THE APPLICATION LAYER - HTTP

COMP 30650: NETWORKS AND INTERNET SYSTEMS

Dr. Gavin McArdle
Email: gavin.mcardle@ucd.ie
Office: A1.09 Computer Science

RECAP

- **Application layer is at top of the stack**
  - Uses layer below it
  - Passes requests to layer below for end-to-end delivery
  - e.g. UDP and TCP
- **Domain Name Services**
  - Resolves domain name to IP Address
  - Name Servers

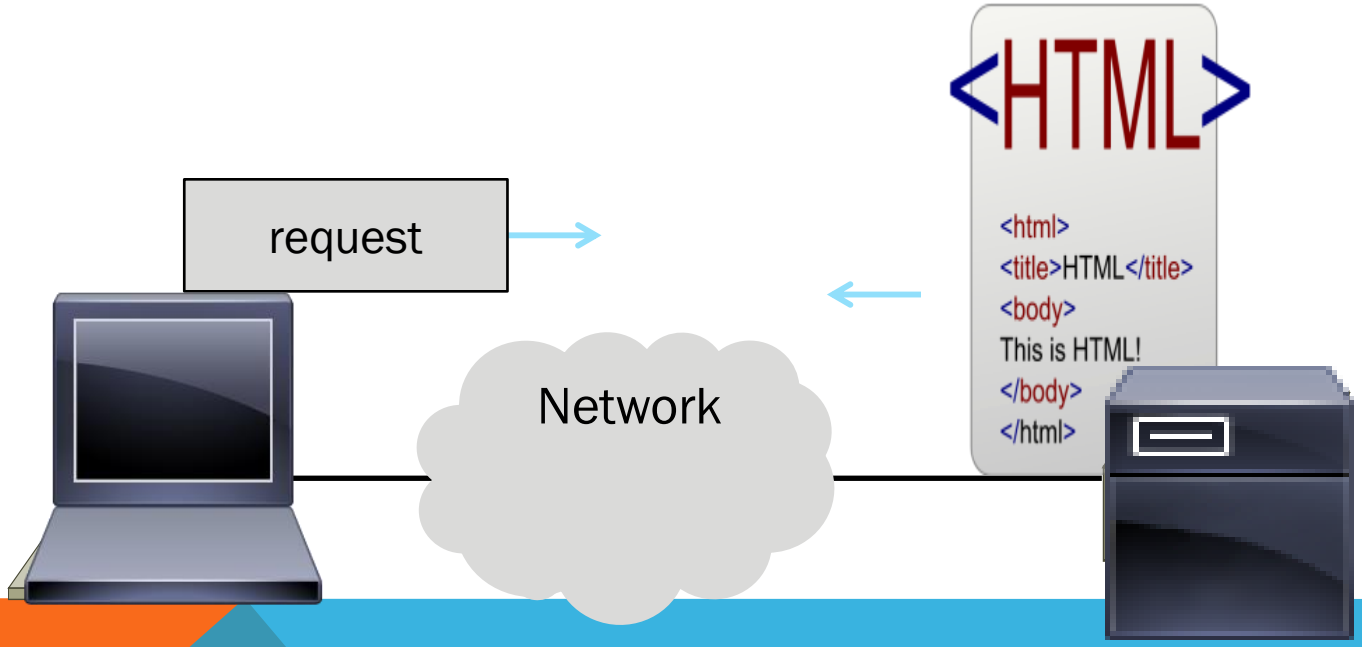
TODAY'S PLAN

- **Hyper Text Transfer Protocol**
  - For Fetching Web pages
- **Improving Performance**
  - Persistent Connections
  - Caching
  - Content Delivery Networks

HTTP

HTTP, (HyperText Transfer Protocol)

- Basis for fetching Web pages



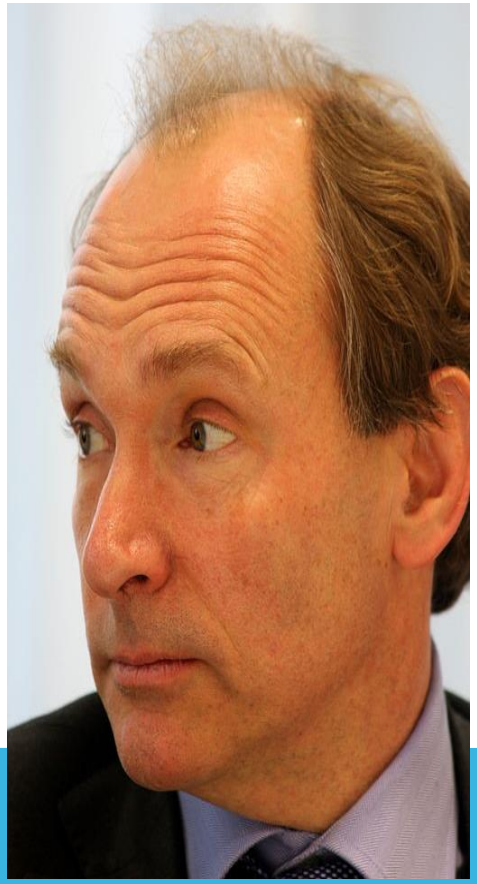
SIR TIM BERNERS-LEE (1955–)

