CS 425 / ECE 428 Distributed Systems Fall 2017

Indranil Gupta (Indy)

Lecture 1: Welcome, and Introduction

Sep 5, 2017

Web: courses.engr.illinois.edu/cs425/

Course Staff

- Myself (*Indy*): 3112 Siebel Center
  - Office Hours Every Tuesday and Thursday right after class – see website
- TAs: (see course website for office hours)
  - *Faria Kalim (lead TA)*
  - *Rui Yang*
  - *Yao Qian*
  - *Ashwini Raina*
  - *Akshun Gupta*
  - *Jayasi Mehar (TA for MCS-Online and MCS-DS)*
  - *Mayank Bhatt (TA for MCS-Online and MCS-DS)*
- More about course logistics later in this lecture…

What This Course is About

- Sports
- Travel to Mars
- Job Interviews
- Company Acquisitions
- (Not Kidding)

What This Course is Really About

- Distributed Systems
- How to Design Algorithms for them
- How to Design The Systems
- How they work in real life
- How to build real distributed systems

Our Main Goal Today

To Define the Term Distributed System

Can you name some examples of Operating Systems?

Can you name some examples of Operating Systems?

…

Linux WinXP Vista 7/8 Unix FreeBSD macOS OSX

2K Aegis Scout Hydra Mach SPIN

OS/2 Express Flux Hope Spring

AntaresOS EOS LOS SQOS LittleOS TINOS

PalmOS WinCE TinyOS iOS

…

What is an Operating System?

What is an Operating System?

- User interface to hardware (device driver)
- Provides abstractions (processes, file system)
- Resource manager (scheduler)
- Means of communication (networking)
- …

FOLDOC definition

(FOLDOC = Free On-Line Dictionary of Computing)

Operating System - The low-level software which handles the interface to peripheral hardware, schedules tasks, allocates storage, and presents a default interface to the user when no application program is running.

Can you name some examples of Distributed Systems?

Can you name some examples of Distributed Systems?

- Client-Server (NFS)
- The Web
- The internet
- A wireless network
- DNS
- Gnutella or BitTorrent (peer to peer overlays)
- A “cloud”, e.g., Amazon EC2/S3, Microsoft Azure
- A datacenter, e.g., NCSA, a Google datacenter, AWS

What is a Distributed System?

FOLDOC definition

A collection of (probably heterogeneous) automata whose distribution is transparent to the user so that the system appears as one local machine. This is in contrast to a network, where the user is aware that there are several machines, and their location, storage replication, load balancing and functionality is not transparent. Distributed systems usually use some kind of client-server organization.

Textbook definitions

- A distributed system is a collection of independent computers that appear to the users of the system as a single computer.

  [Andrew Tanenbaum]

- A distributed system is several computers doing something together. Thus, a distributed system has three primary characteristics: multiple computers, interconnections, and shared state.

  [Michael Schroeder]

Unsatisfactory

- Why are these definitions short?
- Why do these definitions look inadequate to us?
- Because we are interested in the insides of a distributed system
  - design and implementation
  - Maintenance
  - Algorithmics (“protocols” or “distributed algorithms”)

"I shall not today attempt further to define the kinds of material I understand to be embraced within that shorthand description; and perhaps I could never succeed in intelligibly doing so. But I know it when I see it, and the motion picture involved in this case is not that."

[Potter Stewart, Associate Justice, US Supreme Court (talking about his interpretation of a technical term laid down in the law, case Jacobellis versus Ohio 1964)]

