COMP7940
Cloud Computing

Chapter 01

Characterization of Distributed Systems

Learning Outcomes

- Understand the concept of distributed systems
- Explain why we need distributed systems
- Give examples of distributed systems
- Understand the challenges in designing distributed systems

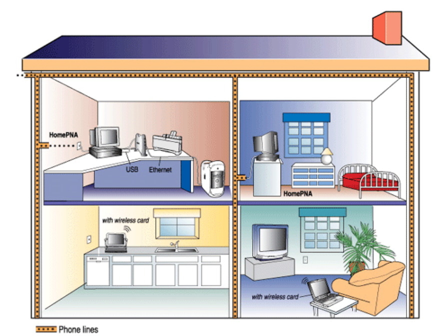
Outline

- Introduction
- Definition of Distributed Systems
- Example Distributed Systems
- Challenges of Distributed Systems
- Summary

Introduction

- Networks of computers are everywhere!
  - Global Internet
  - Mobile phone networks
  - Corporate networks
  - Factory networks
  - Campus networks
  - Home networks
  - In-car networks
  - On board networks in planes and trains

- Benefits of networks of computers: Resource sharing
  - Hardware resources
  - File, database, etc.
  - Information: text, image, audio, video



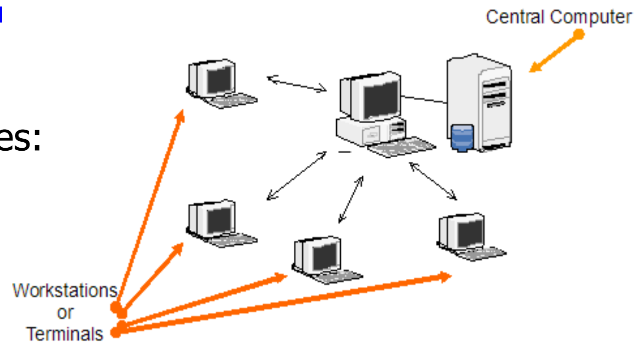
Defining Distributed Systems (DS)

- *"A system in which hardware or software components located at networked computers communicate and coordinate their actions only by message passing."* –– from REF1
- Examples of Distributed Systems:
  - Cluster:
    - A type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource
    - Science Faculty's cluster:
      - http://site.sci.hkbu.edu.hk/hpccc/
  - Cloud:
    - A type of parallel and distributed system consisting of a collection of interconnected and virtualised computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers
      - http://labs.hol.vmware.com/HOL/catalogs/catalog/681

Motivation of Distributed Systems

- To share resources and information. E.g.,
  - Staff in an office share a printer through a LAN.
  - A group of students share files through a NAS server.
  - A company shares its information with the public through its Web server and Internet.
  - Tens of millions of users share storage resources at Google through Google Drive.
  - Billions of users share personal information through Facebook.

Reasons for Distributed Systems

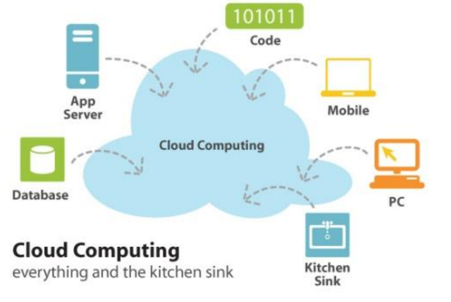


- Functional Separation:
  - Existence of computers with different capabilities and purposes:
    - Clients and Servers
    - Data collection and data processing
- Inherent distribution:
  - Information:
    - Different information is created and maintained by different people (e.g., Web pages)
  - People
    - Computer supported collaborative work (virtual teams, engineering, virtual surgery)
  - Retail store and inventory systems for supermarket chains
- Power imbalance and load variation:
  - Distribute computational load among different computers.
- Reliability:
  - Long term preservation and data backup (replication) at different locations.
- Economies:
  - Sharing a printer by many users and reduce the cost of ownership.
  - Building a supercomputer out of a network of computers.

Examples of Distributed Systems

- They (DS) are based on familiar and widely used computer networks:
  - —Internet
  - —Intranets, and
  - —Wireless networks

- Example DS:
  - —Web (and many of its applications like Facebook)
  - —Data Centers and Clouds
  - —Wide area storage systems
  - —Banking Systems



Example 1: Web Search

- Web search engines such as google.com and baidu.com are critical to our work, study, and life.
- Web search is a very complex task
  - To index the entire contents of the World Wide Web, including web pages, multimedia sources, scanned books, etc.
  - Over 63 billion pages and one trillion unique web addresses (2011)
- A complicated distributed system is designed to provide web search service
  - A very large number of networked computers located at many data centres
  - A distributed file system to support very large data storage and processing
  - An associated structured distributed storage system for fast access to very large datasets
  - A lock service that offers distributed system functions such as distributed locking and agreement
  - A programming model for managing very large parallel and distributed computations