Inventor of the Web

- Dominant Internet app since mid 90s
- He now directs the W3C

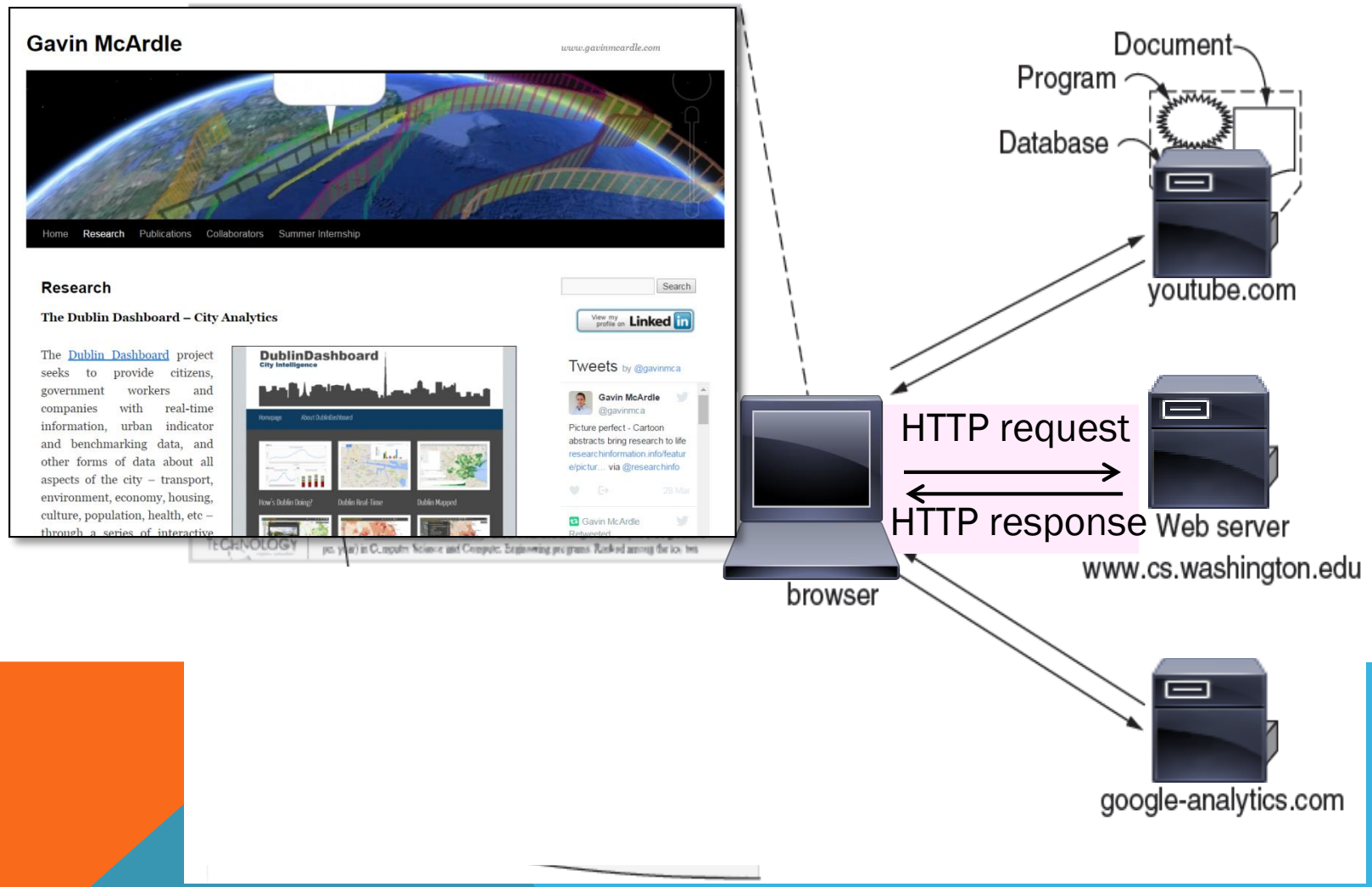
Developed Web at CERN in ‘89

- Browser, server and first HTTP
- Popularized via Mosaic (‘93), Netscape
- First WWW conference in ’94 …



Source: By Paul Clarke, CC-BY-2.0, via Wikimedia Commons

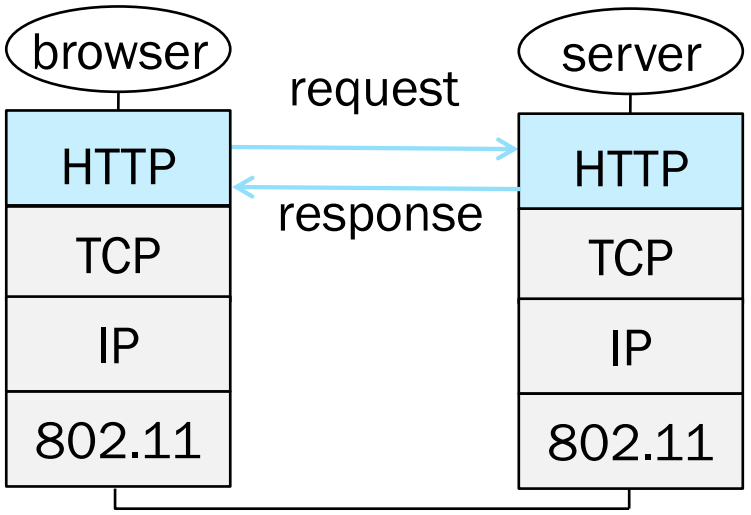
WEB CONTEXT


Gavin McArdle
www.gavinmcardle.com
Home
Research
Publications
Collaborators
Summer Internship
Research
The Dublin Dashboard – City Analytics
The Dublin Dashboard project seeks to provide citizens, government workers and companies with real-time information, urban indicator and benchmarking data, and other forms of data about all aspects of the city – transport, environment, economy, housing, culture, population, health, etc – through a series of interactive
DublinDashboard
City Intelligence
Search
Tweets by @gavinmca
Gavin McArdle
@gavinmca
Picture perfect - Cartoon abstracts bring research to life researchinformation.info/feature/pictur... via @researchinfo
Document
Program
Database
youtube.com
HTTP request
HTTP response
Web server
www.cs.washington.edu
browser
google-analytics.com

WEB PROTOCOL CONTEXT

HTTP is a request/response protocol for fetching Web resources

- Runs on TCP, typically port 80
- Part of browser/server app



FETCHING A WEB PAGE WITH HTTP

Start with the page URL:

http://en.wikipedia.org/wiki/Vegemite

Steps:

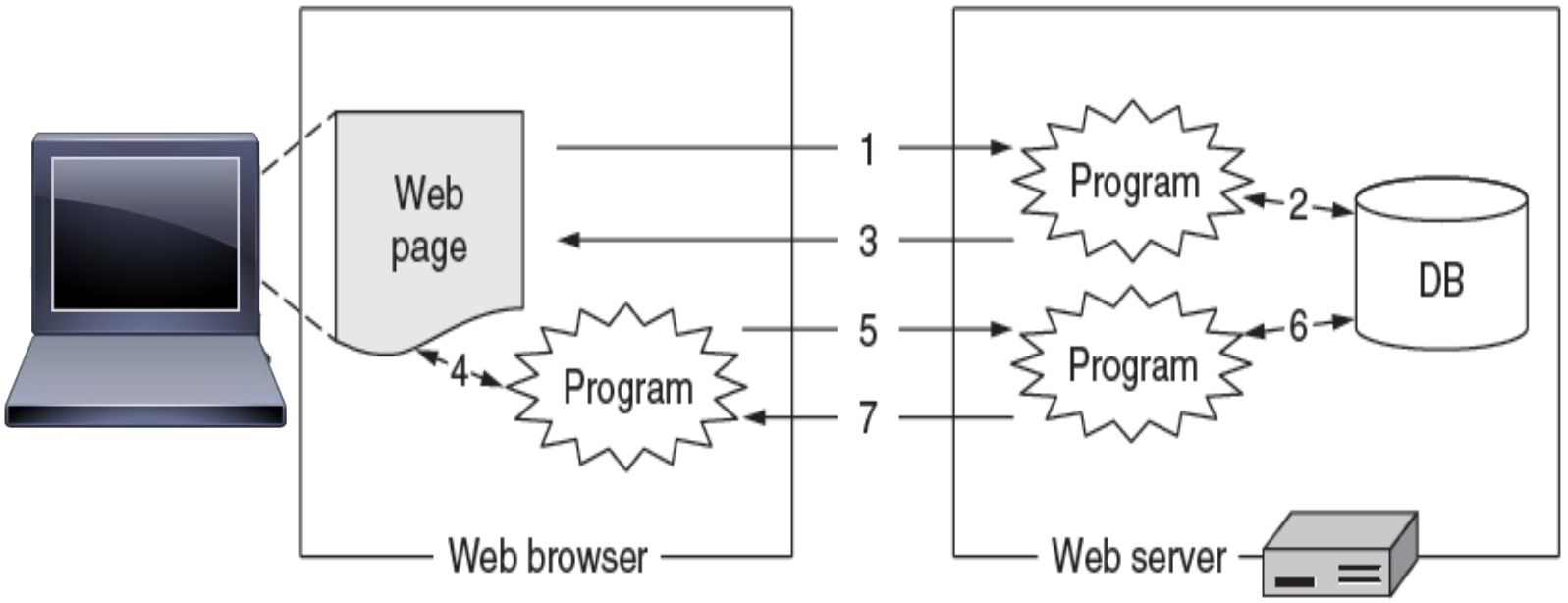
- Resolve the server to IP address (DNS)
- Set up TCP connection to the server
- Send HTTP request for the page
- (Await HTTP response for the page)
- Execute / fetch embedded resources / render
  Clean up any idle TCP connections

