites, accessed by crawling. Follow links to new pages, but beware programs.
- Indexing, which benefits from known and discovered structure (such as XML) to increase relevance

Streaming Audio and Video

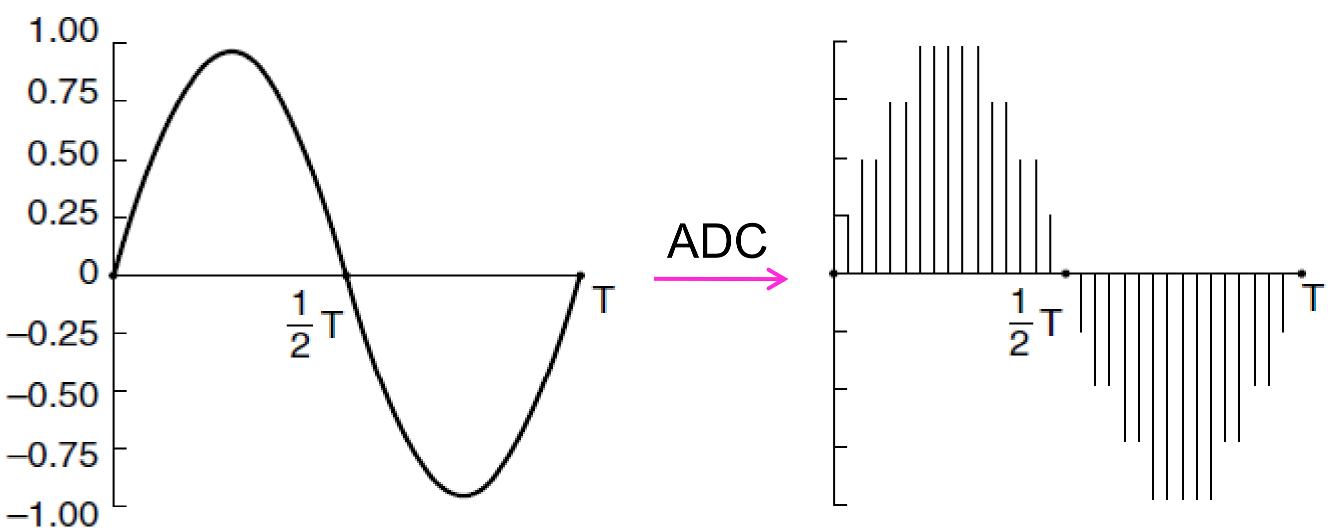
Audio and video have become key types of traffic, e.g., voice over IP, and video streaming.

- Digital audio »
- Digital video »
- Streaming stored media »
- Streaming live media »
- Real-time conferencing »

Digital Audio (1)

ADC (Analog-to-Digital Converter) produces digital audio from a microphone

- Telephone: 8000 8-bit samples/second (64 Kbps); computer audio is usually better quality (e.g., 16 bit)



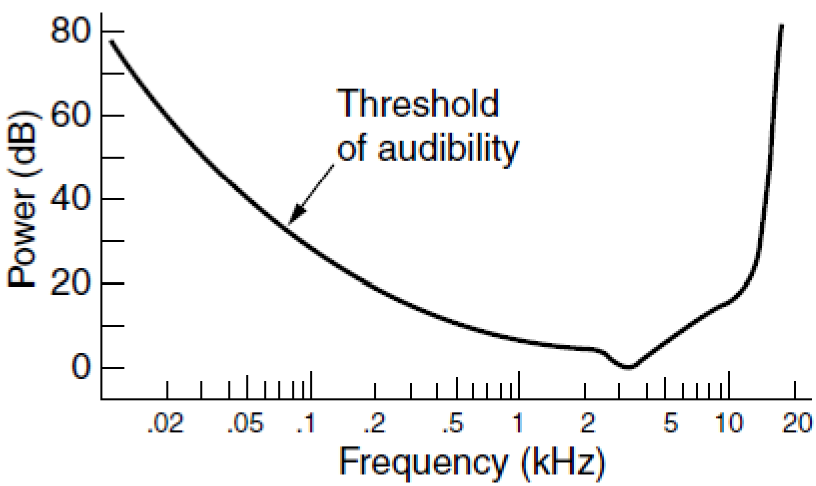
Continuous audio (sine wave)

Digital audio (sampled, 4-bit quantized)

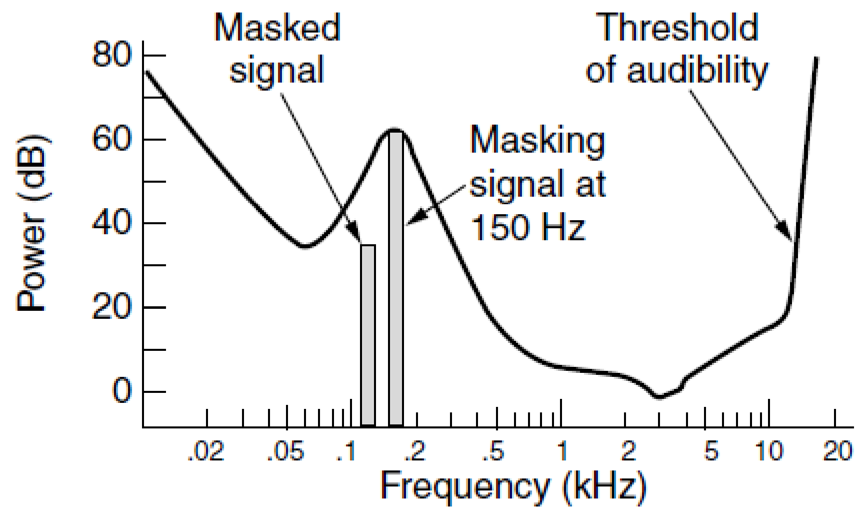
Digital Audio (2)

Digital audio is typically compressed before it is sent

- Lossy encoders (like AAC) exploit human perception
- Large compression ratios (can be >10X)



Sensitivity of the ear varies with frequency



A loud tone can mask nearby tones

Digital Video (1)

Video is digitized as pixels (sampled, quantized)

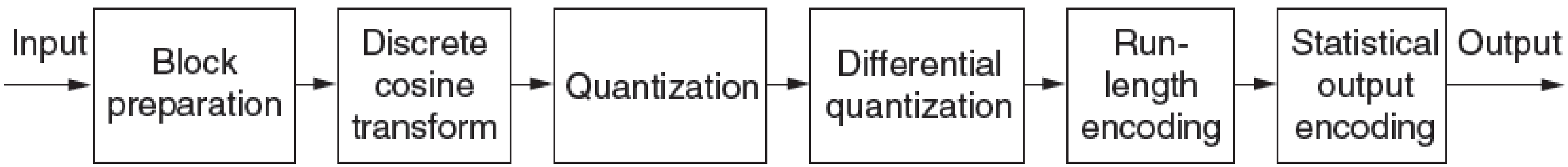
- TV quality: 640x480 pixels, 24-bit color, 30 times/sec

Video is sent compressed due to its large bandwidth

- Lossy compression exploits human perception
  - E.g., JPEG for still images, MPEG, H.264 for video
- Large compression ratios (often 50X for video)
- Video is normally > 1 Mbps, versus >10 kbps for speech and >100 kbps for music

Digital Video (2)

JPEG lossy compression sequence for one image:

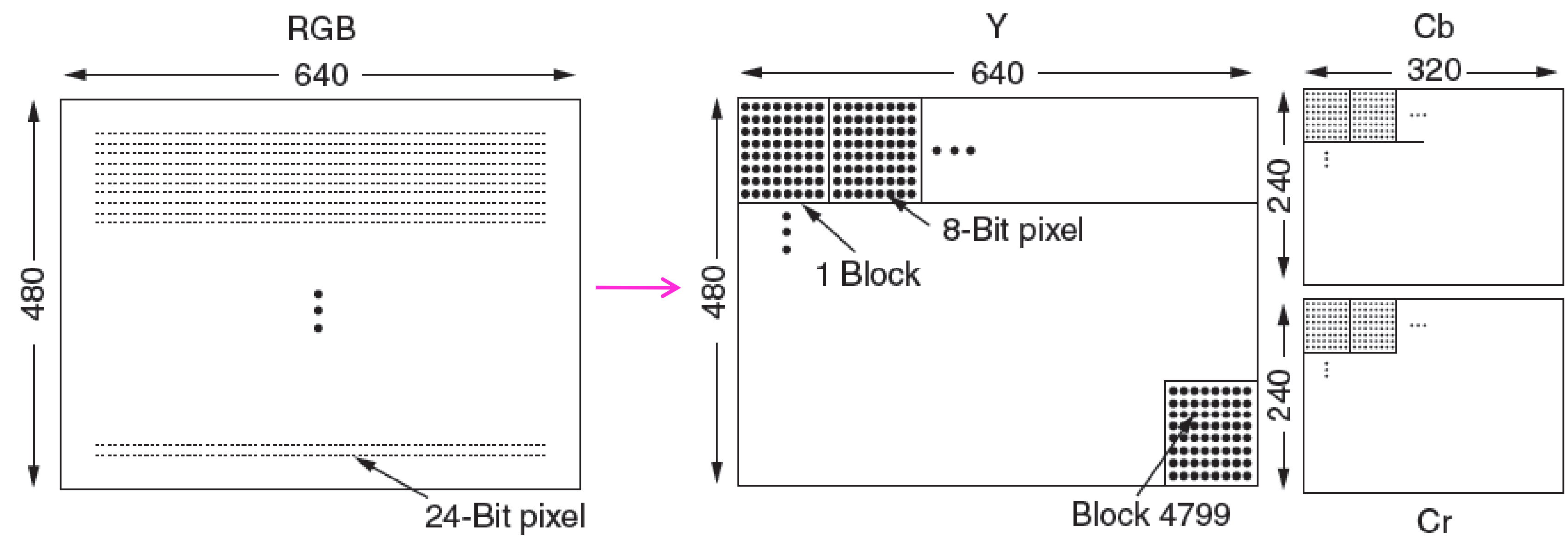


Step 1 Step 2 Step 3 Step 5

Digital Video (3)