Example 2: Massively Multiplayer Online Games (MMOGs)

- MMOGs have the following technical challenges
  - Fast response time for good user experience
  - Real-time propagation of events to many players
  - How to maintain a consistent view of the shared world?
- Possible solutions:
  - Client-server architecture: a single copy of the state of the game world is maintained by a cluster of computers
    - The load is partitioned by allocating individual "star systems" to particular computers within the cluster.
    - Ensure fast user response through optimizing network protocols and rapid response to incoming events.
  - Distributed architecture: the universe is partitioned across many servers which may also be geographically distributed
    - Users are dynamically allocated a particular server based on current usage patterns and network delays to the server.

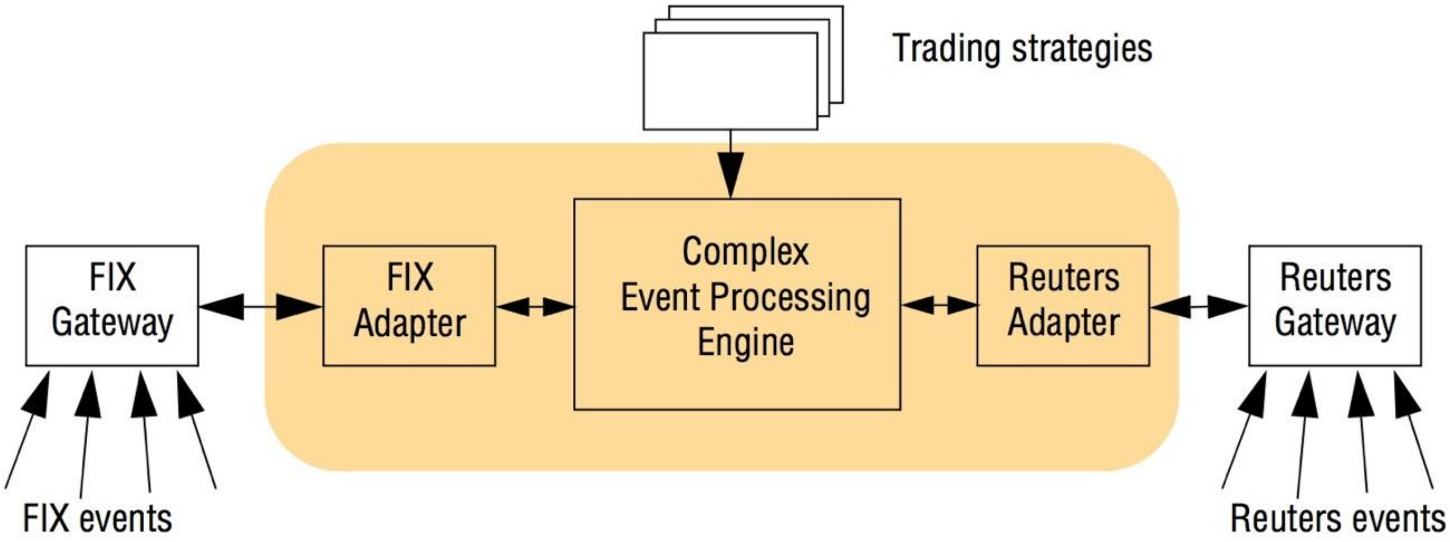
Example 3: Financial Trading

- Financial industry needs cutting edge distributed systems for real-time access to a wide range of information such as share prices and trends, and economic & political events.

- Distributed event-based systems are used to make intelligent trading strategies
  - Many suppliers provide different types of financial events (share price movements, interest rate changes, unemployment rate changes, etc.)
  - Many users subscribe to the suppliers to receive the events reliably in a timely manner

Example 3 (Cont.)

REF1 Figure 1.2. An example of financial trading system



- Heterogeneity: event sources are in a variety of formats
  - Example events: Reuter market data events, FIX events, etc.
  - FIX: Financial Information eXchange protocol
- Adapters are used to translate heterogeneous formats into a common internal format.
- The event processing engine needs to handle the incoming event streams in real-time, e.g., looking for patterns that indicate a trading opportunity (e.g., HSBC's stock prices in HK and London are slightly different due to exchange rate).

Example 3 (Cont.)