STATIC VS DYNAMIC WEB PAGES

Static web page is a file contents, e.g., image

Dynamic web page is the result of program execution

- Javascript on client, PHP on server, or both



HTTP PROTOCOL

Commands used in the request

Fetch page →

Upload data →

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 PROTOCOL

Codes returned with the response

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 PROTOCOL

Many header fields specify capabilities and content

- E.g., Content-Type: text/html, Cookie: lect=8-4-http

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 PERFORMANCE PLT (PAGE LOAD TIME)

PLT is the key measure of web performance

- From click until user sees page
- Small increases in PLT decrease sales

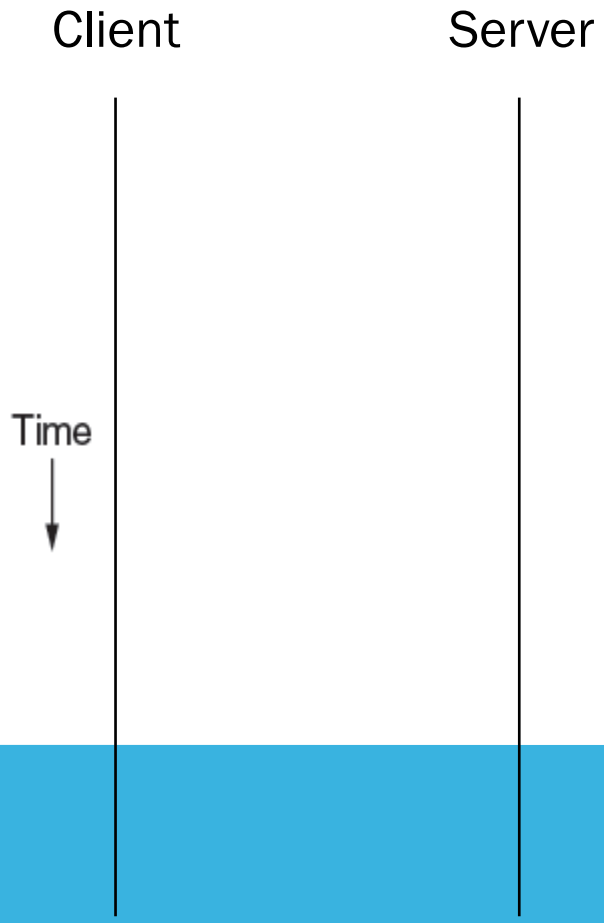
PLT depends on many factors

- Structure of page/content
- HTTP (and TCP!) protocol
- Network RTT and bandwidth

EARLY PERFORMANCE

HTTP/1.0 uses one TCP connection to fetch one web resource

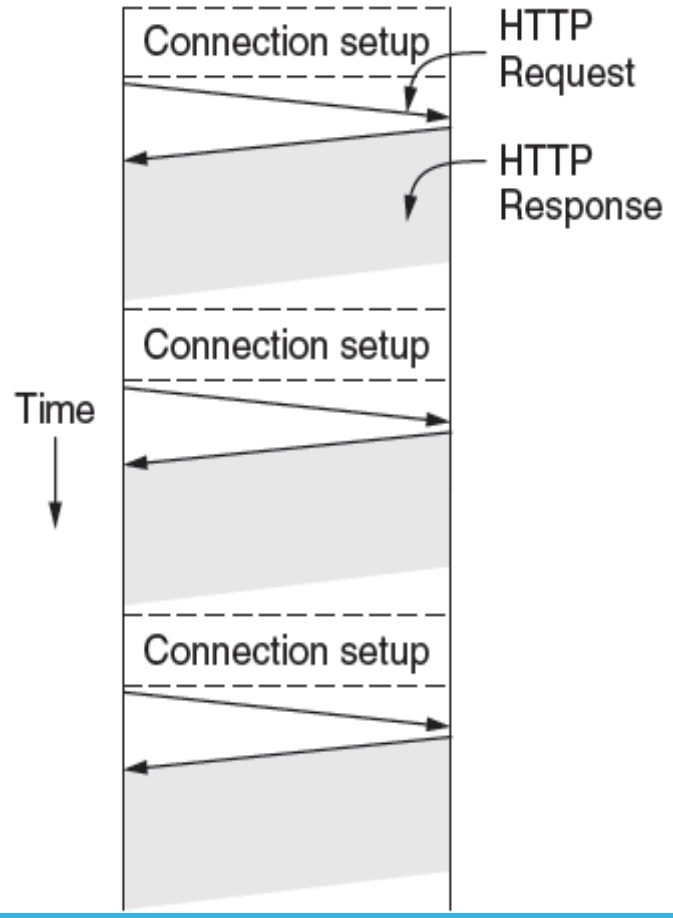
- Made HTTP very easy to build
- But gave fairly poor PLT …