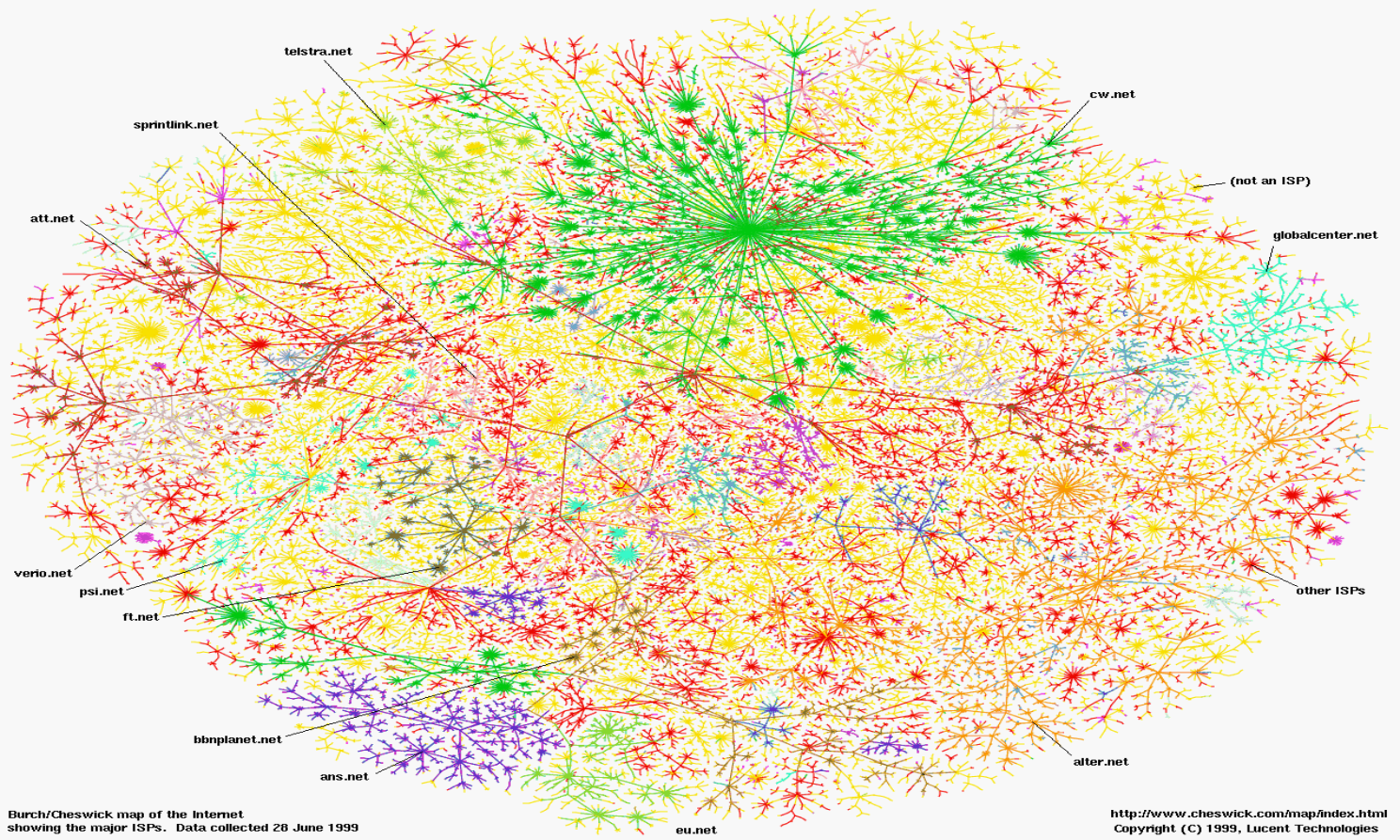
Which is a Distributed System – (A) or (B)?

(A)

(A) Facebook Social Network Graph among humans

Source: https://www.facebook.com/note.php?note_id=469716398919

(B)



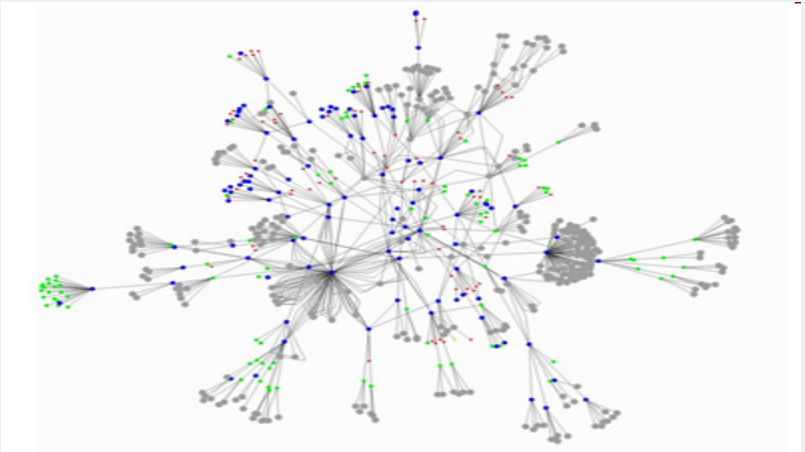
(B) The internet (Internet Mapping Project, color coded by ISPs)