- Sample trading script (from www.progress.com):

```
WHEN
    MSFT price moves outside 2% of MSFT Moving Average
FOLLOWED-BY (
    MyBasket moves up by 0.5%
    AND
        HPQ's price moves up by 5%
        OR
        MSFT's price moves down by 2%
)
ALL WITHIN
    any 2 minute time period
THEN
    BUY MSFT
    SELL HPQ
```

MSFT: Microsoft
HPQ: Hewlett Packard

Consequences of Distributed Systems

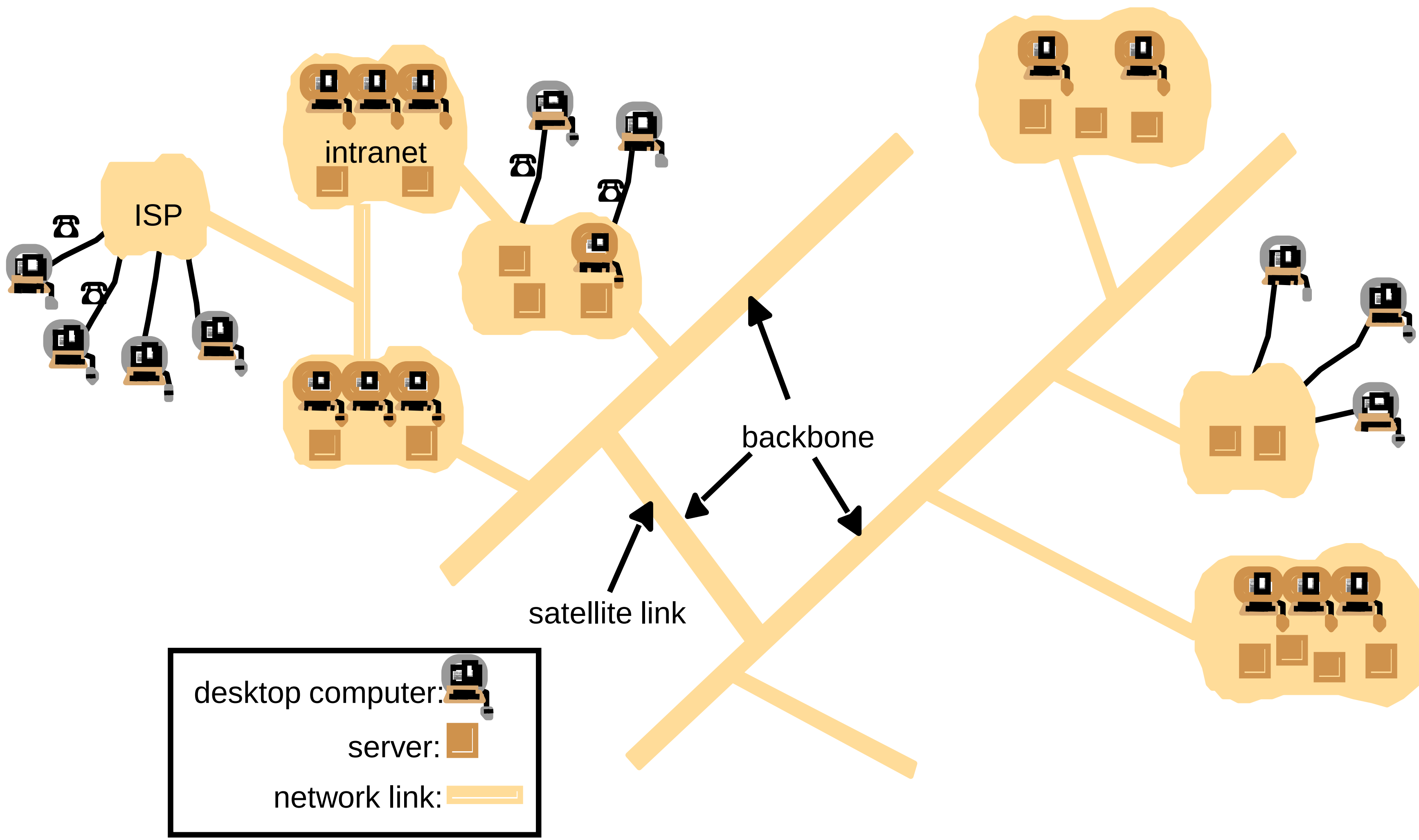
- Computers in distributed systems may be on separate continents, in the same building, or the same room. DSs have the following consequences:
  - Concurrency – each computer is autonomous.
    - Carry out tasks independently
    - Tasks coordinate their actions by exchanging messages.
    - System capacity can be increased by adding more resources.
  - No global clock
    - The clocks in different computers are different. The accuracy of clock synchronization is limited.
  - Independent Failures
    - Each component (network, computer, etc.) of a distributed system can fail independently,
      - but the other components do not know about the crash.