EARLY PERFORMANCE

HTTP/1.0 used one TCP connection to fetch one web resource

- Made HTTP very easy to build
- But gave fairly poor PLT…



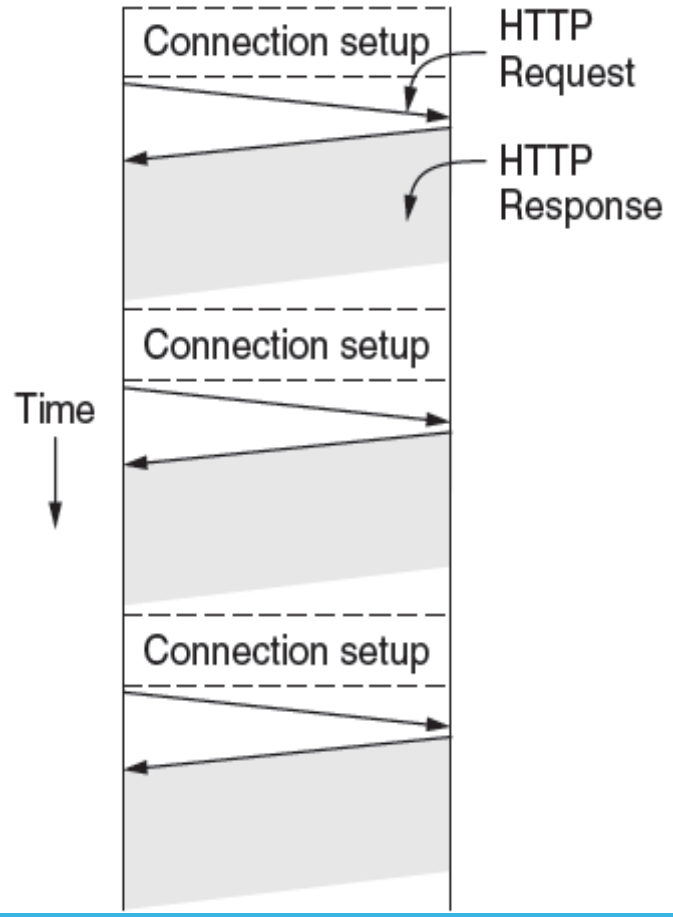
EARLY PERFORMANCE

Many reasons why PLT is larger than necessary

- Sequential request/responses, even when to different servers
- Multiple TCP connection setups to the same server

Network is not used effectively

- Worse with many small resources / page



WAYS TO DECREASE PLT

1. Reduce content size for transfer
   - Smaller images, gzip
2. Change HTTP to make better use of available bandwidth
3. Change HTTP to avoid repeated transfers of the same content
   - Caching, and proxies
4. Move content closer to client
   - CDNs

PARALLEL CONNECTIONS

One simple way to reduce PLT

- Browser runs multiple HTTP instances in parallel
- Server is unchanged; already handled concurrent requests for many clients

How does this help?

- Single HTTP wasn't using network much …
- So parallel connections aren't slowed much
- Pulls in completion time of last fetch

PERSISTENT CONNECTIONS

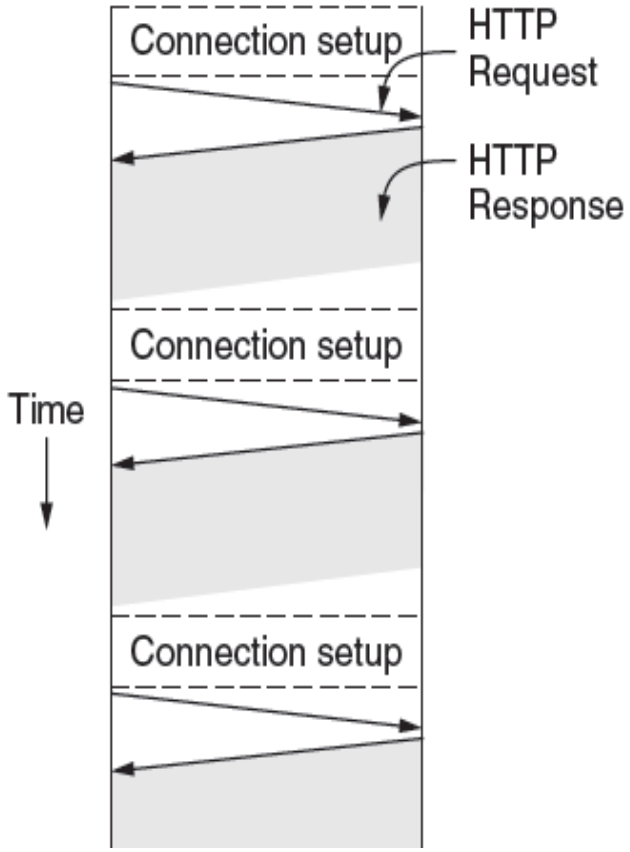
Parallel connections compete with each other for network resources

- Exacerbates network bursts, and loss
- Setup has a repeated overhead

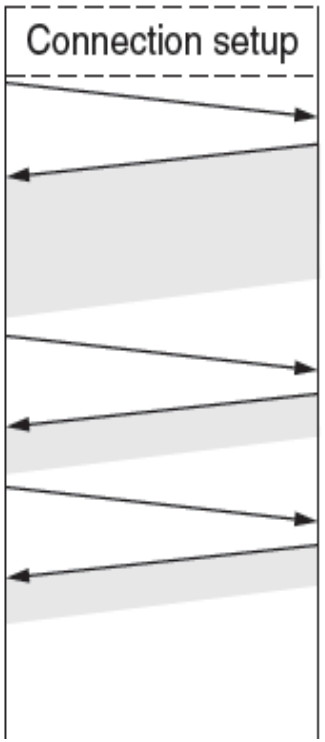
Persistent connection alternative

- Make 1 TCP connection to 1 server
- Use it for multiple HTTP requests
- Widely used in HTTP/1.1