Step 1: Pixels are mapped to luminance/chrominance (YCbCr) color space and chrominance is sub-sampled

- The eye is less sensitive to chrominance



Input 24-bit RGB pixels

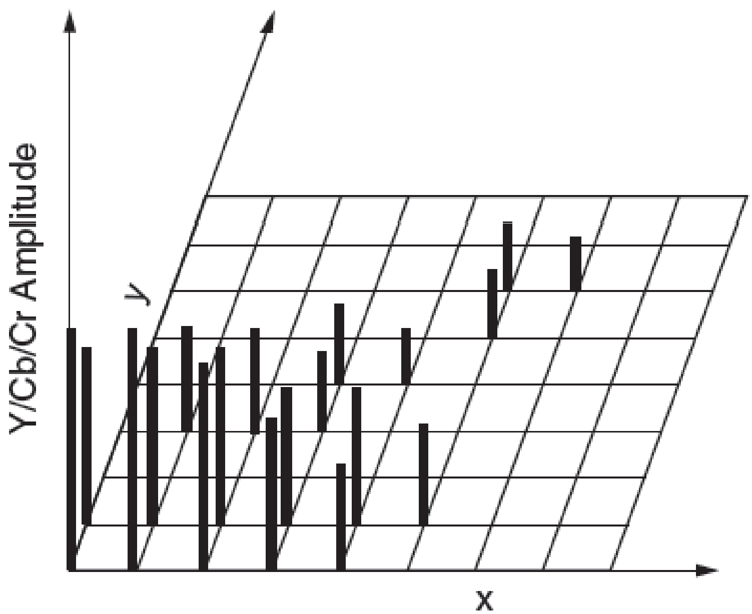
8-bit luminance pixels

8-bit chrominances for every 4 pixels

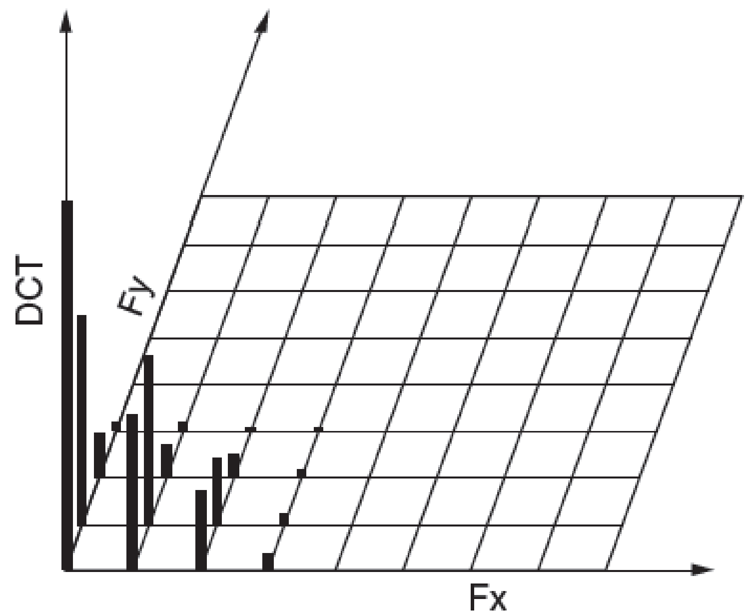
Digital Video (4)

Step 2: Each component block is transformed to spatial frequencies with DCT (Discrete Cosine Transformation)

- Captures the key image features



One component block



Transformed block

Digital Video (5)

Step 3: DCT coefficients are quantized by dividing by thresholds; reduces bits in higher spatial frequencies

- Top left element is differenced over blocks (Step 4)

DCT coefficients

150	80	40	14	4	2	1	0
92	75	36	10	6	1	0	0
52	38	26	8	7	4	0	0
12	8	6	4	2	1	0	0
4	3	2	0	0	0	0	0
2	2	1	1	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Input

/

Quantization table

1	1	2	4	8	16	32	64
1	1	2	4	8	16	32	64
2	2	2	4	8	16	32	64
4	4	4	4	8	16	32	64
8	8	8	8	8	16	32	64
16	16	16	16	16	16	32	64
32	32	32	32	32	32	32	64
64	64	64	64	64	64	64	64

Thresholds

=

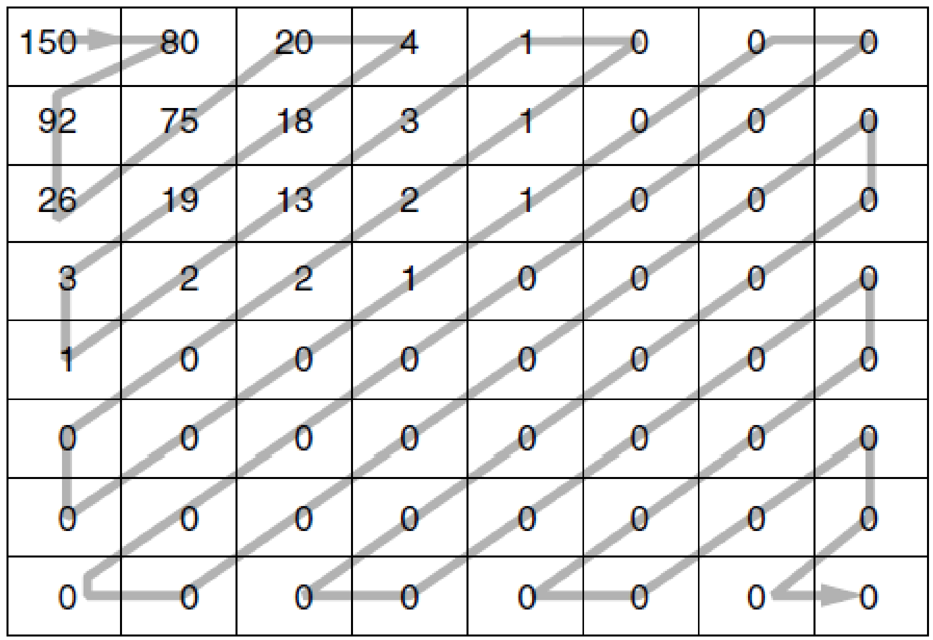
Quantized coefficients

150	80	20	4	1	0	0	0
92	75	18	3	1	0	0	0
26	19	13	2	1	0	0	0
3	2	2	1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Output

Digital Video (6)

Step 5: The block is run-length encoded in a zig-zag order. Then it is Huffman coded before sending (Step 6)

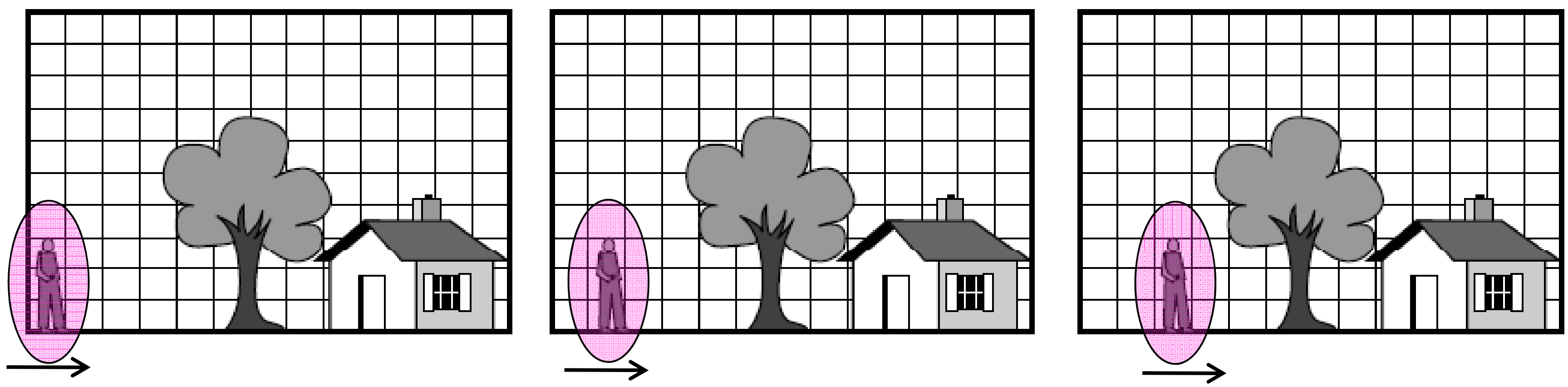


Order in which the block coefficients are sent

Digital Video (7)