Trends in Distributed Systems

- Distributed systems are undergoing significant change with the following influential trends
  - The emergence of pervasive networking technology
  - The emergence of mobile and ubiquitous computing
  - The increasing demand for multimedia services
  - The view of distributed systems as a utility

Pervasive Networking and the modern Internet

REF Fig. 1.3 A typical portion of the Internet



Pervasive Networking and the modern Internet (cont.)

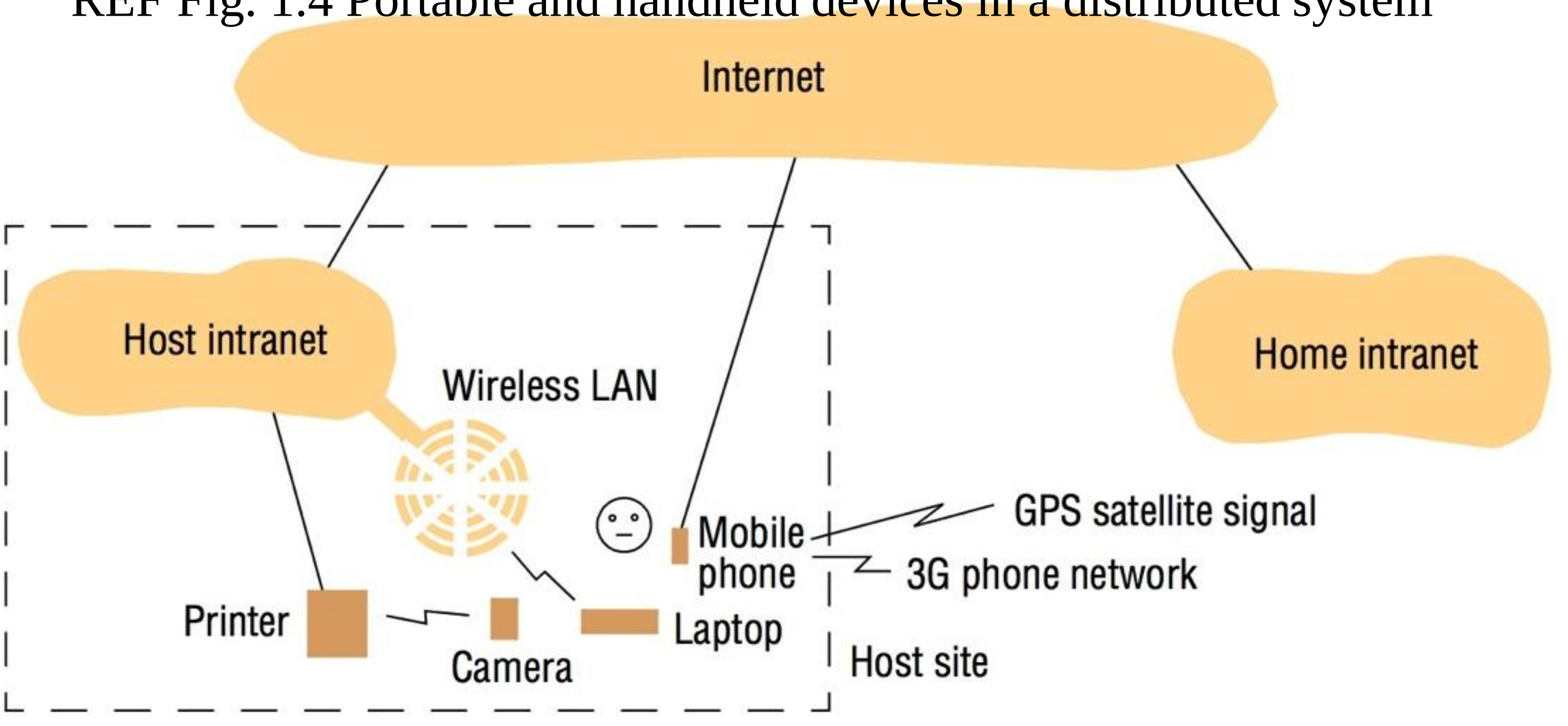
- Pervasive: spread throughout
- The Internet is a vast collection of computer networks of many different types and hosts various types of services.
  - Wireless: WiFi, WiMAX, Bluetooth, 4G or 5G, satellite, etc.
  - Wired: electronic, optical, tradition phone network, etc.
- Maintenance of intranet often an issue in distributed systems
  - No risk if no connection to internet, e.g., some intranets of police or security or law enforcement agencies are not connected to the internet.
  - Firewalls are used to limit services from/to an Intranet, e.g., no FTP or Remote Desktop access is allowed to intranet.
    - Firewalls are problematic by impeding legitimate access to services when resource sharing between internal and external users is required.