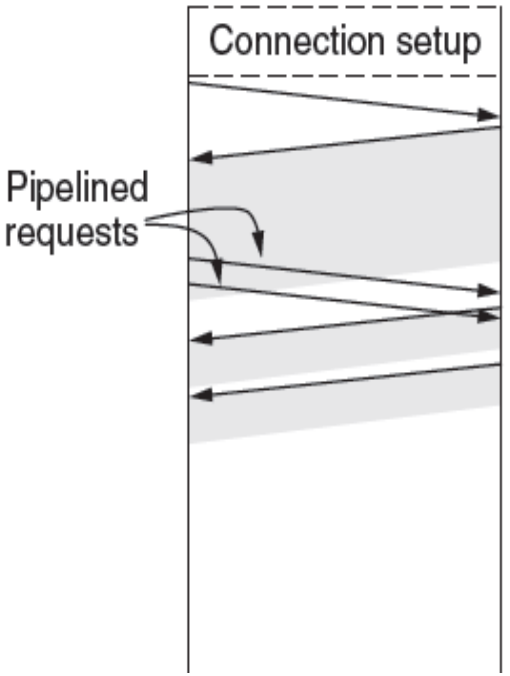
PERSISTENT CONNECTIONS



One request per connection



Sequential requests per connection

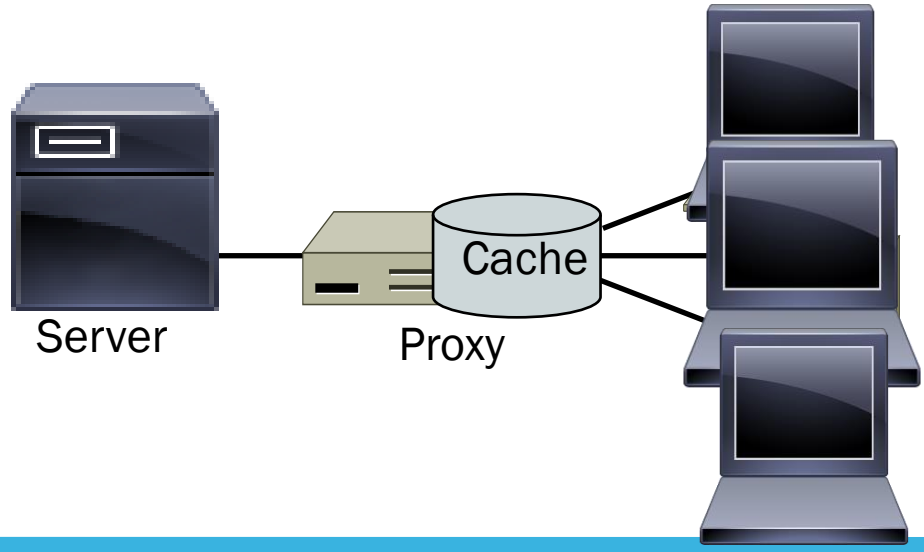


Pipelined requests per connection

WEB CACHING

A cache is hardware or software that is used to store something, usually data, temporarily in a computing environment

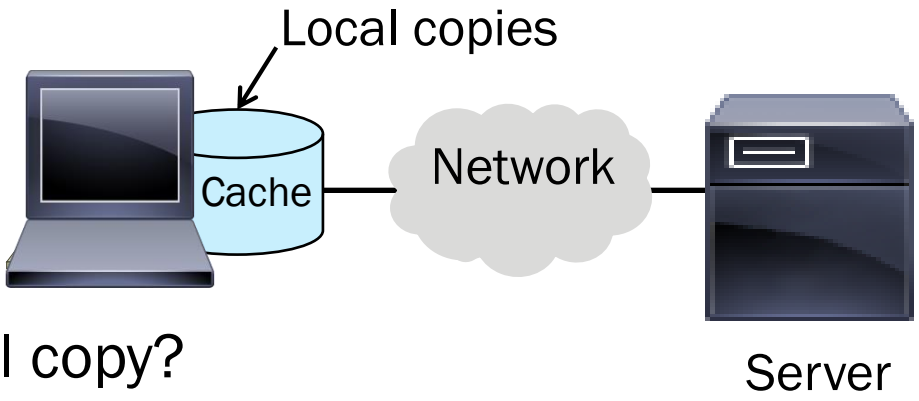
- **Two ways to cache web data/content**
  - Local HTTP cache
  - Proxy Cache
- **Enable Reuse**
  - Improve performance
    - Data is closer to user.



WEB CACHING

Users often revisit web pages

- Big win in performance from reusing local copy!
- This is caching



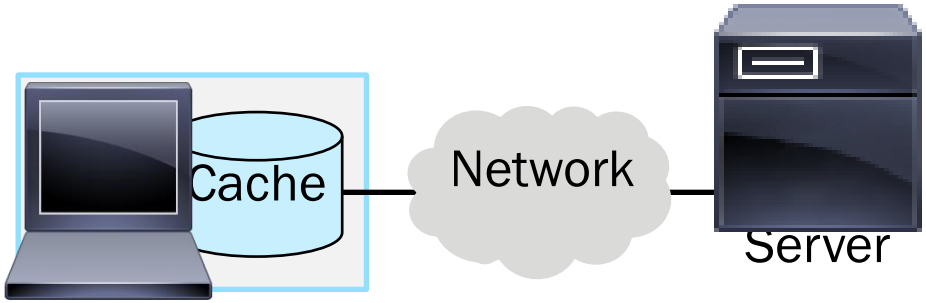
Key question:

- When is it OK to reuse local copy?

WEB CACHING

Locally determine copy is still valid

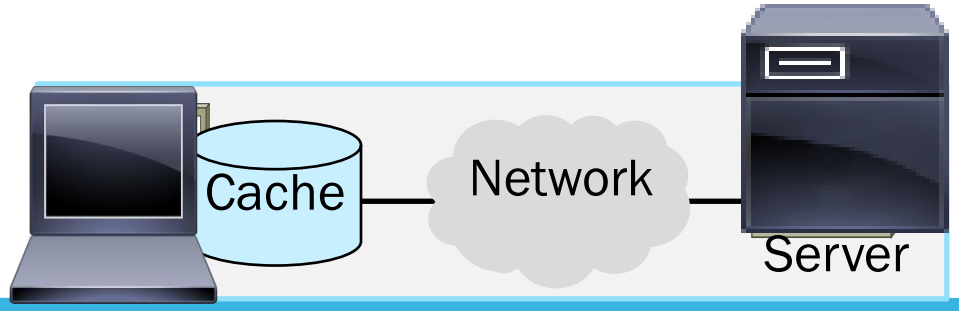
- Based on expiry information such as “Expires” header from server
- Or use a heuristic to guess (cacheable, freshly valid, not modified recently)
- Content is then available right away