MPEG compresses over a sequence of frames, further using motion tracking to remove temporal redundancy

- I (Intra-coded) frames are self-contained
- P (Predictive) frames use block motion predictions
- B (Bidirectional) frames may base prediction on future frame

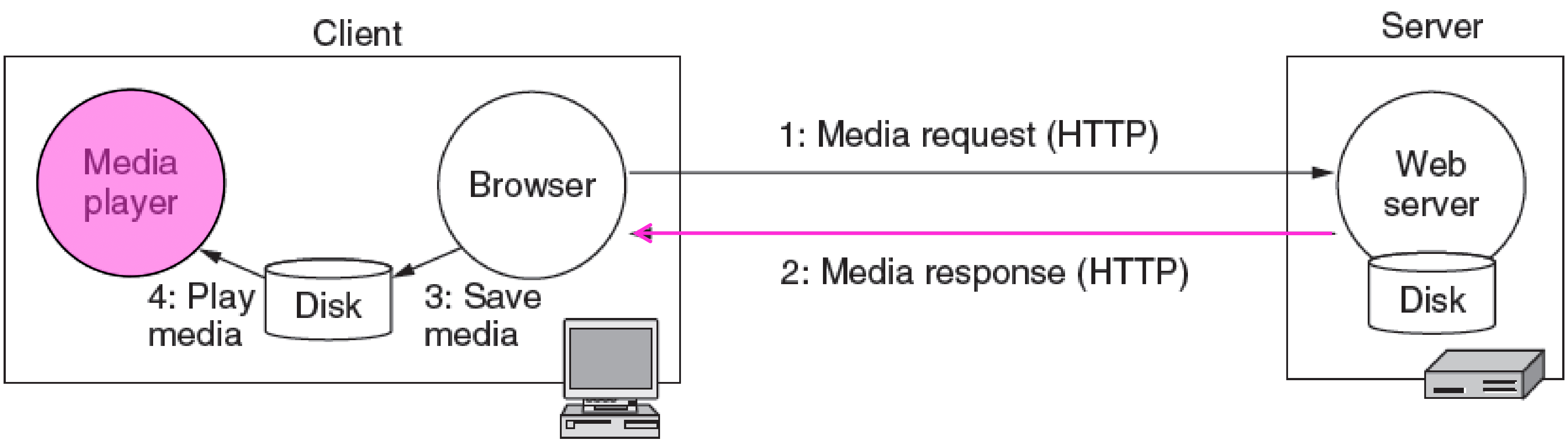


Three consecutive frames with stationary and moving components

Streaming Stored Media (1)

A simple method to stream stored media, e.g., for video on demand, is to fetch the video as a file download

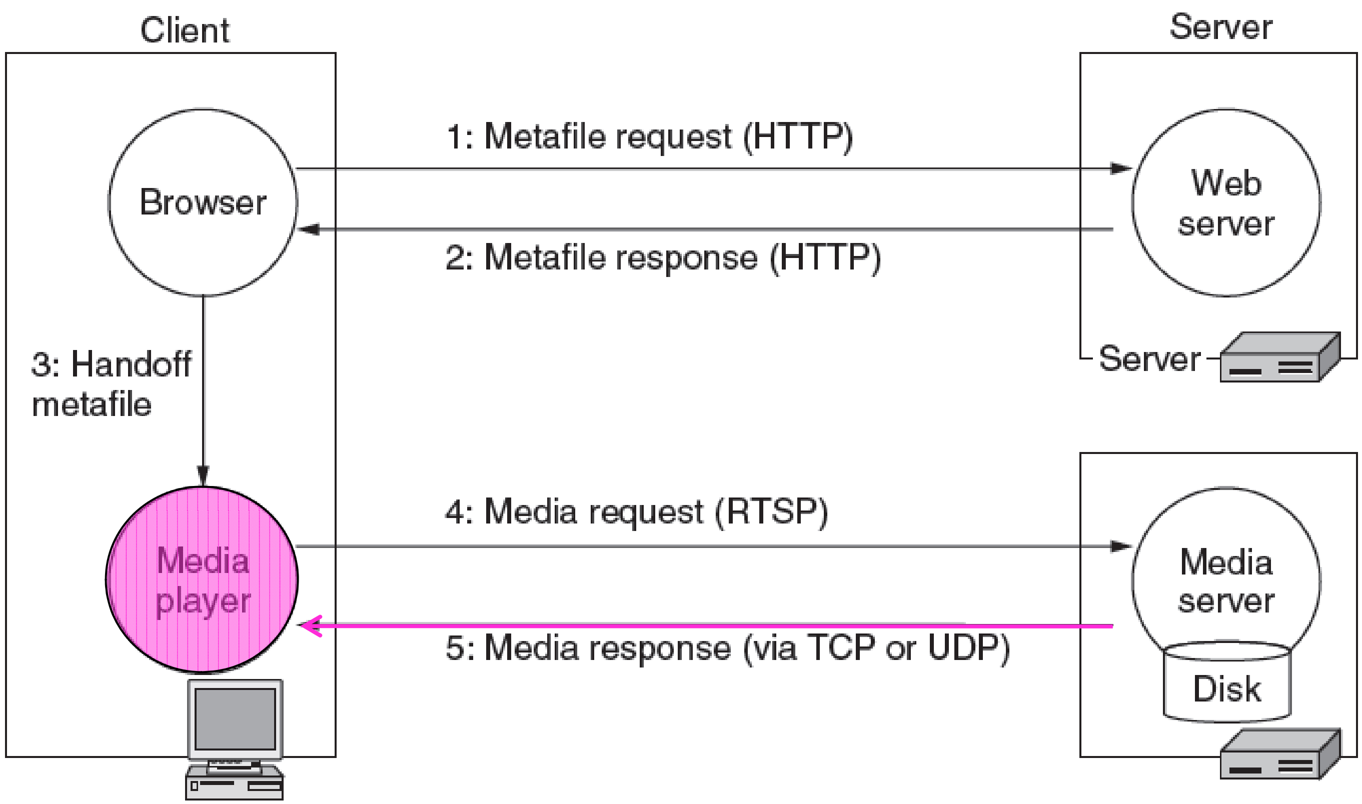
- But has large startup delay, except for short files



Streaming Stored Media (2)

Effective streaming starts the playout during transport

- With RTSP (Real-Time Streaming Protocol)



Streaming Stored Media (3)

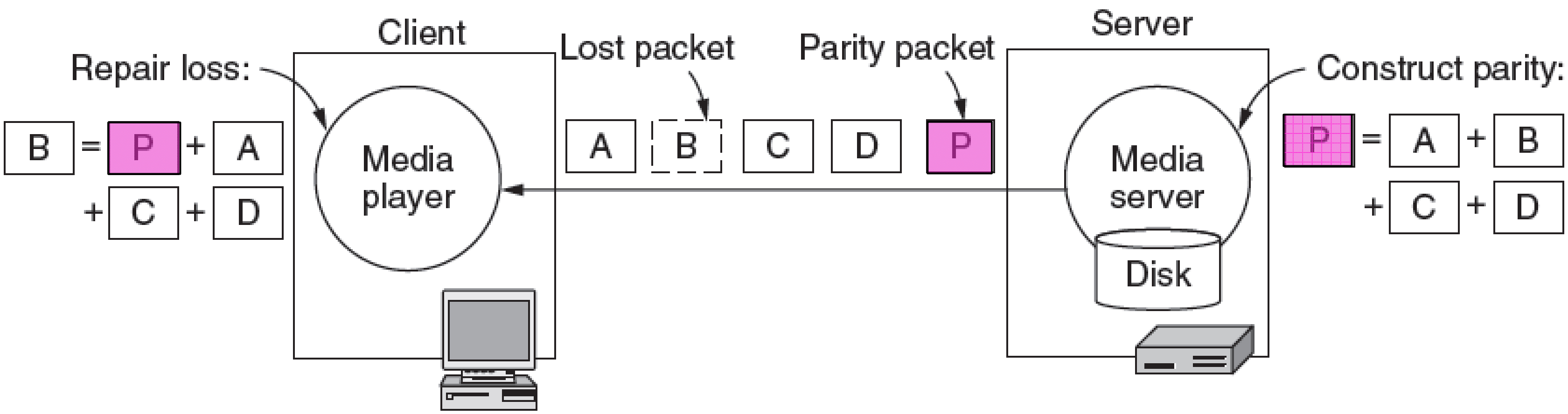
Key problem: how to handle transmission errors

Strategy	**Advantage**	**Disadvantage**
Use reliable transport (TCP)	Repairs all errors	Increases jitter significantly
Add FEC (e.g., parity)	Repairs most errors	Increases overhead, decoding complexity and jitter
Interleave media	Masks most errors	Slightly increases decoding complexity and jitter

Streaming Stored Media (4)