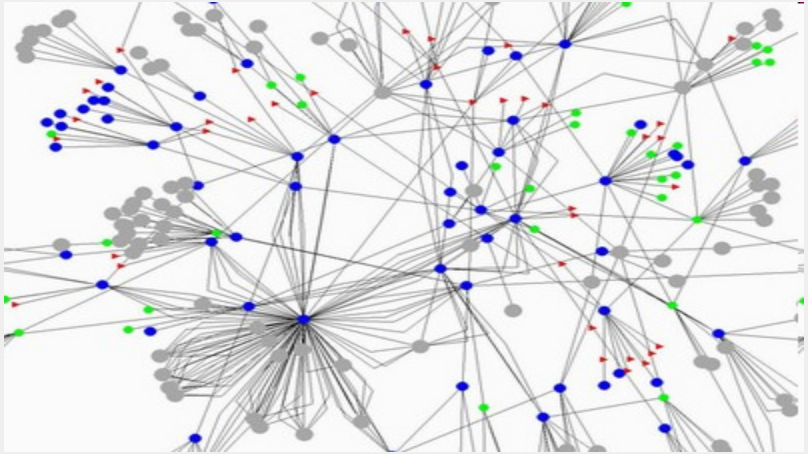
A working definition for us

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

- Entity=a process on a device (PC, PDA)
- Communication Medium=Wired or wireless network
- Our interest in distributed systems involves
  - design and implementation, maintenance, algorithmics

Gnutella Peer to Peer System





What are the "entities" (nodes)?

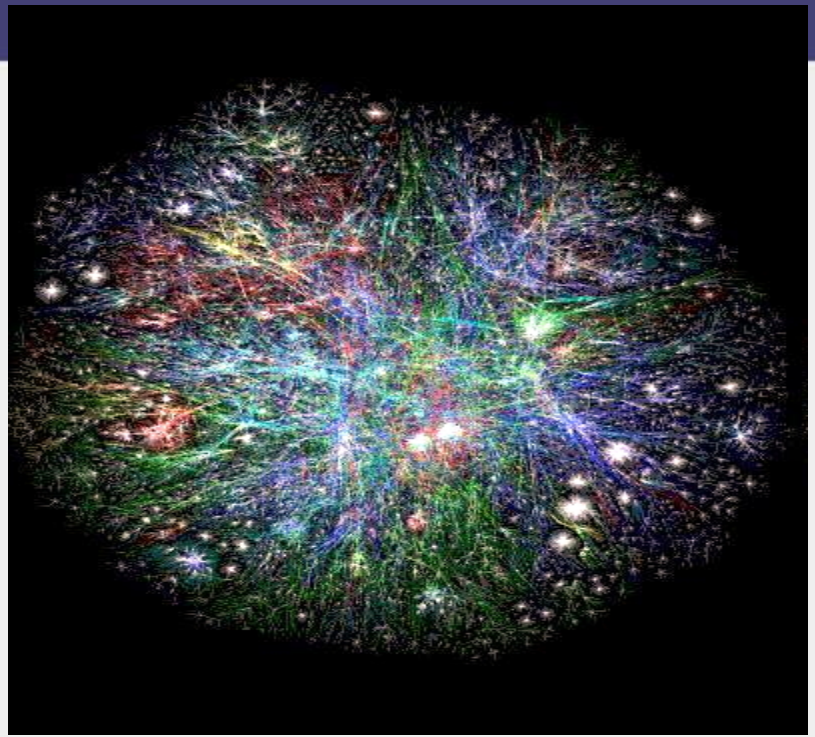
What is the communication medium (links)?

Source: GnuMap Project

Web Domains

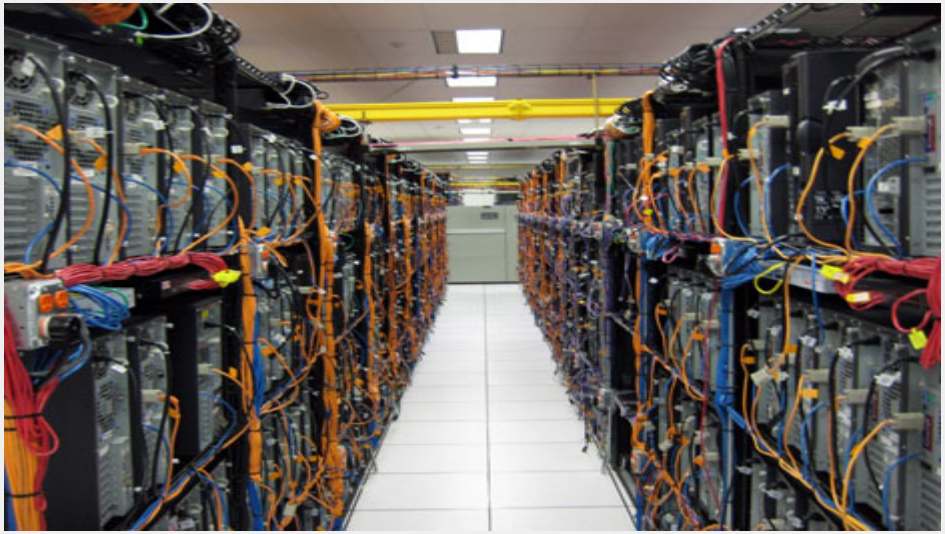
What are the “entities” (nodes)?