Mobile and Ubiquitous Computing

- Mobile computing: performing computing tasks while the user is on the move, away from his/her usual environment
  - Mobile devices: mobile phones, laptops, cameras, watches, etc.
  - Mobile devices can still access resources in home intranet (e.g., looking up a photo in your home hard disk via HKBU WiFi).

Mobile and Ubiquitous Computing (cont.)

REF Fig. 1.4 Portable and handheld devices in a distributed system



- A mobile phone user (e.g., student) using a host intranet (e.g., on HKBU campus).
- He can still connect to his home intranet.

Mobile and Ubiquitous Computing (cont.)

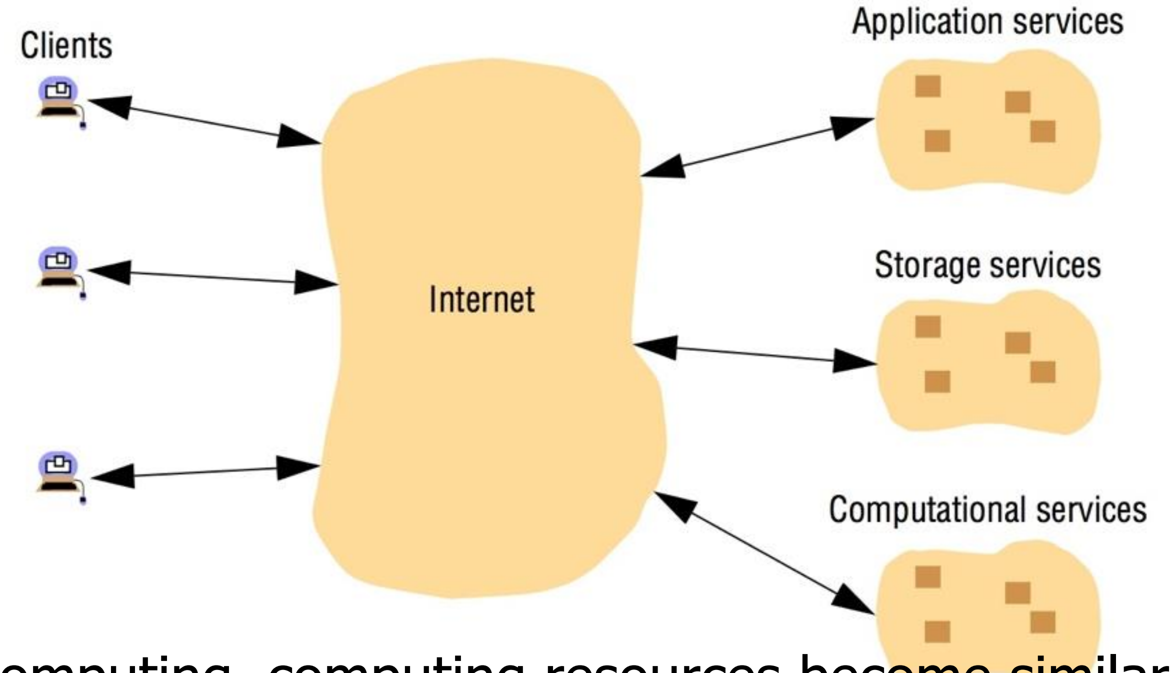
- Ubiquitous computing is the harnessing of many small, cheap computational devices in users' physical environments, including home, office and natural settings.
  - Printers, TVs, washing machines, Hi-Fi systems, cars, refrigerators, door locks, etc.
  - Needs to access resources conveniently located nearby (location-aware or context-aware computing).
- Ubiquitous and mobile computing may overlap.
  - E.g., At HKBU, you use your mobile phone (mobile computing) to print a pdf file in your phone using a lab printer (ubiquitous computing).
- Needs to support spontaneous interoperation.
  - Associations between devices are routinely created and destroyed.
  - Automatic service discovery

Distributed Multimedia Systems

- Needs to support the storage, transmission, and presentation of discrete media types, e.g., text, pictures, video, etc.
  - For each type, support different encoding and decoding formats (e.g., .mov, .wmv, .avi etc.)
- To broadcast across a group of users.
- Must preserve real-time relationship between elements of a media type.
  - Preserve frame sequence
  - Maintain minimum speed (# of frames per sec)
  - Maximum delay or latency
  - Quality of service (QoS)

Distributed Computing as a Utility

REF Fig. 1.5 Cloud Computing



- In cloud computing, computing resources become similar to other utilities such as water or electricity.
  - Resources: hard disk space, RAM, CPU, virtual network, virtual machine, virtual OS (key enabling technology of cloud), virtual firewall, software such as gmail, Google Map/Earth, etc.
- Cloud service providers: Amazon, Google, Alibaba, Microsoft, IBM, etc.
  - Big business

Challenges of Distributed Systems: An Example

- Suppose you own a web-based online store (such as amazon.com)
  - Customers can connect their computer to the web server:
    - Browse product catalog
    - Search and compare
    - Place orders
    - …

List of Challenges

- Heterogeneity
  - Heterogeneous components must be able to interoperate
- Openness
  - Interfaces should be publicly available to ease inclusion of new components
- Security
  - The system should only be used in the way intended
- Scalability
  - System should work efficiently with an increasing number of users
  - System performance should increase with inclusion of additional resources

List of Challenges (Cont.)