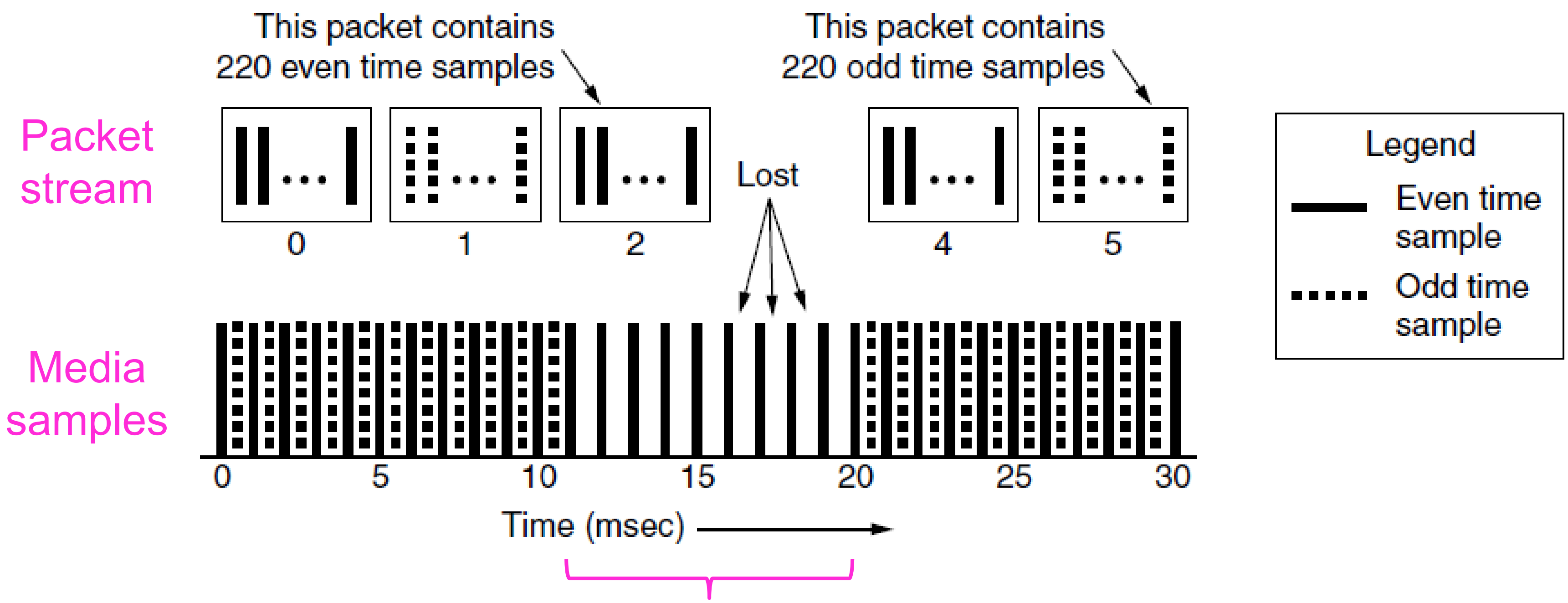
Parity packet can repair one lost packet in a group of N

- Decoding is delayed for N packets



Streaming Stored Media (5)

Interleaving spreads nearby media samples over different transmissions to reduce the impact of loss

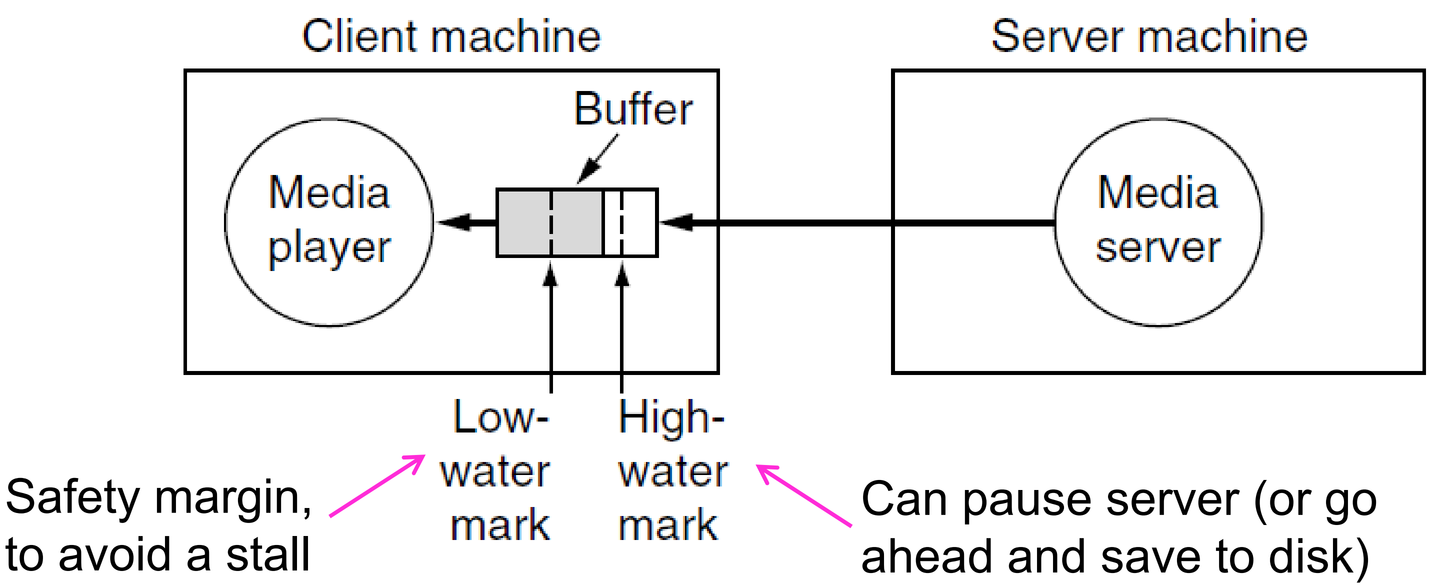


Loss reduces temporal resolution; doesn't leave a gap

Streaming Stored Media (6)

Key problem: media may not arrive in time for playout due to variable bandwidth and loss/retransmissions

- Client buffers media to absorb jitter; we still need to pick an achievable media rate



Streaming Stored Media (7)

RTSP commands

- Sent from player to server to adjust streaming

Command	**Server action**
DESCRIBE	List media parameters
SETUP	Establish a logical channel between the player and the server
PLAY	Start sending data to the client
RECORD	Start accepting data from the client
PAUSE	Temporarily stop sending data
TEARDOWN	Release the logical channel

Streaming Live Media (1)

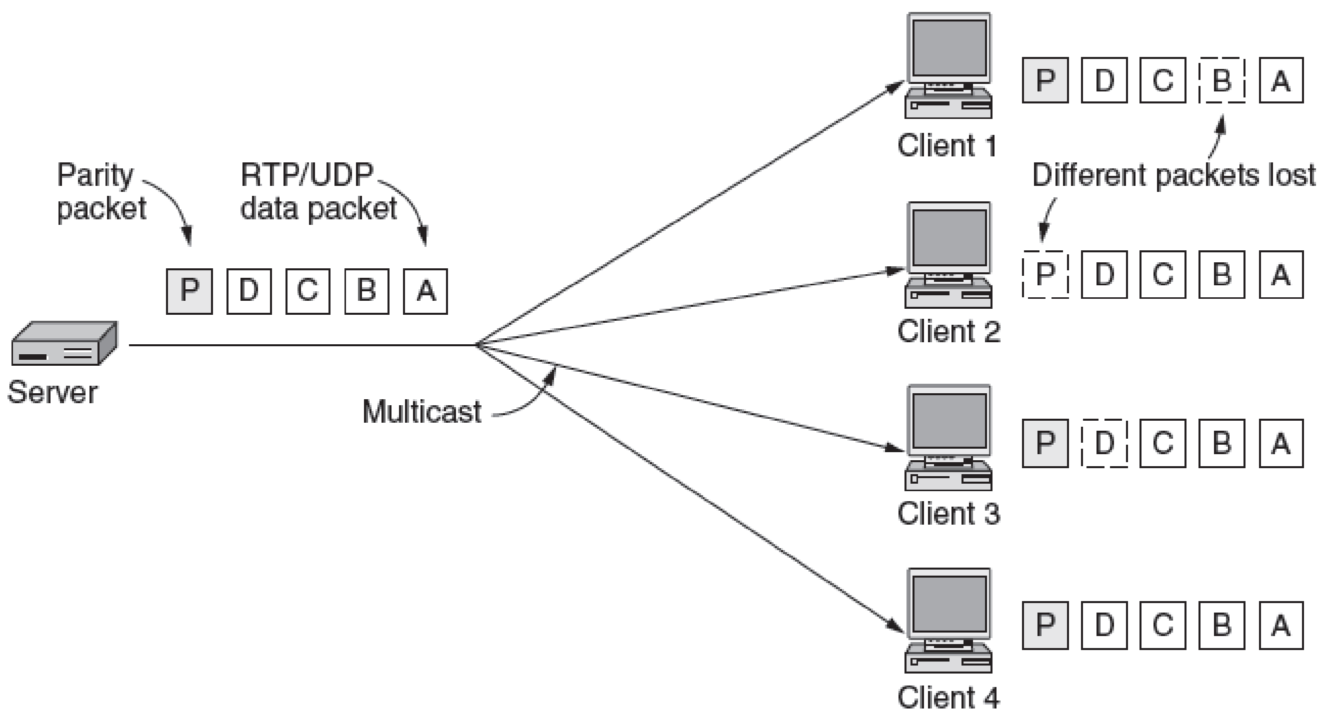
Streaming live media is similar to the stored case plus:

- Can’t stream faster than “live rate” to get ahead
  - Usually need larger buffer to absorb jitter
- Often have many users viewing at the same time
  - UDP with multicast greatly improves efficiency. It is rarely available, so many TCP connections are used.
  - For very many users, content distribution is used [later]

Streaming Live Media (2)

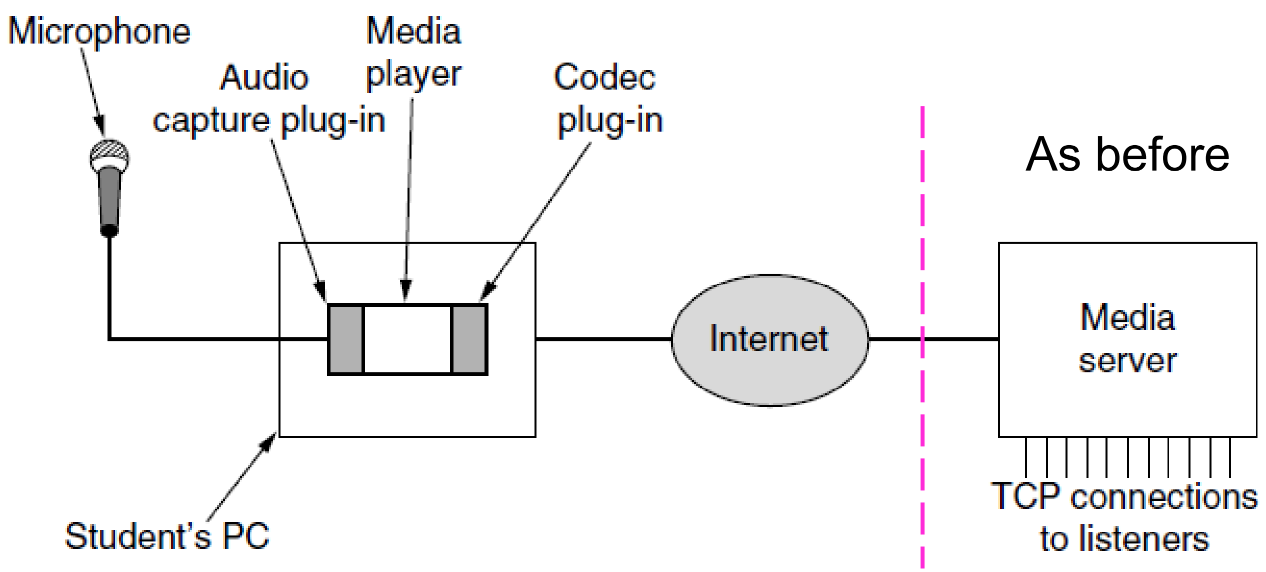
With multicast streaming media, parity is effective

- Clients can each lose a different packet and recover



Streaming Live Media (2)

Production side of a student radio station.



Real-Time Conferencing (1)

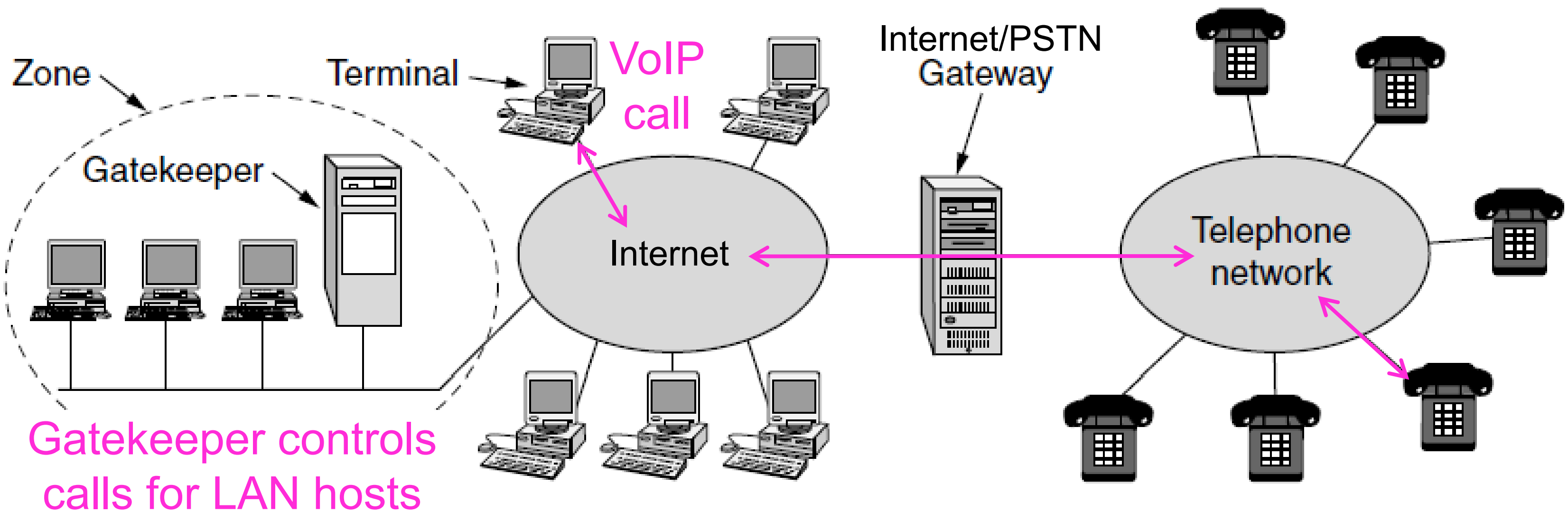
Real-time conferencing has two or more connected live media streams, e.g., voice over IP, Skype video call

Key issue over live streaming is low (interactive) latency

- Want to reduce delay to near propagation
- Benefits from network support, e.g., QoS
- Or, benefits from ample bandwidth (no congestion)

Real-Time Conferencing (2)

H.323 architecture for Internet telephony supports calls between Internet computers and PSTN phones.



Real-Time Conferencing (3)

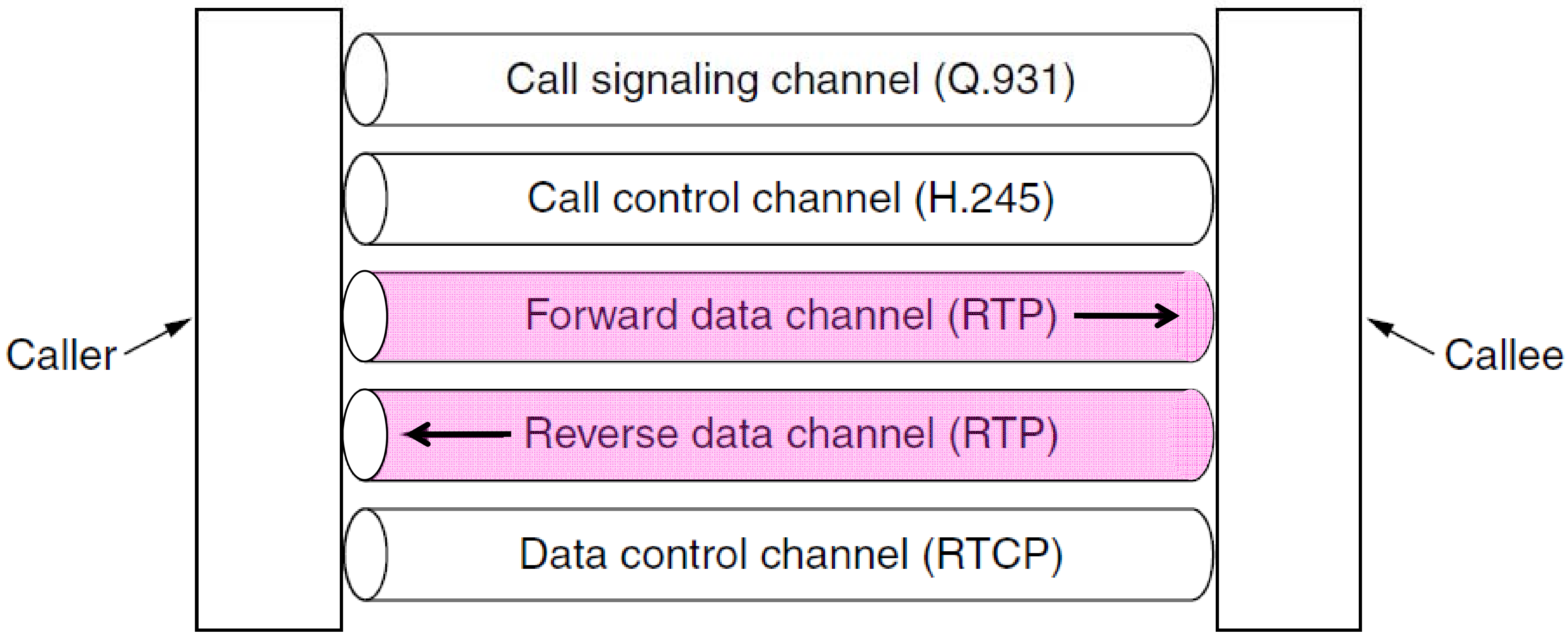
H.323 protocol stack:

- Call is digital audio/video over RTP/UDP/IP
- Call setup is handled by other protocols (Q.931 etc.)

