

(World Wide) Web

- a way to connect computers that provide information (servers) with computers that ask for it (clients like you and me)
 - uses the Internet, but it's not the same as the Internet
- URL (uniform resource locator, e.g., `http://www.amazon.com`)
 - a way to specify what information to find, and where
- HTTP (hypertext transfer protocol)
 - a way to request specific information from a server and get it back
- HTML (hypertext markup language)
 - a language for describing information for display
- browser (Firefox, Safari, Internet Explorer, Opera, Chrome, ...)
 - a program for making requests, and displaying results
- embellishments
 - pictures, sounds, movies, ...
 - loadable software
- the set of everything this provides

Web history

- 1989: Tim Berners-Lee at CERN
 - a way to make physics literature and research results accessible on the Internet
- 1991: first software distributions
- Feb 1993: Mosaic browser
 - Marc Andreessen at NCSA (Univ of Illinois)
- Mar 1994: Netscape
 - first commercial browser
- technical evolution managed by World Wide Web Consortium
 - non-profit organization at MIT, Berners-Lee is director
 - official definition of HTML and other web specifications
 - see `www.w3.org`

HTTP: Hypertext transfer protocol

- What happens when you click on a URL?
- client opens TCP/IP connection to host, sends request

```
GET /filename HTTP/1.0
```
- server returns
 - header info
 - HTML
- since server returns the text, it can be created as needed
 - can contain encoded material of many different types (MIME)
- URL format

```
service://hostname/filename?other_stuff
```
- `filename?other_stuff` part can encode
 - data values from client (forms)
 - request to run a program on server (cgi-bin)
 - anything else

e.g. `http://www.google.com/search?q=mime &ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a`



some detail on HTTP protocol

Request:

Request line: **method object protocol**
Headers: many options, most optional
empty line
message body (*optional*)

Example methods

GET retrieval
POST submitting data to be processed (in body)

Mandatory header

HOST URL sending request to

HTTP protocol: continuing some details

Response:

protocol status
Date:
Server: software information
Last-Modified:
Etag: determine cached version & current identical
Accept-Ranges:
Content-Length:
Connection: close
Content-Type: *Internet media type*

text of requested object

(A sample of header fields shown in blue)

Example from Wikipedia entry for HTTP:

Request:

`GET /index.html HTTP/1.1`
Host: `www.example.com`

Response

`HTTP/1.1 200 OK`
Date: `Mon, 23 May 2005 22:38:34 GMT`
Server: `Apache/1.3.3.7 (Unix) (Red-Hat/Linux)`
Last-Modified: `Wed, 08 Jan 2003 23:11:55 GMT`
Etag: `"3f80f-1b6-3e1cb03b"`
Accept-Ranges: `bytes`
Content-Length: `438`
Connection: `close`
Content-Type: `text/html; charset=UTF-8`

text of page

Embellishments

- **original design of HTTP just returns text to be displayed**
- **now includes pictures, sound, video, ...**
 - need helpers or plug-ins to display non-text content
e.g., GIF, JPEG graphics; sound; movies
- **forms filled in by user**
 - need a program on the server to interpret the information (cgi-bin)
- **HTTP is stateless**
 - server doesn't remember anything from one request to next
 - need a way to remember information on the client: cookies
- **active content: download code to run on the client**
 - Javascript and other interpreters
 - Java applets
 - plug-ins
 - ActiveX

Forms and CGI programs

- **"common gateway interface"**
 - standard way to request the server to run a program
 - using information provided by the client via a form
- **if the target file on server is an executable program**
 - e.g., in /cgi-bin directory
- **or if it has the right kind of name**
 - e.g., something.cgi
- **run it on the server to produce HTML to send back to client**
 - using the contents of the form as input
 - output depends on client request: created on the fly, not just a file
- **CGI programs can be written in any programming language**
 - often Perl, PHP, Java

Example CGI program in Perl (mailform.cgi modified)

```
#!/usr/local/bin/perl -w
use CGI;
my $query = new CGI;
print $query->header;
print $query->start_html(-title=>'Form results');
print "<h1> Form results </h1>\n";
my $urcomp = $query->remote_host();
my $urIP = $query->remote_addr();
print "<P> Your computer is $urcomp\n";
print "<P> Your IP address is $urIP\n";
print "<P>\n";
foreach $name ($query->param) {
    print "<br> $name:";
    foreach $value ($query->param($name)) {
        print " $value";
    }
    print "\n";
}
```