What is the communication medium (links)?



Source: http://www.vlib.us/web/worldwideweb3d.html

Datacenter



What are the "entities" (nodes)?

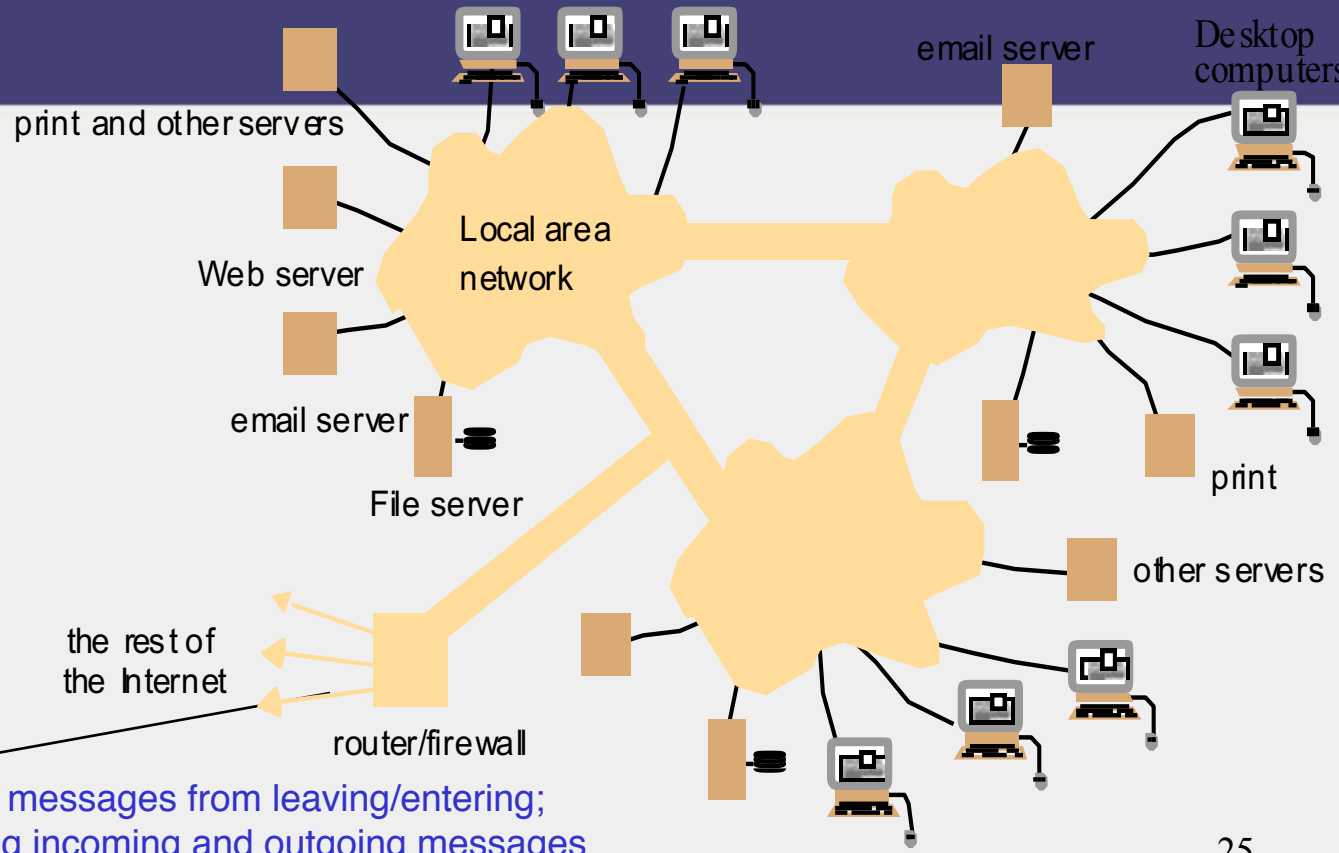
What is the communication medium (links)?