<table>
<tr><td>Audio</td><td>Video</td><td colspan="4">Control</td></tr>
<tr><td>G.7xx</td><td>H.26x</td><td rowspan="2">RTCP</td><td rowspan="2">H.225
(RAS)</td><td rowspan="2">Q.931
(Signaling)</td><td rowspan="2">H.245
(Call
Control)</td></tr>
<tr><td colspan="2">RTP</td></tr>
<tr><td colspan="4">UDP</td><td colspan="2">TCP</td></tr>
<tr><td colspan="6">IP</td></tr>
<tr><td colspan="6">Link layer protocol</td></tr>
<tr><td colspan="6">Physical layer protocol</td></tr>
</table>

Real-Time Conferencing (4)

Logical channels that make up an H.323 call



Real-Time Conferencing (5)

SIP (Session Initiation Protocol) is an alternative to H.323 to set up real-time calls

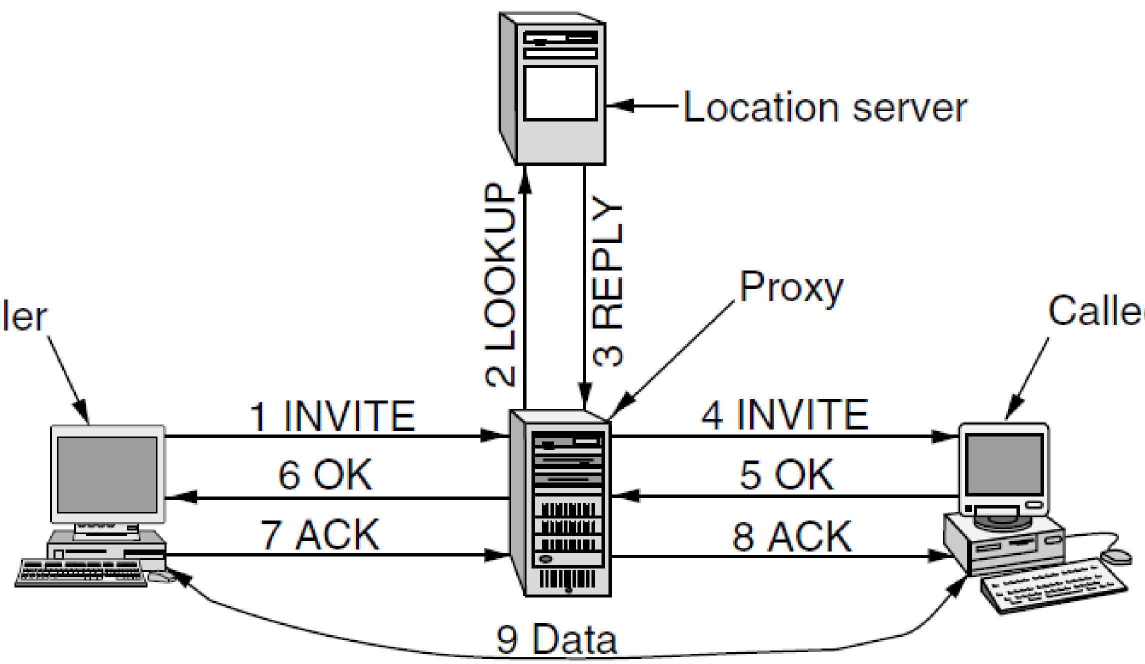
- Simple, text-based protocol with URLs for addresses
- Data is carried with RTP / RTCP as before

Method	Description
INVITE	Request initiation of a session
ACK	Confirm that a session has been initiated
BYE	Request termination of a session
OPTIONS	Query a host about its capabilities
CANCEL	Cancel a pending request
REGISTER	Inform a redirection server about the user's current location

Real-Time Conferencing (6)

Setting up a call with the SIP three-way handshake

- Proxy server lets a remote callee be connected
- Call data flows directly between caller/callee



Real-Time Conferencing (7)

Item	H.323	SIP
Designed by	ITU	IETF
Compatibility with PSTN	Yes	Largely
Compatibility with Internet	Yes, over time	Yes
Architecture	Monolithic	Modular
Completeness	Full protocol stack	SIP just handles setup
Parameter negotiation	Yes	Yes
Call signaling	Q.931 over TCP	SIP over TCP or UDP
Message format	Binary	ASCII
Media transport	RTP/RTCP	RTP/RTCP
Multiparty calls	Yes	Yes
Multimedia conferences	Yes	No
Addressing	URL or phone number	URL
Call termination	Explicit or TCP release	Explicit or timeout
Instant messaging	No	Yes
Encryption	Yes	Yes
Size of standards	1400 pages	250 pages
Implementation	Large and complex	Moderate, but issues
Status	Widespread, esp. video	Alternative, esp. voice

Comparison of H.323 and SIP.

Content Delivery

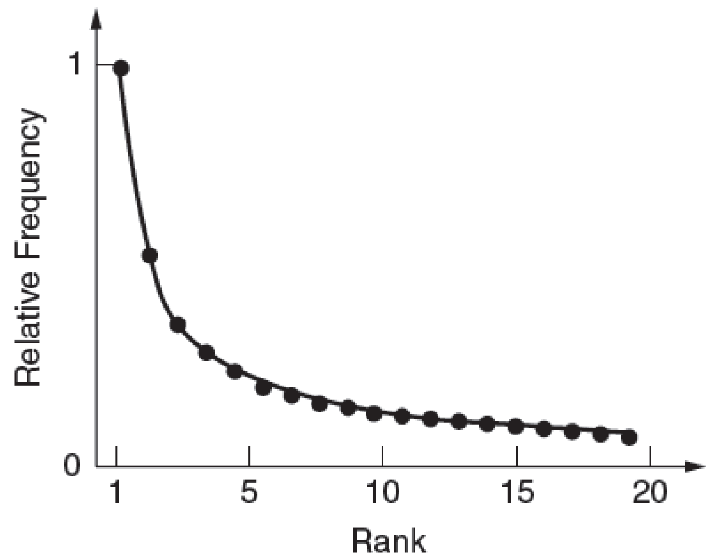
Delivery of content, especially Web and video, to users is a major component of Internet traffic

- Content and Internet traffic »
- Server farms and Web proxies »
- Content delivery networks »
- Peer-to-peer networks »

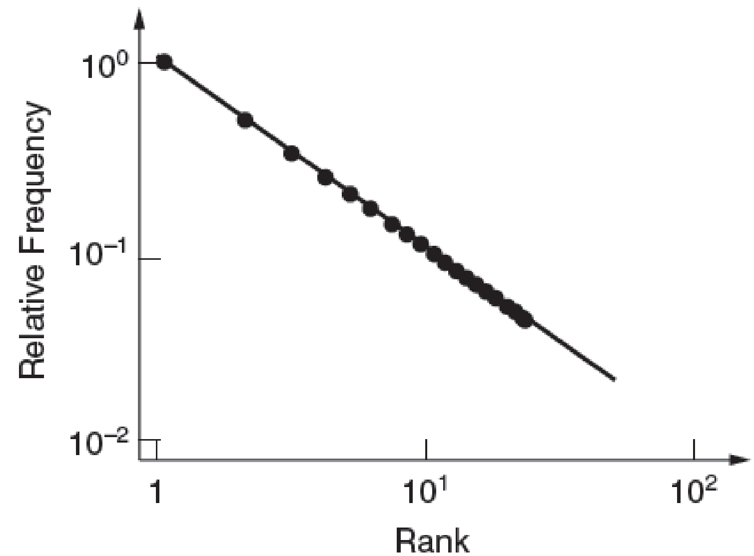
Content and Internet Traffic

Internet traffic:

1. Shifts seismically (email→FTP→Web→P2P→video)
2. Has many small/unpopular and few large/popular flows – mice and elephants



Zipf popularity distribution, 1/k

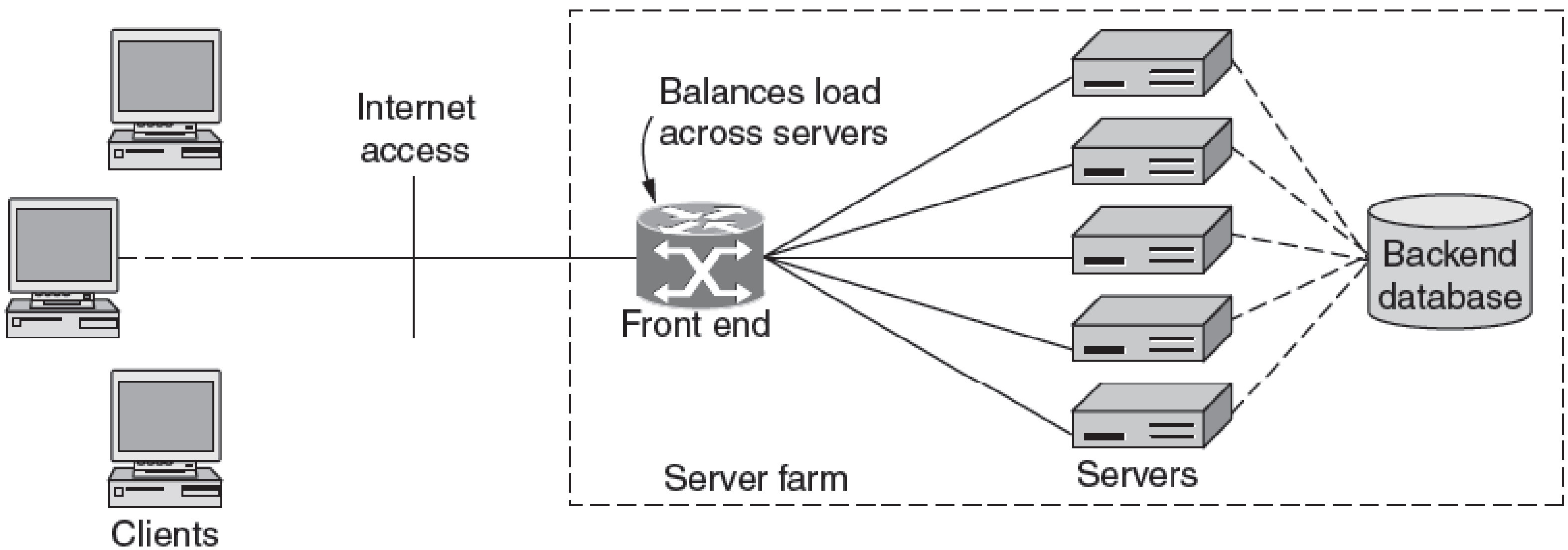


Shows up as a line on log-log plot

Server Farms and Web Proxies (1)