Web pages: Information passed and actions initiated

- **HTTP requests identify host and address:**
 - my \$urcomp = \$query->remote_host();
 - my \$urIP = \$query->remote_addr();
- **Initiate actions with Javascript**
 - onmouseover etc
- **Links with "extra"**
 - Google ads

Cookies

- **HTTP is stateless: doesn't remember from one request to next**
- **cookies intended to deal with stateless nature of HTTP**
 - remember preferences, manage "shopping cart", etc.
- **cookie: one line of text sent by server to be stored on client**
 - stored in browser while it is running (transient)
 - stored in client file system when browser terminates (persistent)
- **when client reconnects to same domain,**
 - browser sends the cookie back to the server**
 - sent back verbatim; nothing added
 - sent back only to the same domain that sent it originally
 - contains no information that didn't originate with the server
- **in principle, pretty benign**
- **but heavily used to monitor browsing habits, for commercial purposes**

Cookie crumbs

- **get a page from xyz.com**
 - it contains
 - this causes a page to be fetched from DoubleClick.com
 - which now knows your IP address and what page you were looking at
- **DoubleClick sends back a suitable advertisement**
 - with a cookie that identifies "you" at DoubleClick
- **next time you get any page that contains a doubleclick.com image**
 - the last DoubleClick cookie is sent back to DoubleClick
 - the set of sites and images that you are viewing is used to
 - update the record of where you have been and what you have looked at
 - send back targeted advertising (and a new cookie)
- **this does not necessarily identify you personally so far**
- **but if you ever provide personal identification, it can be (and will be) attached**
- **defenses:**
 - turn off all cookies; turn off "third-party" cookies
 - don't reveal information
 - clean up cookies regularly

Cookie crumbs (2)

- **modern versions are very dynamic**
 - e.g., Yahoo Right Media, Doubleclick Ad Exchange, ...
- **person requests a web page**
- **web page publisher notifies exchange that space on that page is available**
 - might also include information about the person, like
 - past online activity, viewing and shopping habits, geographical location, demographics, maybe even actual identity
- **advertisers bid on the ad space**
 - amount depends on person's attributes and location, ad budget, etc.
- **winner's advertisement inserted into the page**

- **elapsed time: 10-100 milliseconds?**

Cookie crumbs (3)

- **other kinds of tracking tools**

- **web bugs, web beacons, single-pixel gifs**
 - tiny image that reports the use of a particular page
 - these can be used in mail messages, not just browsers

- **Flash cookies ("local shared object")**
 - cookie-like mechanism used by Flash
 - Save up to 100KB vs 4KB regular cookies
 - Must go to their site to control (lab 8)
 - Going to their site gives them info about you
 - Set allowed disk space to 0 for specific domain
 - still allows empty directory with domain name (Wikipedia)

Plugins

- **programs that extend browser, mailer, etc.**
 - browser provides API, protocol for data exchange
 - extension focuses on specific application area
 - e.g., documents, pictures, sound, movies, scripting language, ...
 - may exist standalone as well as in plugin form
 - Acrobat, Flash, Quicktime, RealPlayer, Windows Media Player, ...

- **scripting languages interpret downloaded programs**
 - Javascript
 - Java
 - compiled into instructions for a virtual machine (like toy machine on steroids)
 - instructions are interpreted by virtual machine in browser

Active X (Microsoft)

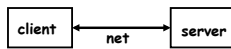
- **write programs in any language (C, C++, Visual Basic, ...)**
- **compile into machine instructions for PC**
- **when a web page that uses an ActiveX object is accessed**
 - browser downloads compiled native machine instructions
 - checks that they are properly signed ("authenticated") by creator
 - runs them

- **each ActiveX object comes with digital certificate from supplier**
 - can't be forged
 - run the program if you trust the supplier
- **more efficient than an interpreter**
- **no restrictions on what an ActiveX object can do**
 - no assurance that it works properly!