WEB CACHING

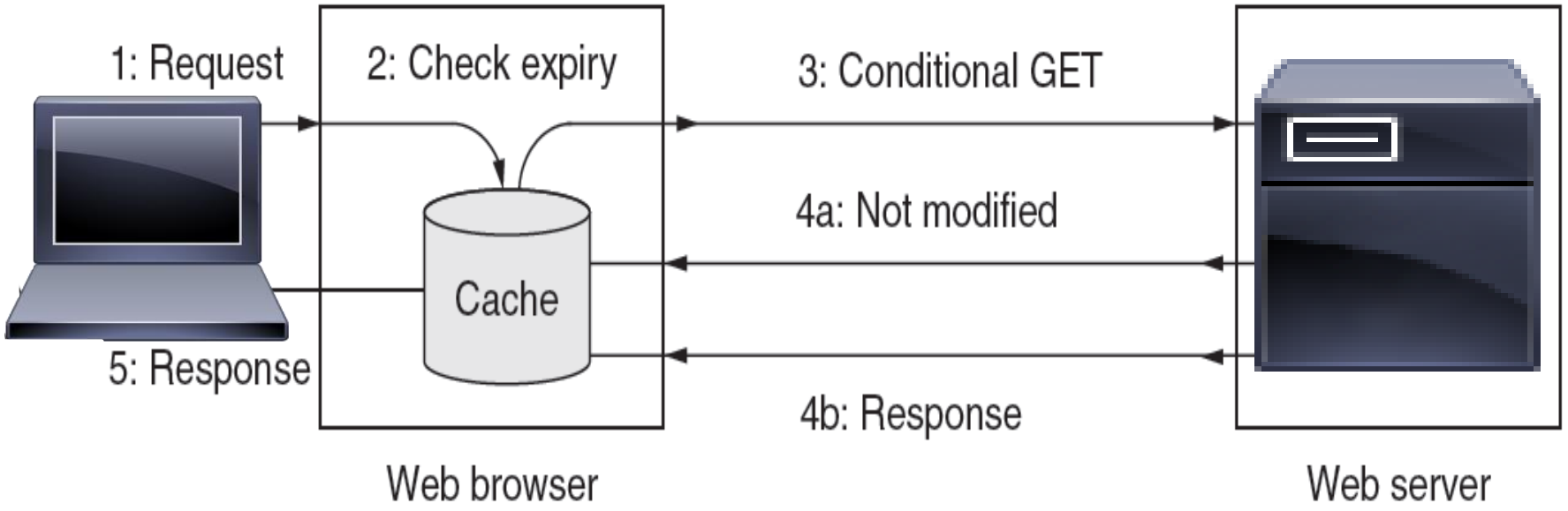
Revalidate copy with remote server

- Based on timestamp of copy such as “Last-Modified” header from server
- Or based on content of copy such as “Etag” header from server
- Content is available after 1 RTT



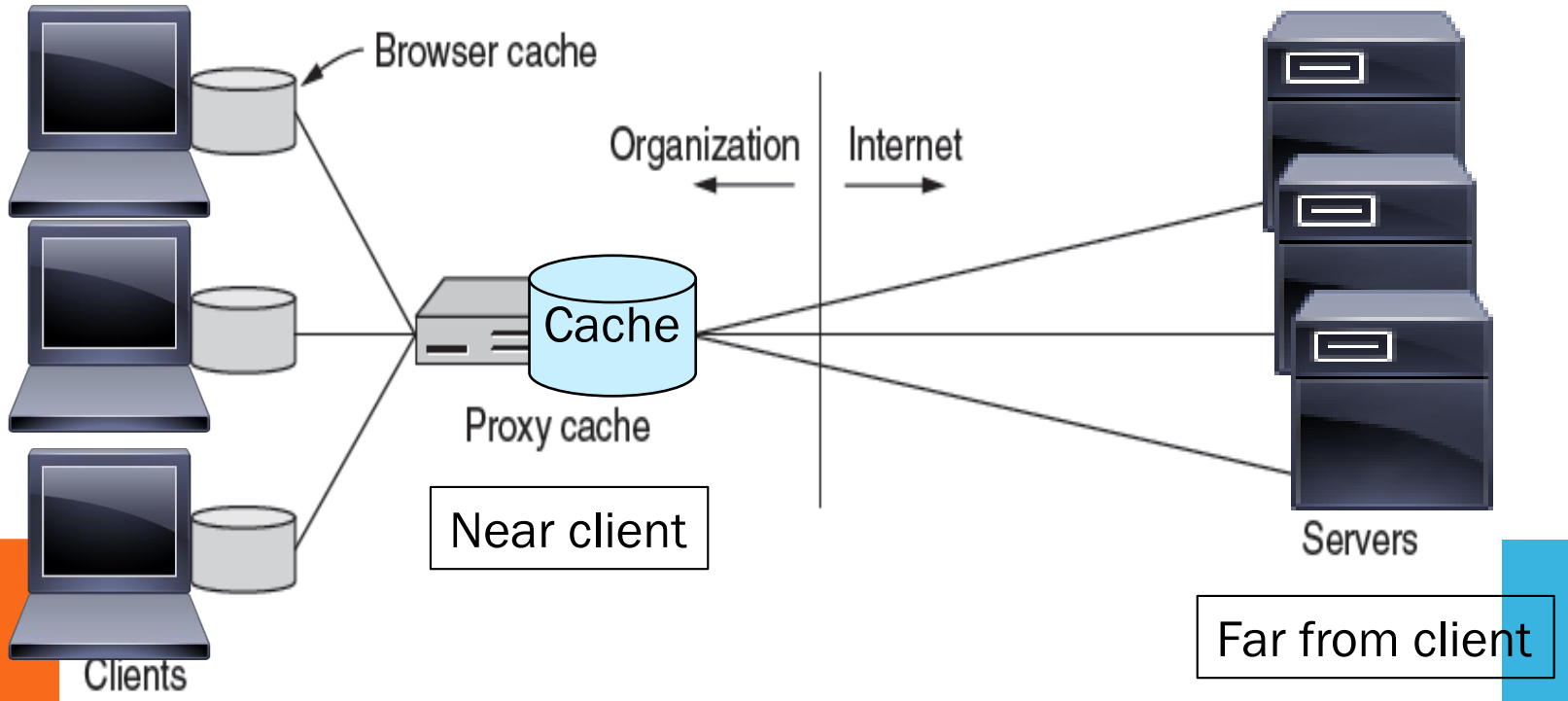
WEB CACHING

Putting the pieces together:



WEB PROXIES

Clients contact proxy; proxy contacts server



WEB PROXIES

Place intermediary between pool of clients and external web servers

- Benefits for clients include greater caching and security checking
- Organizational access policies too

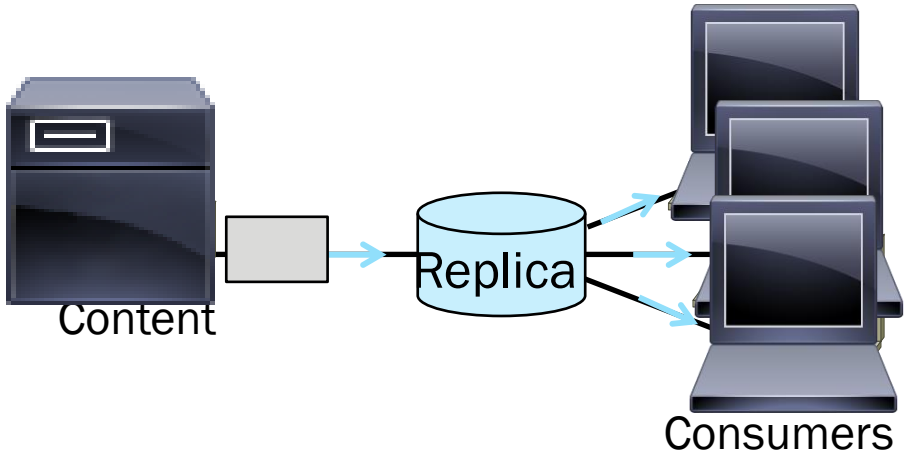
Proxy caching

- Clients benefit from larger, shared cache
- Benefits limited by secure / dynamic content, as well as “long tail”