Server farms enable large-scale Web servers:

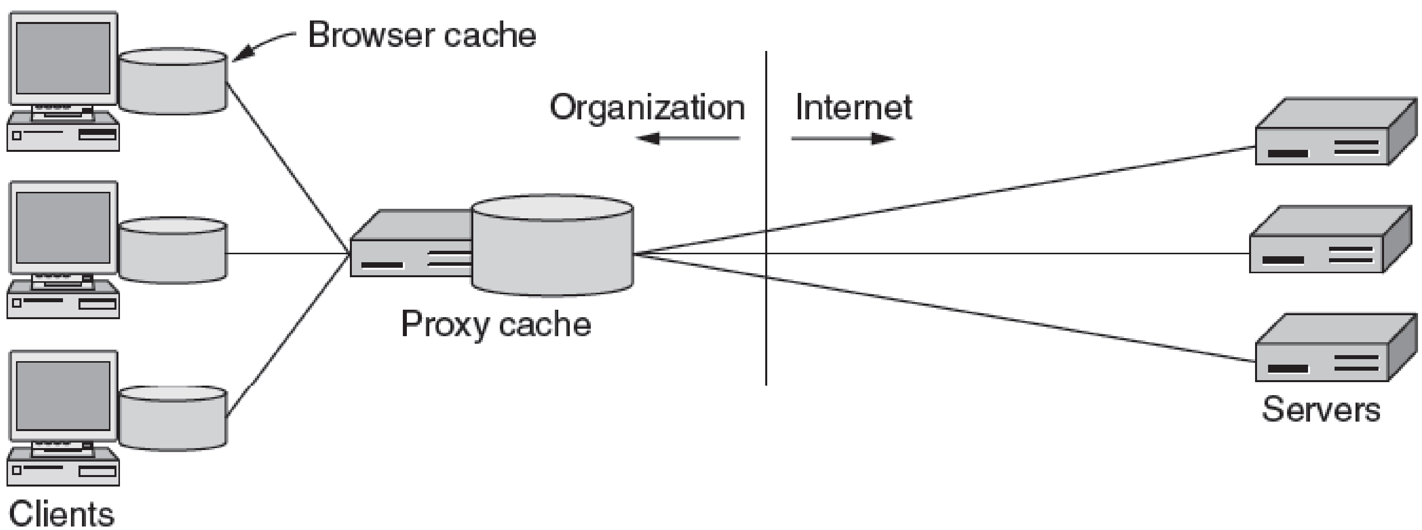
- Front-end load-balances requests over servers
- Servers access the same backend database



Server Farms and Web Proxies (2)

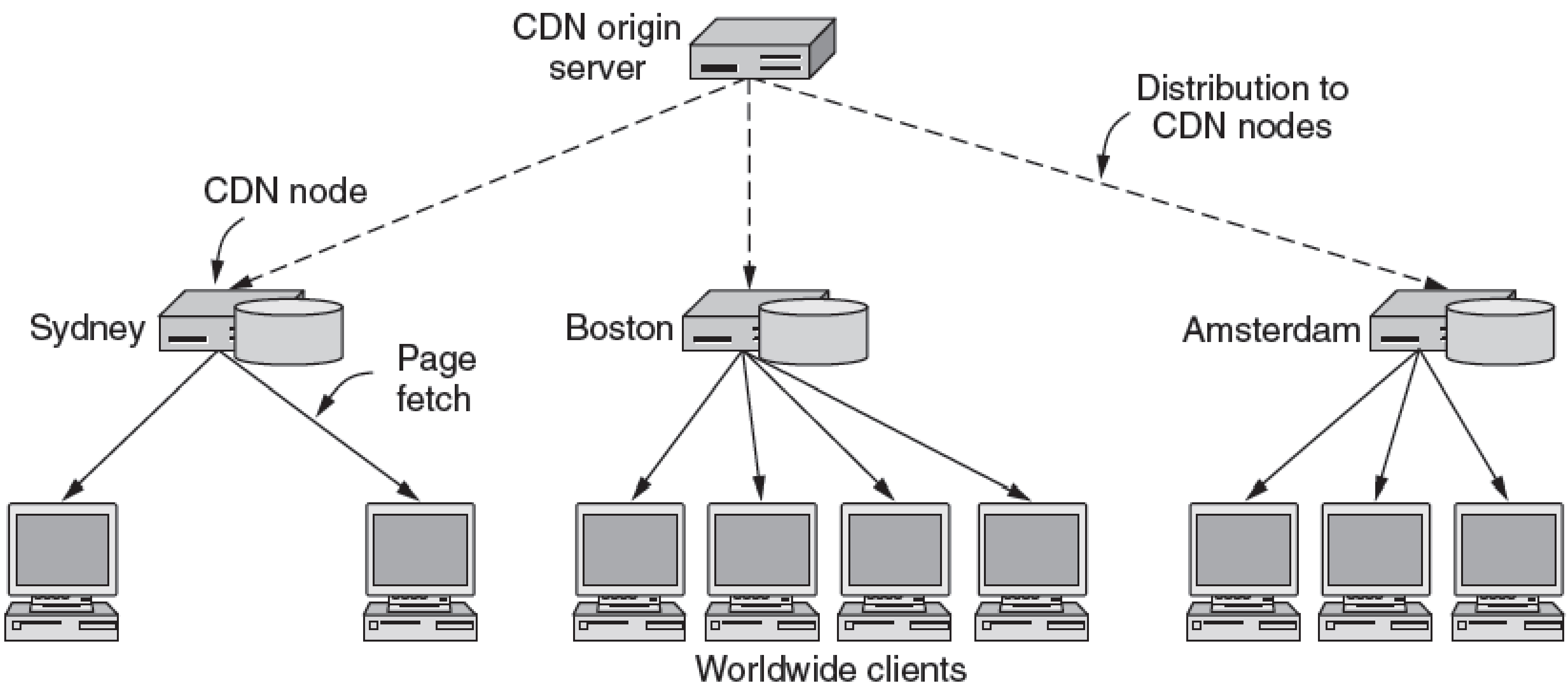
Proxy caches help organizations to scale the Web

- Caches server content over clients for performance
- Also implements organization policies (e.g., access)



CDNs – Content Delivery Networks (1)

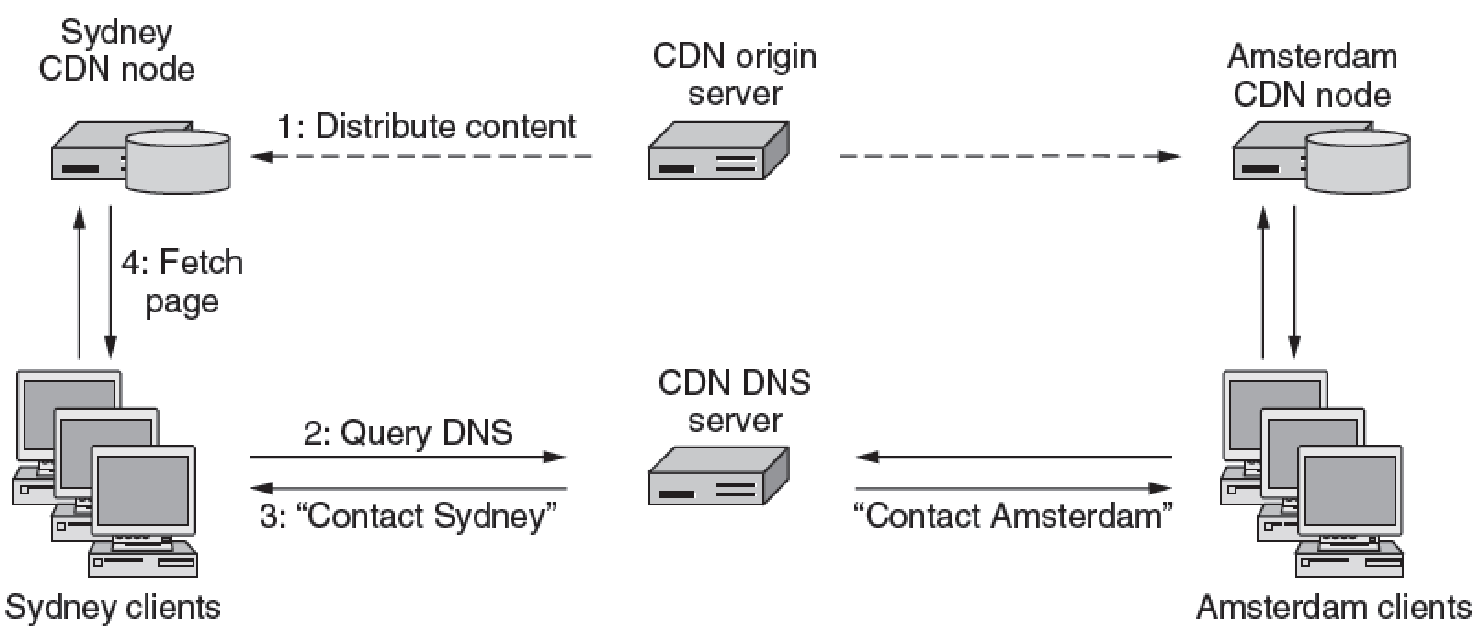
CDNs scale Web servers by having clients get content from a nearby CDN node (cache)