- **the most risky of the active-content models**

Potential security & privacy problems

- **attacks against client**
 - release of client information
 - cookies: client remembers info for subsequent visits to same server
 - adware, phishing, spyware, viruses, ...
 - spyware: client sends info to server upon connection (Sony, ...)
 - often from unwise downloading
 - buggy/misconfigured browsers, etc., permit vandalism, theft, hijacking, ...
- **attacks against server**
 - client asks server to run a programs when using cgi-bin
 - server-side programming has to be careful
 - buggy code on server permits break-in, theft, vandalism, hijacking, ...
 - denial of service attacks
- **attacks against information in transit**
 - eavesdropping
 - encryption helps
 - masquerading
 - needs authentication in both directions



Privacy on the Web

- **what does a browser send with a Web request?**
 - IP address, browser type, operating system type
 - referrer (URL of the page you were on)
 - cookies

- **what do "they" know about you?**
 - whatever you tell them, implicitly or explicitly
 - public records are really public
 - lots of big databases like phone books
 - universal numbers make it easier to track you (SSN, telephone, Ethernet)
 - log files everywhere
 - aggregators really collect a lot of information for advertising
 - spyware, key loggers and similar tools collect for nefarious purposes

- **who owns your information?**
 - in the USA, they do

Viruses

- **old threat, new technologies**
 - new connectivity makes them more dangerous
- **basic problem: running someone else's software on your machine**
 - bugs and ill-advised features make it easier
- **operates by hiding executable code inside something benign**
 - e.g., .EXE file or script in mail or document, downloaded content
- **Melissa, ILoveYou, Anna Kournikova viruses use Visual Basic**
 - applications (Word, Excel, Powerpoint, Outlook) have VB interpreter
 - a document like a .doc file or email message can contain a VB program
 - opening the document causes the VB program to be run
- **virus detectors**
 - scan for suspicious patterns, suspicious activities, changes in files

A list of malware

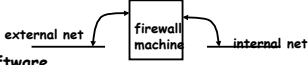
see *The Difference Between a Virus, Worm and Trojan Horse* - Webopedia.com

- **Virus: within or attached to another program or medium**
 - Must run it or open document (etc.) that causes it to run
 - Spread when vehicle sent around
- **Worm: program that spreads from computer to computer w/out human action**
 - Self-replicating
 - uses a system mechanism to send files or information between computers
e.g. send copy of self to everyone in your email address book
 - can overwhelm computer memory or network bandwidth
- **Trojan horse: program that presents as a legitimate program without your knowledge and sends over network to**
 - voluntarily download/open thinking something want
 - Does something bad
- **Spyware: program that gathers information about your system without your knowledge and sends over network to**
 - Usually download with other software
- **Adware: form of spyware that gathers info for ad placement**

Bots, botnets, etc.

- **bots: software robots running automated tasks over Internet**
 - e.g., web spider collecting web page info for search engines
- **botnet: collection of "zombie" computers that can be controlled remotely**
 - most often Windows PCs
 - infected via viruses, worms, trojan horses, etc.
 - controlled by chat protocol, web page visits, peer to peer
 - exploits include denial of service attacks, spam, click fraud, adware, spyware, ...

Defenses

- use strong passwords
 - popups off, cookies off, spam filter on
 - turn off previewers and HTML mail readers
 - anti-virus software on and up to date
 - turn on macro virus protection in Word, etc.; turn off ActiveX
 - run spyware detectors
 - use a firewall
- 
- ```
graph LR;
 external[external net] --> firewall[firewall machine];
 firewall --> internal[internal net];
```
- try less-often targeted software
    - Mac OS X, Linux, Firefox, Thunderbird, ...
  - **be careful and suspicious all the time**
    - don't view attachments from strangers
    - don't view unexpected attachments from friends
    - don't just read/accept/click/install when requested
    - don't install file-sharing programs
    - be wary when downloading any software

## Important Web-related activities

- Web crawling
- Cloud computing

## Crawling the Web

- Search engines must gather the documents that they index and search
- Retrieve documents by following links document to document

```
start with a list of likely URLs
While list not empty {
 fetch data from next URL from the list
 extract parts to be indexed, deliver to index builder
 extract URLs
 delete duplicate URLs (ones seen recently)
 delete irrelevant ones (advertisements, ...)
 add remaining URLs to end of list
}
```

### Main Issues I

- **starting set of pages?**
  - a.k.a "seed" URLs
- **How detect duplicates quickly**
- **can visit whole of Web?**
- **how determine order to visit links?**
  - graph model:
    - breadth first vs depth first
    - what are pros and cons of each?
    - "black holes"
  - other aspects /considerations
    - how deep want to go?
    - associate priority with links
    - what kind of files save?

25

### "Black holes" and other "baddies"

- **"Black hole": Infinite chain of pages**
  - dynamically generated
  - not always malicious
    - link to "next month", which uses perpetual calendar generator
- **Other bad pages**
  - other behavior damaging to crawler?
    - servers
  - spam content
    - use URLs from?