- Fault handling
  - Detecting failures/Masking failures/Tolerating failures/Recovery from failures/Redundancy
- Concurrency
  - Shared access to resources must be possible
- Distribution transparency
  - Distribution should be hidden from the user as much as possible

Challenges I

- What if
  - Your company connects to the internet using optical fibers, but a customer connects to your website from an old-fashioned telephone network in a 3rd world country?
  - Your customer uses a completely different hardware?
    - PC, MAC, mobile phone…
    - Mobile code: program code that can be transferred from one computer to another and run at the destination, e.g., Java applets running on Java virtual machines.

Challenges I (con't)

- Your customer uses a completely different operating system?
  - Windows, Unix, Android, IOS…
  - All have implementations of the Internet protocols, but
  - interfaces (function calls) for exchanging messages are different in different OS.
- Your customer uses a different way of representing data?
  - ASCII, EBCDIC, …
  - big endian, little endian integers
- **Heterogeneity**

Challenges II

- When building the system…
  - Do you want to write the whole software on your own (network, database,…)?
  - What about updates, new technologies?
  - **Reuse** and **Openness** (Standards)
    - The openness of distributed systems is determined by the degree to which new resource-sharing services can be added and made available for use by a variety of client programs.
    - Cannot be achieved unless the specification and documentation of the key software interfaces of the components are made available to software developers (<u>published</u>).
      - E.g., file transfer, email, telnet, etc.
    - Open distributed system can be constructed from heterogeneous hardware and software. The conformance of each component to the published standard must be <u>carefully tested</u>.

Challenges III

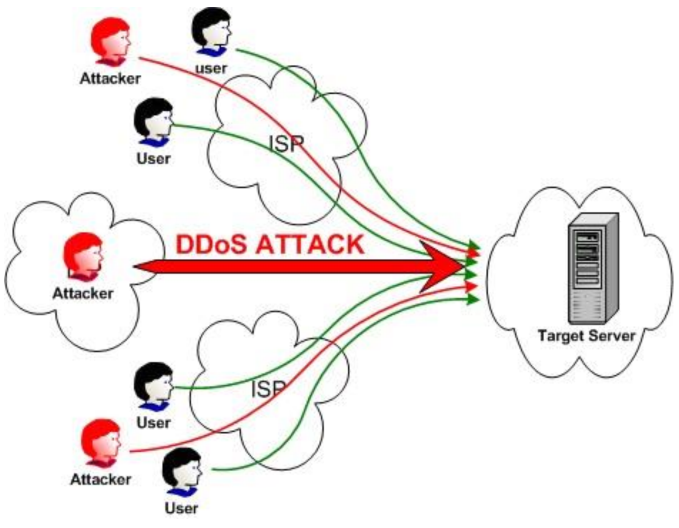
- What if
  - Someone tries to break into your system to steal data?
  - ... sniffs for information?
  - ... your customer orders something and does not accept the delivery saying he did not?
  - **Security**
    - Confidentiality, integrity, authenticity, availability

Security I

- Resources are accessible to authorized users and used in the way they are intended
- Confidentiality
  - Protection against disclosure to unauthorized individual information
  - E.g. ACLs (access control lists) to provide authorized access to information
- Integrity
  - Protection against alteration or corruption
  - E.g. changing the account number or amount value in a money order

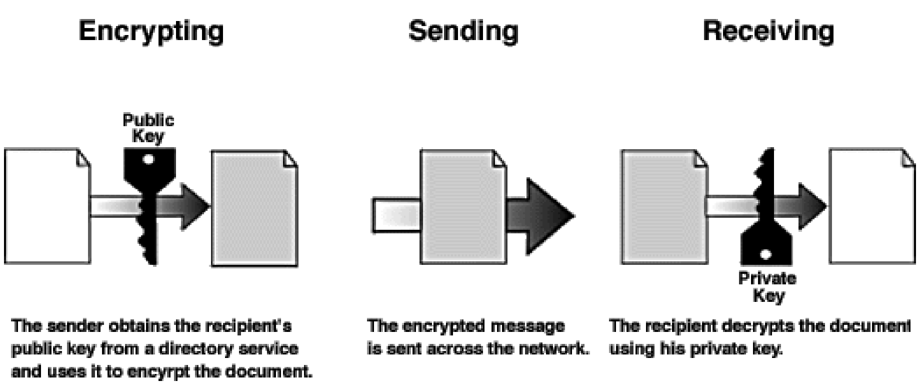
Security II

- Availability
  - Protection against interference targeting access to the resources.
  - E.g. denial of service (DoS, DDoS) attacks
- Authenticity or Non-repudiation
  - Proof of sending / receiving an information
  - E.g. digital signature