The internet – Quick Refresher

- Underlies many distributed systems.
- A vast interconnected collection of computer networks of many types.
- Intranets – subnetworks operated by companies and organizations.
- Intranets contain LANs (local area networks).
- WAN – wide area networks, consists of subnets (intranets, LANs, etc.)
- ISPs – Internet Service Providers. Companies that provide modem links and other types of connections to users.
- Intranets (actually the ISPs' core routers) are linked by backbones – network links of large bandwidth, such as satellite connections, fiber optic cables, and other high-bandwidth circuits.
- UC2B? Google Fiber? (MAN = Metropolitan Area Networks)

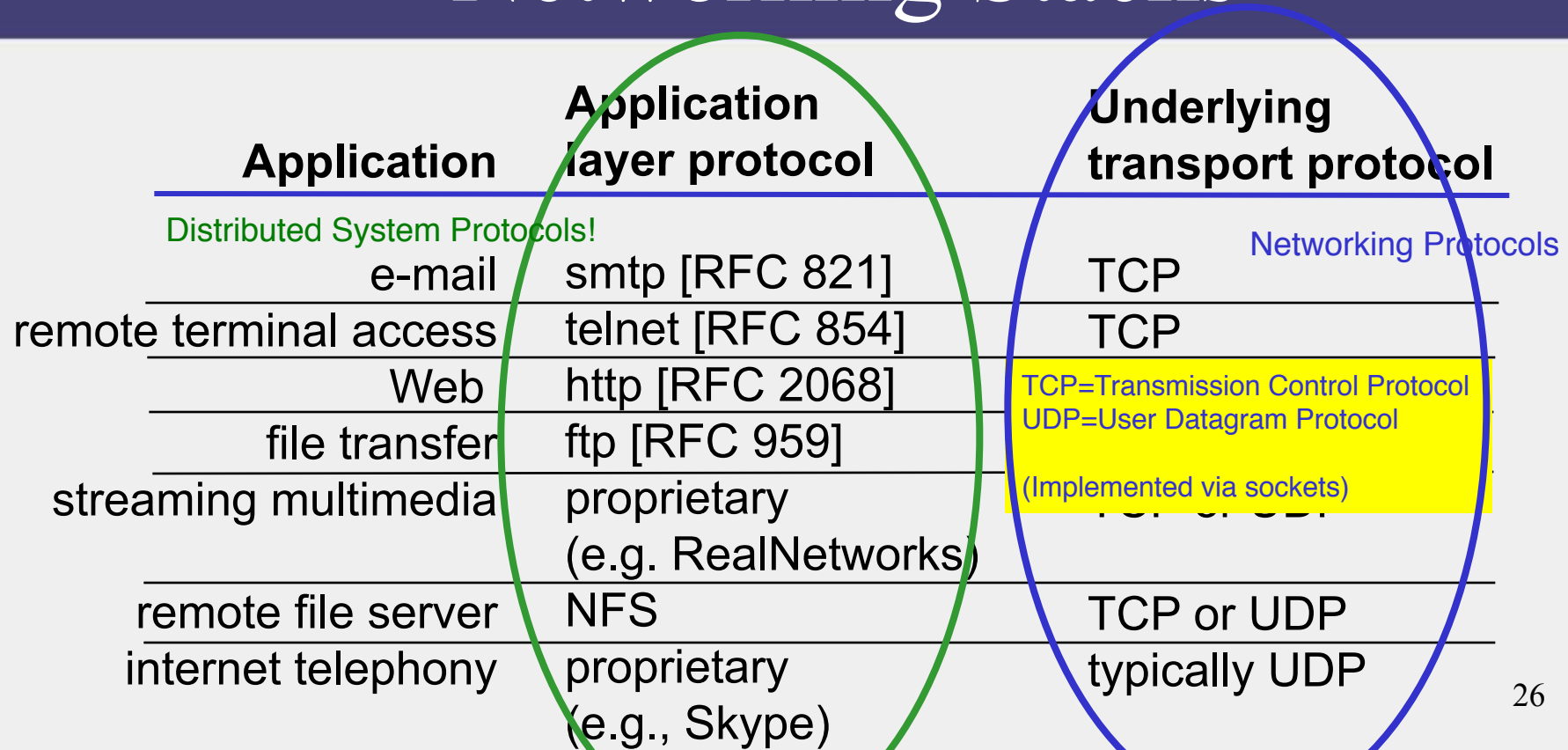
An Intranet & a distributed system