**Robust crawlers must deal with black holes and other damaging behavior**

26

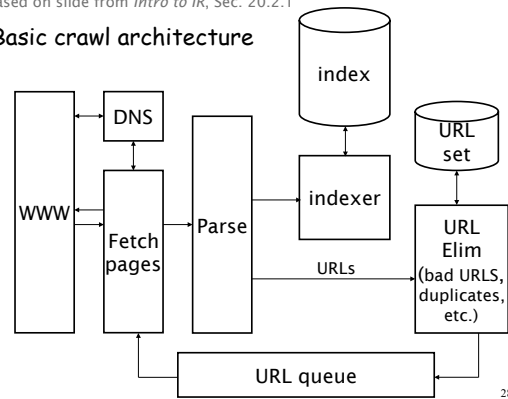
### Main Issues II

- **Web is dynamic**
  - time to crawl "once"
  - how mix crawl and re-crawl
  - priority of pages
- **Social behavior**
  - crawl only pages allowed by owner
    - robot exclusion protocol: *robots.txt*
  - not flood servers
    - expect many pages to visit on one server

27

Based on slide from *Intro to IR*, Sec. 20.2.1

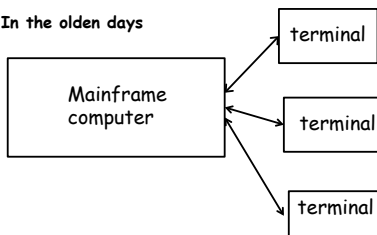
### Basic crawl architecture



28

### Cloud computing

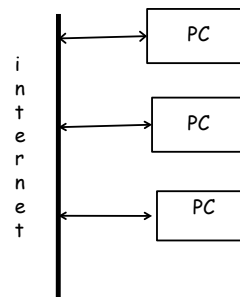
- **In the olden days**



Mainframe owned by company

### Cloud computing

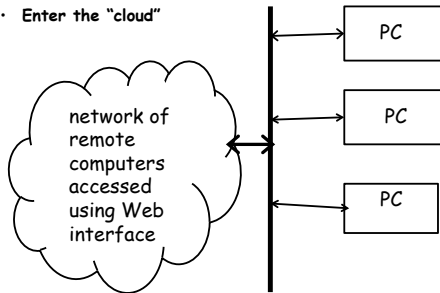
- **Then**



Self-sufficient personal computers, communicate through networks

## Cloud computing

- Enter the "cloud"



- Applications run on computer(s) accessed over internet by Web protocols

Wikipedia: [cloud computing](#)

"A type of computing, comparable to grid computing that relies on [sharing computing resources](#) rather than having local servers or personal devices to handle applications. The goal of cloud computing is to [apply](#) traditional supercomputing, or [high-performance computing power](#), normally used by military and research facilities, to perform tens of trillions of computations per second, [in consumer-oriented applications](#) such as financial portfolios or even to deliver personalized information, or power immersive computer games.

"To do this, cloud computing [networks large groups of servers](#), usually those with low-cost consumer PC technology, with specialized connections to [spread data-processing chores](#) across them. This shared IT infrastructure contains [large pools of systems that are linked together](#). Often, virtualization techniques are used to maximize the power of cloud computing."

## Cloud computing pros

- Can have access to powerful computers for calculation
- Can have access to large amounts of storage
- Do not need "high-end" personal computer
  - reduce cost
  - reduce applications you need locally
  - reduce upgrades for hardware
  - reduce risk of malware
- Easy to coordinate interaction with others
  - joint authorship (Google docs)
  - distributed organization (Google calendar)

A sea change?

## Cloud computing cons

- Someone else has your programs and/or data
  - trust?
  - if provider goes away?
  - if provider is acquired?
- Privacy concerns: data can be aggregated across applications
  - Search, email, documents, calendar ...

## Wrap-up: Web basics

- standard protocol and exchange format so [servers that have information can provide it to clients that request it](#)
  - hypertext transfer protocol ( HTTP ) on top of internet TCP/IP
  - information identified through uniform resource locator ( URL )
  - technical evolution managed by World Wide Web Consortium ( W3C )
- [browsers make requests and display results](#)
  - originally HTML
  - expanded to other media: plug-ins for browsers
  - expanded to forms interpreted by server
  - expanded to active content like Javascript run on client
- [Web one of main sources of security and privacy problems](#)
  - HTTP stateless but sends identifying info to server
  - cookies and other means for server to remember client
  - malware downloaded through Web, but many other vehicles too: email ...
- [from Web have developed new functionality](#)
  - search engines
  - cloud computing