Security Mechanisms

- Encryption
  - E.g. Blowfish, AES, RSA
- Authentication
  - E.g. password, biometrics, public key authentication
- Authorization
  - E.g. access control lists

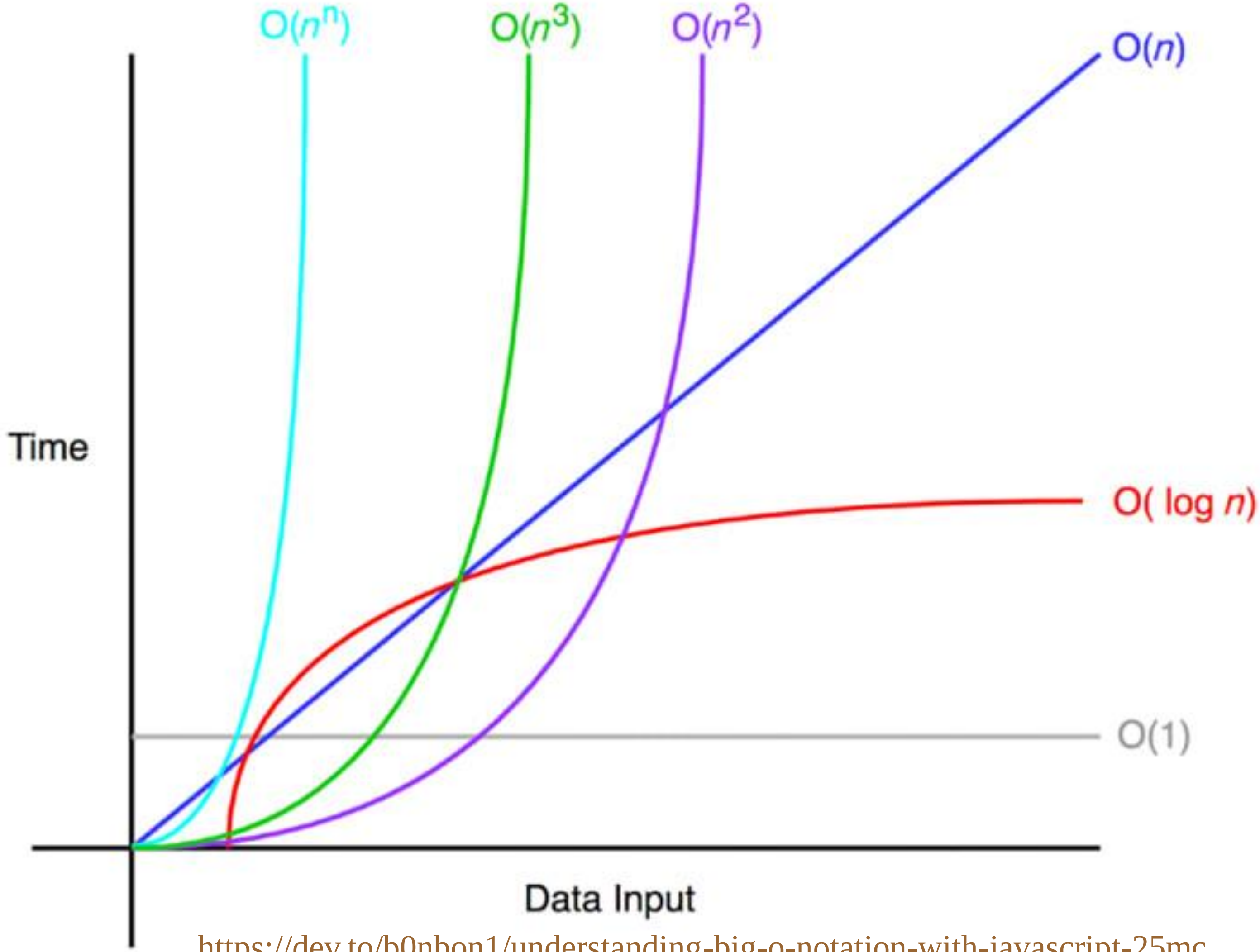


Challenges IV

- What if
  - You are so successful that millions of people are visiting your online store at the same time?
  - **Scalability**
    - A system is scalable if it remains effective when there is a significant increase in the number of resources and the number of users.
    - Controlling the cost of physical resources
      - For a system with n users to be scalable, the quantity of physical resources required should be at most $O(n)$.
    - Controlling the performance loss
      - Consider the management of a set of data whose size is proportional to the number of users or resources.