CONTENT DELIVERY NETWORKS

CDNs (Content Delivery Networks)

- Efficient distribution of popular content; faster delivery for clients



CONTEXT

As the web took off in the 90s, traffic volumes grew and grew. This:

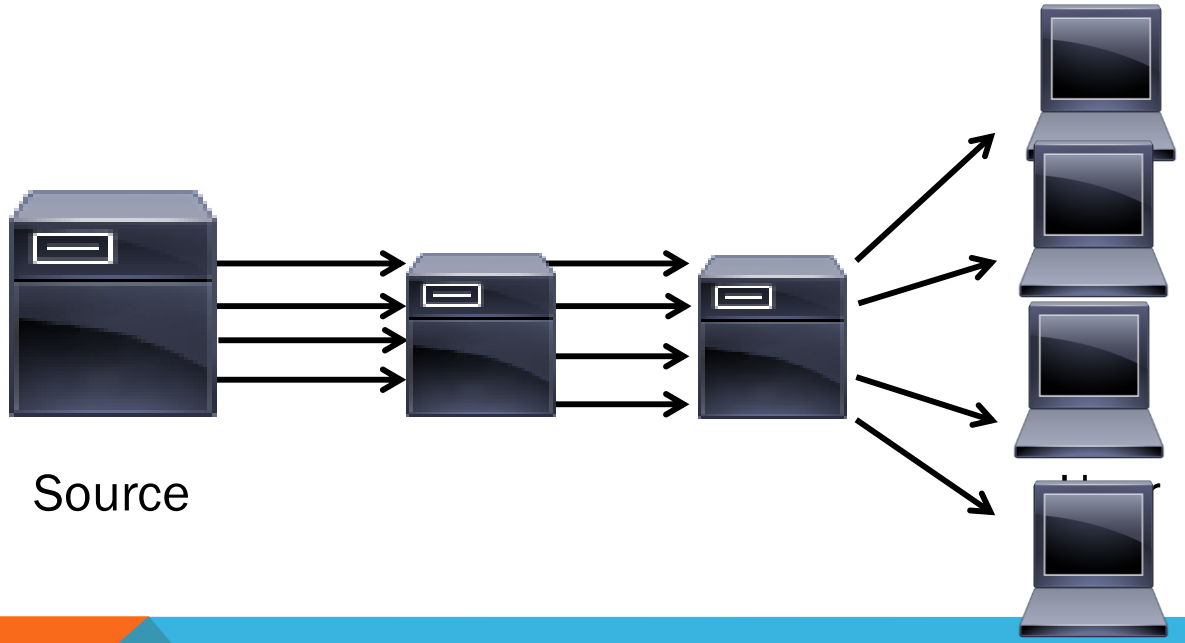
1. Concentrated load on popular servers
2. Led to congested networks and need to provision more bandwidth
3. Gave a poor user experience

Idea:

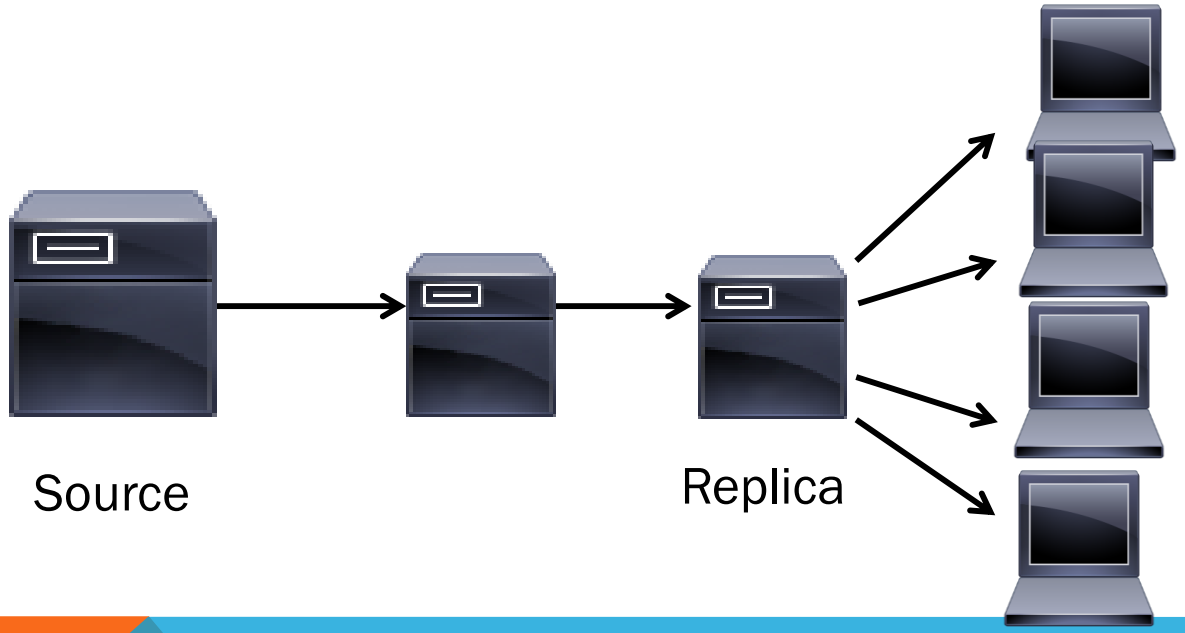
- Place popular content near clients
- Helps with all three issues above

BEFORE CDNS

Sending content from the source to 4 users takes 4 x 3 = 12 “network hops” in the example



AFTER CDNS



HOW TO PLACE CONTENT NEAR CLIENTS?