Content Delivery Networks (2)

Directing clients to nearby CDN nodes with DNS:

- Client query returns local CDN node as response
- Local CDN node caches content for nearby clients and reduces load on the origin server



Content Delivery Networks (3)

Origin server rewrites pages to serve content via CDN

```
<html>
<head> <title> Fluffy Video </title> </head>
<body>
<h1> Fluffy Video's Product List </h1>
<p> Click below for free samples. </p>

<a href="koalas.mpg"> Koalas Today </a> <br>
<a href="kangaroos.mpg"> Funny Kangaroos </a> <br>
<a href="wombats.mpg"> Nice Wombats </a> <br>
</body>
</html>
```

Traditional Web page on server

```
<html>
<head> <title> Fluffy Video </title> </head>
<body>
<h1> Fluffy Video's Product List </h1>
<p> Click below for free samples. </p>

<a href="http://www.cdn.com/fluffyvideo/koalas.mpg"> Koalas Today </a> <br>
<a href="http://www.cdn.com/fluffyvideo/kangaroos.mpg"> Funny Kangaroos </a> <br>
<a href="http://www.cdn.com/fluffyvideo/wombats.mpg"> Nice Wombats </a> <br>
</body>
</html>
```

Page that distributes content via CDN

Peer-to-Peer Networks (1)

P2P (Peer-to-Peer) is an alternative CDN architecture with no dedicated infrastructure (i.e., servers)

- Clients serve content to each other as peers

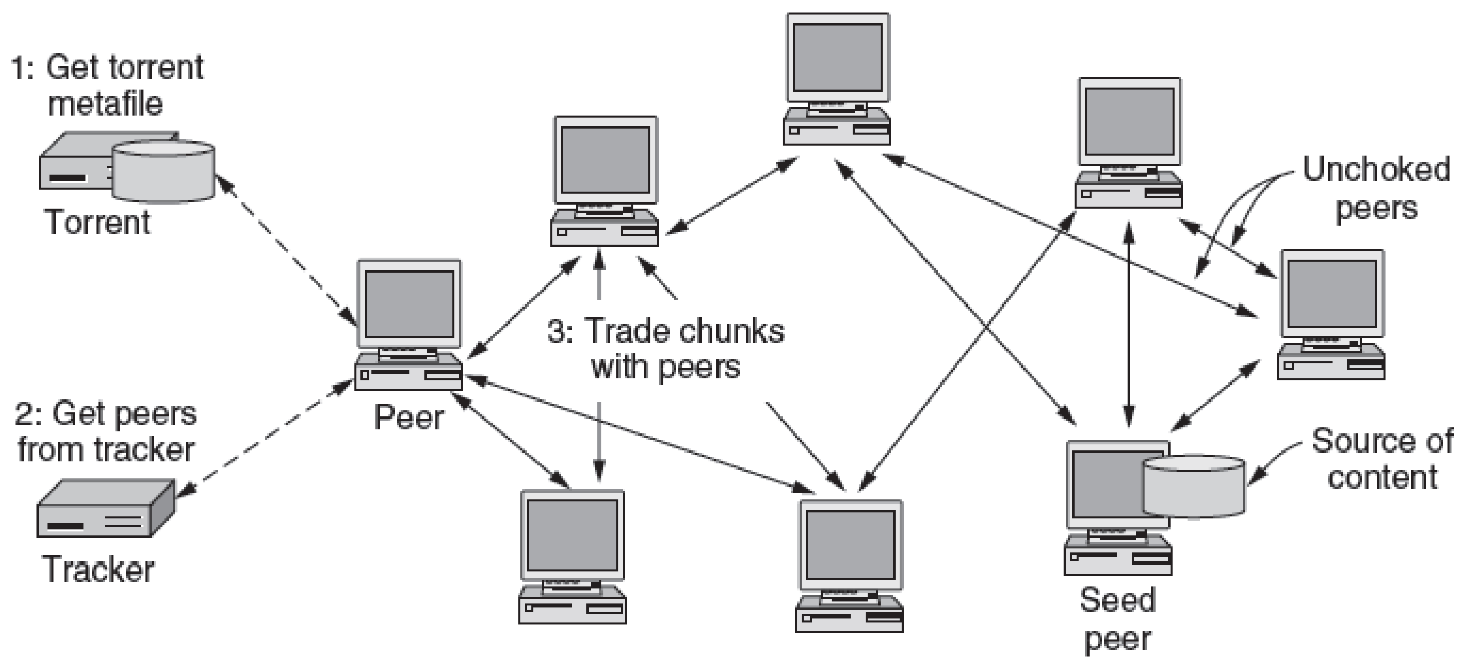
Challenges when servers are removed:

1. How do peers find each other?
2. How do peers support rapid content downloads?
3. How do peers encourage each other to upload?

Peer-to-Peer Networks (2)

BitTorrent lets peers download torrents

- Peers find each other via Tracker in torrent file
- Peers swap chunks (parts of content) with partners, preferring those who send most quickly [2]
- Many peers speed download; preference helps uploads [3]



Peer-to-Peer Networks (3)

Distributed Hash Tables (DHTs) are a fully distributed index that scales to very many clients/entries

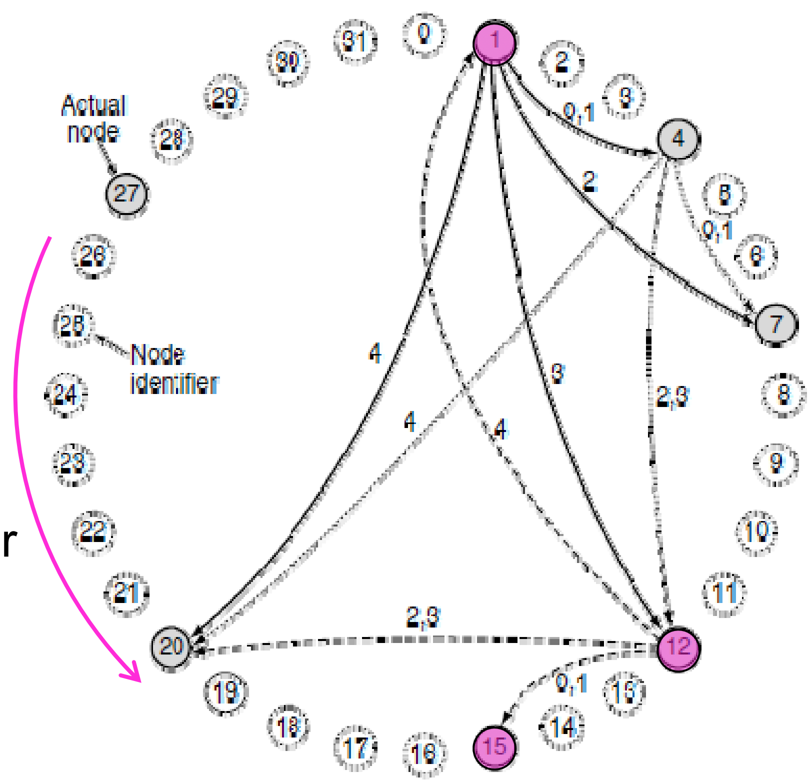
- Need to follow O(log N) path for N entries
- Can use as Tracker to find peers with no servers [1]
- Look up torrent (identifier) in DHT to find IP of peers
- Kademlia is used in BitTorrent

Peer-to-Peer Networks (3)

A Chord ring of 32 identifiers. Finger tables [at right, and as arcs] are used to navigate the ring.

- Example: path to look up 16 from 1 is 1 → 12 → 15

Identifier values are stored at predecessor



Start	IP addr of successor
2	4
3	4
5	7
9	12
17	20

Node 1's finger table

Start	IP addr of successor
5	7
6	7
8	12
12	12
20	20

Node 4's finger table

Start	IP addr of successor
13	15
14	15
16	20
20	20
28	1

Node 12's finger table

End

Chapter 7