Running over this Intranet is a distributed file system

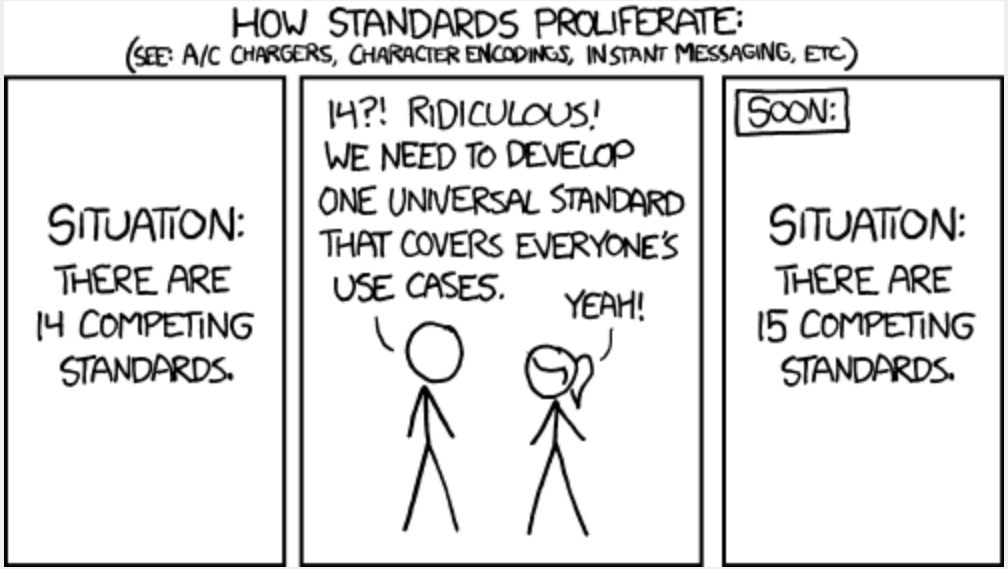


prevents unauthorized messages from leaving/entering; implemented by filtering incoming and outgoing messages via firewall “rules” (configurable)

Networking Stacks



The History of internet Standards

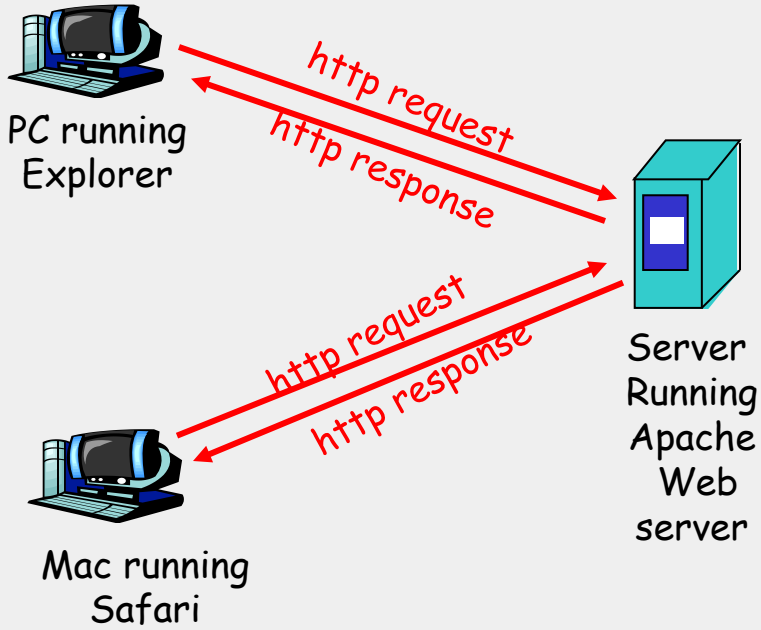


Source: http://xkcd.com/927/

The Heart of the World Wide Web: the HTTP Standard

HTTP: hypertext transfer protocol

- WWW's application layer protocol
- client/server model
  - *client:* browser that requests, receives, and "displays" WWW objects
  - *server:* WWW server, which is storing the website, sends objects in response to requests
- http1.0: RFC 1945
- http1.1: RFC 2068
  - Leverages same connection to download images, scripts, etc.