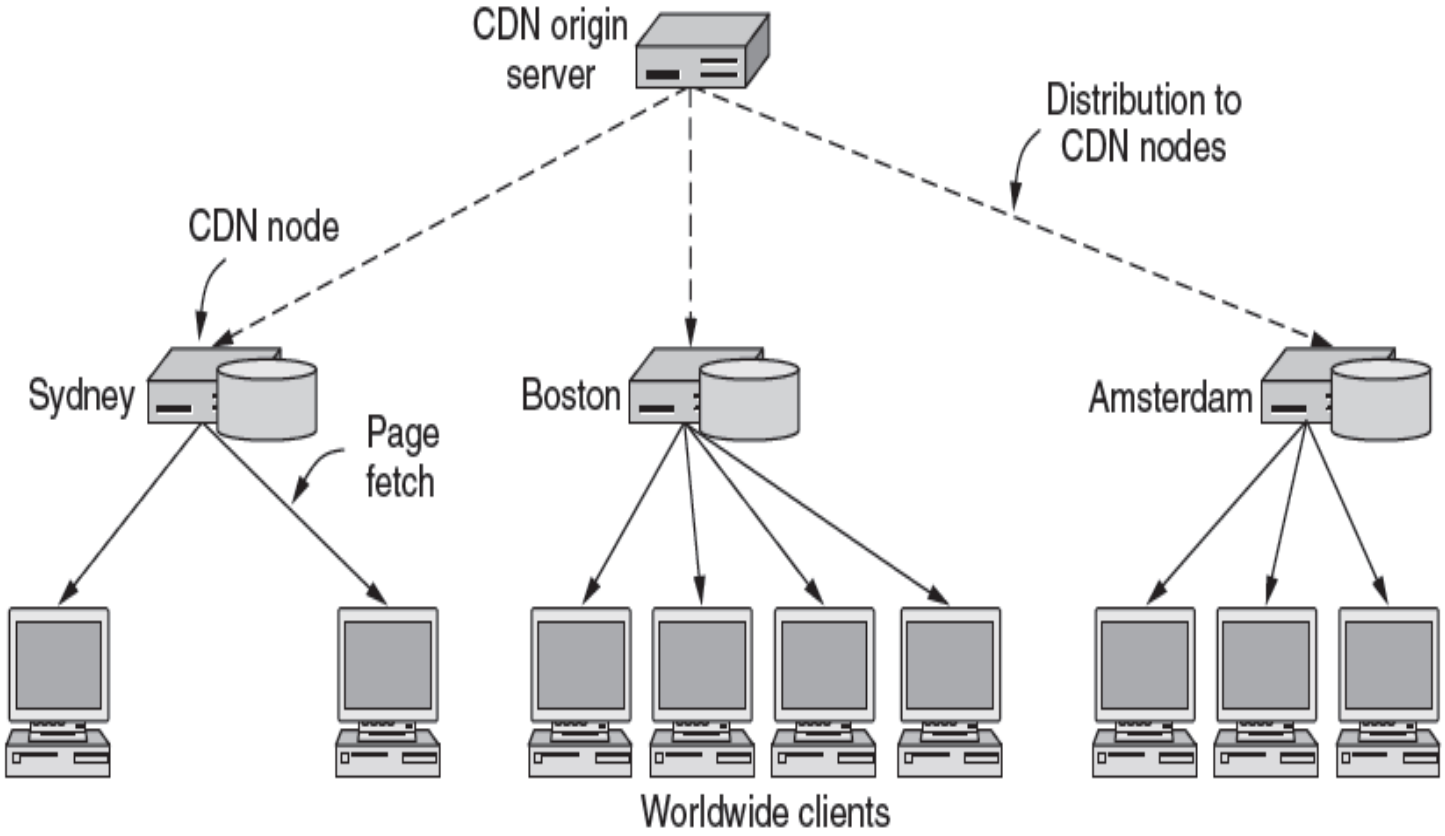
Use browser and proxy caches

- Helps, but limited to one client or clients in one organization

Want to place replicas across the Internet for use by all nearby clients

- Done by clever use of DNS

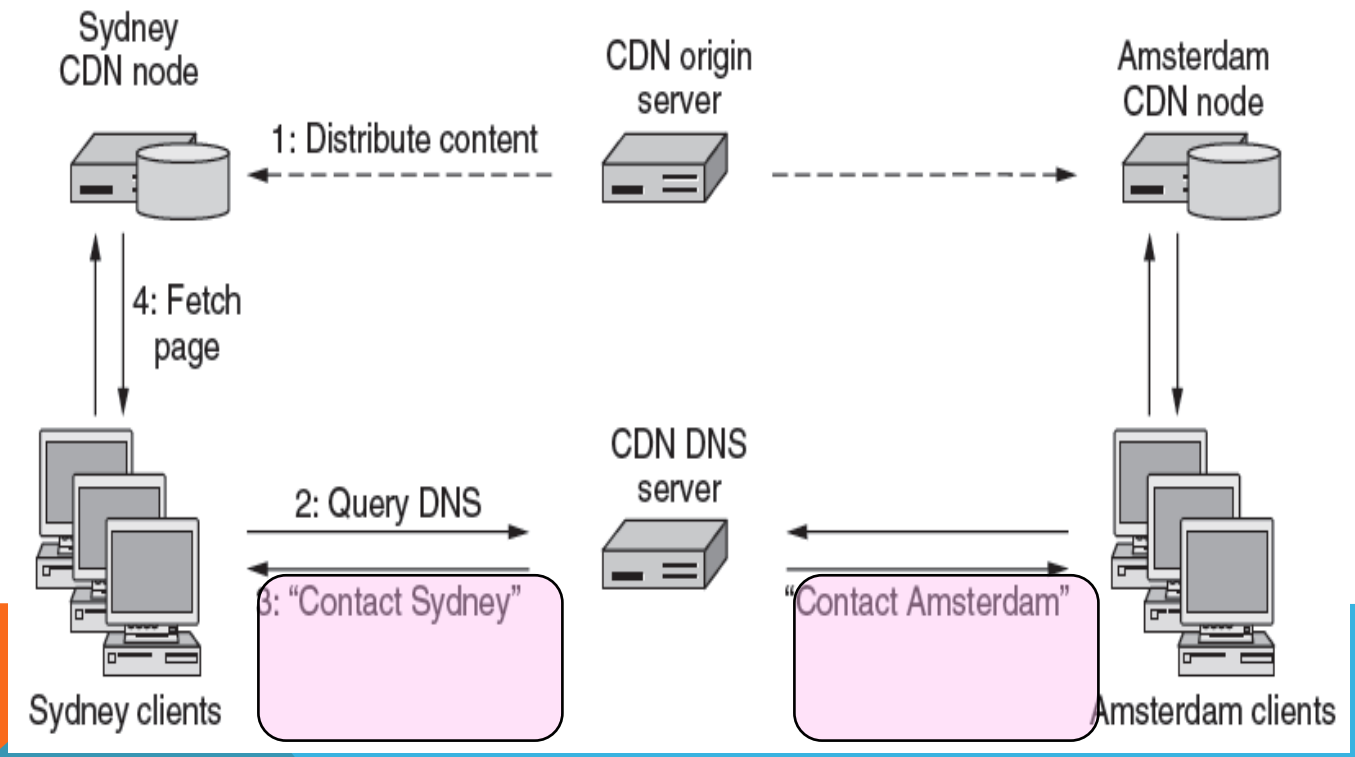
CONTENT DELIVERY NETWORK



CONTENT DELIVERY NETWORK

DNS resolution of site gives different answers to clients

- Tell each client the site is the nearest replica (map client IP)



BUSINESS MODEL

Clever model pioneered by Akamai

- Placing site replica at an ISP is win-win
- Improves site experience and reduces bandwidth usage of ISP