The HTTP Protocol: More

http: TCP transport service:

- client initiates a TCP connection (creates socket) to server, port 80
- server accepts the TCP connection from client
- http messages (application-layer protocol messages) exchanged between browser (http client) and WWW server (http server)
- TCP connection closed

http is "stateless"

- server maintains no information about past client requests

Why?

Protocols that maintain session "state" are complex!

- **past history (state) must be maintained and updated.**
- **if server/client crashes, their views of "state" may be inconsistent, and hence must be reconciled.**
- **RESTful protocols are stateless.**

HTTP Example

Suppose user enters URL www.cs.uiuc.edu/ (contains text, references to 10 jpeg images)

1a. http client initiates a TCP connection to http server (process) at www.cs.uiuc.edu. Port 80 is default for http server.

1b. http server at host www.cs.uiuc.edu waiting for a TCP connection at port 80. "accepts" connection, notifying client

2. http client sends a http *request message* (containing URL) into TCP connection socket

3. http server receives request messages, forms a *response message* containing requested object (index.html), sends message into socket

time

HTTP Example (cont.)

5. http client receives a response message containing html file, displays html, Parses html file, finds 10 referenced jpeg objects

4. http server closes the TCP connection (if necessary).

6. *Steps 1-5 are then repeated for <u>each</u> of 10 jpeg objects*

time

For fetching referenced objects, have 2 options:

- non-persistent connection: only one object fetched per TCP connection
  - some browsers create multiple TCP connections *simultaneously* - one per object
- persistent connection: multiple objects transferred within one TCP connection

Your Shell as a Web browser

1. Telnet to your favorite WWW server:

```
telnet www.google.com 80
```

Opens TCP connection to port 80 (default http server port) at www.google.com
Anything typed in sent to port 80 at www.google.com

2. Type in a GET http request:

```
GET /index.html
Or
GET /index.html HTTP/1.0
```

By typing this in (may need to hit return twice), you send this minimal (but complete) GET request to http server

3. Look at response message sent by http server!

What do you think the response is?

Does our Working Definition work for the http Web?

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and that communicate through an unreliable communication medium.

- Entity=a process on a device (PC, PDA)
- Communication Medium=Wired or wireless network
- Our interest in distributed systems involves
  - design and implementation, maintenance, study, algorithmics

"Important" Distributed Systems Issues

- No global clock; no single global notion of the correct time (asynchrony)
- Unpredictable failures of components: lack of response may be due to either failure of a network component, network path being down, or a computer crash (failure-prone, unreliable)
- Highly variable bandwidth: from 16Kbps (slow modems or Google Balloon) to Gbps (Internet2) to Tbps (in between DCs of same big company)
- Possibly large and variable latency: few ms to several seconds
- Large numbers of hosts: 2 to several million

Many Interesting Design Problems

-
-
- Real distributed systems
  - Cloud Computing, Peer to peer systems, Hadoop, key-value stores/NoSQL, distributed file systems, sensor networks, measurements, graph processing, stream processing, …
- Classical Problems
  - Failure detection, Asynchrony, Snapshots, Multicast, Consensus, Mutual Exclusion, Election, …
- Concurrency
  - RPCs, Concurrency Control, Replication Control, Paxos, …
- Security
  - ACLs, Capabilities, …
- Others…
-

Typical Distributed Systems Design Goals

- Common Goals:
  - Heterogeneity – can the system handle a large variety of types of PCs and devices?
  - Robustness – is the system resilient to host crashes and failures, and to the network dropping messages?
  - Availability – are data+services always there for clients?
  - Transparency – can the system hide its internal workings from the users? (warning: term means the opposite of what the name implies!)
  - Concurrency – can the server handle multiple clients simultaneously?
  - Efficiency – is the service fast enough? Does it utilize 100% of all resources?
  - Scalability – can it handle 100 million nodes without degrading service? (nodes=clients and/or servers) How about 6 B? More?
  - Security – can the system withstand hacker attacks?
  - Openness – is the system extensible?

“Important” Issues

- If you’re already complaining that the list of topics we’ve discussed so far has been perplexing…
  - You’re right!
  - It was meant to be (perplexing)

- The Goal for the Rest of the Course: see enough examples and learn enough concepts so these topics and issues will make sense
  - We will revisit many of these slides in the very last lecture of the course!

“Concepts”?

- Which of the following inventions do you think is the most important?
  1. Car
  2. Wheel
  3. Bicycle

“What lies beneath?” Concepts!

How will We Learn?

All this information is contained in handout on course website: "Course Information and Schedule"

- **Web: courses.engr.illinois.edu/cs425/**

- Textbook, Recommended but not Required
  - Colouris, Dollimore, Kindberg and Blair (5th edition)
  - CDK Textbook
  - If you use a different or older version, be sure to check problem numbers, etc.
  - Textbook is a great source of practice problems for the exams (midterm and final)

- Lectures

How will We Learn? (2)

All this information contained in handout on course website: "Course Information and Schedule"

- **Web: courses.engr.illinois.edu/cs425/**
- Homeworks (HWs)
  - Four in total, two before midterm (mid-Oct) and two after
  - About 2-3 weeks per homework
  - Solutions need to be typed, figures can be hand-drawn
- Programming assignments (MPs, i.e., Machine Programming Assignments)
  - **Only for 4 credit students** (3 credit students welcome to do it, but we won't be able to grade it or take it as extra credit)
  - You will be building a distributed stream processing system in stages
  - Four in total, two before midterm (mid-Oct) and two after
- Exams/quizzes
  - Midterm (mid-Oct) + Final

What assistance is available to you?

- Lectures
  - Lecture slides will be placed online at course website
    - "Tentative" version before lecture
    - "Final" version after lecture
- Asking Questions (most preferable first)
  1. **Piazza**: Your first stop for all questions (see if your question has already been asked)
     - We will try to ensure that every question is answered within 24 hours of posting (during weekdays).
     - Do not post solutions or code on Piazza – that'll immediately earn you a zero on the HW/MP.
  2. **Office Hours** (posted on course website): Every weekday has at least one office hours scheduled.
  3. **Email staff** mailing list (don't email individual instructor/TA except for private issues)
- Course Prerequisite: CS 241 or ECE 391 or equivalent OS/networking course (latter need instructor permission)

Course Staff (Again)

- Myself (*Indy*): 3112 Siebel Center
  - Office Hours Every Tuesday and Thursday right after class – see website
- TAs: (see course website for office hours)
  - *Faria Kalim (lead TA)*
  - *Rui Yang*
  - *Yao Qian*
  - *Ashwini Raina*
  - *Akshun Gupta*
  - *Jayasi Mehar (TA for MCS-Online and MCS-DS)*
  - *Mayank Bhatt (TA for MCS-Online and MCS-DS)*

Individual vs. Group Work

- Homeworks
  - Individual only
  - All work must be your own
- MPs
  - In groups of 2
  - Within group: Can discuss everything
- For both HWs and MPs
  - You can discuss with others (and course staff) lecture concepts and the HW/MP question/spec itself, but **you cannot discuss solutions or even ideas.**
  - You cannot copy solutions/code or look at someone else's solutions
    - We will check (we use Moss, we also compare HWs)
- First violation: zero on HW/MP. Second violation: F in course. (Both have happened in the past!)

Speaking of HWs and MPs

- HW1 and MP1 have both been released!
- Don't worry – you have time
- But start early! Start now.
- You must let us know the composition of your group for MP by this Thursday 9/7 5 pm.
  - If you don't meet this deadline you won't get VMs to do your MP!
  - Instructions on how to inform us are on the course website
- You can start on MP right away (don't need lectures for it)
- Due in about 2 weeks
  - MP1 due 9/17 (demos on 9/18)
  - HW1 due 9/26

3 cr vs 4 cr vs MCS-DS

	3 cr (On-campus and Online)	4 cr (On-campus and Online)	MCS-DS
HWs 1-4 On-campus	Y	Y	Y
MP On-campus	N	Y	N
Coursera Quizzes 5 + 5=10 Coursera Finals: Part 1 + Part 2	N	N	Y
Coursera Programming Projects Part 1 + Part 2	N	N	Y
On-campus Midterm Exam + On-campus Final Exam Coursera	Y	Y	Y

Table 1: Assignments for Sections in CS425/ECE428. Y = Yes. N = No.

Wrap-Up

- (Reading for today's lecture: Relevant parts of Chapter 1)
- All students:
  - Go to course website **https://courses.engr.illinois.edu/cs425/**
  - Todos:
    - (All) Sign up for Piazza by today
    - (4 cr) Form MP group by 9/7
    - (All) Fill out Student Survey sheet by today
    - View 6 lecture videos for "Intro to Clouds" (Clouds A – Clouds F) by 9/7
  - Not yet registered?
    - Fill out waitlist (posted now on course webpage)