Big O notation

Usually Big O is used to measure time-complexity, but is also possible to measure other resource



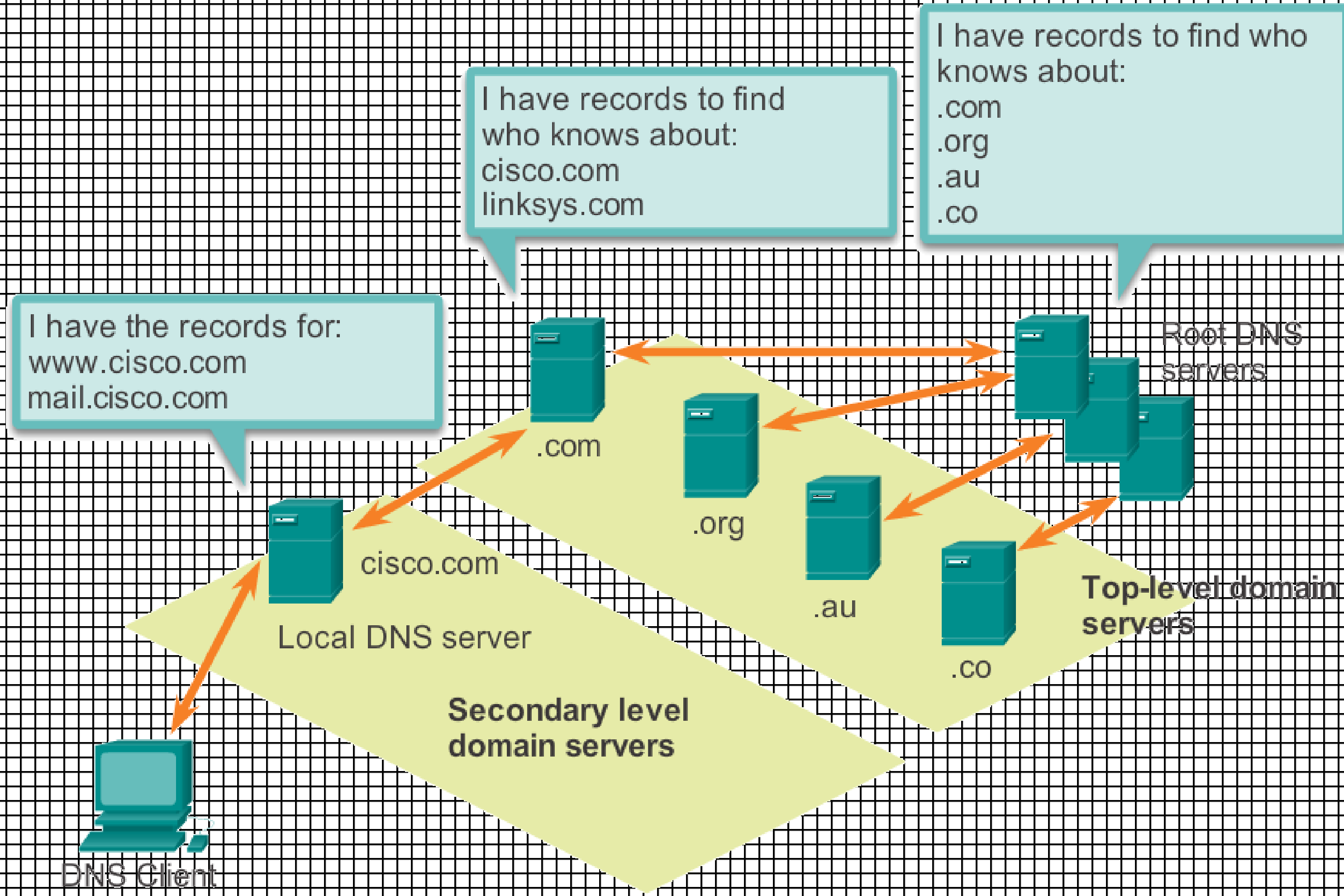
https://dev.to/b0nbon1/understanding-big-o-notation-with-javascript-25mc

Challenges IV

—Scalability (cont.)

- Preventing software resources running out
  - E.g., IP addresses (IPv4) using 32 bits are running out.
  - Now migrating to IPv6 using 128 bits – not easy task, a lot of software code must be changed.
- Avoiding performance bottlenecks
  - Algorithms should be decentralized.
  - Example
    - The predecessor of the DNS keeps the name table in a single master file that could be downloaded to any computer – a performance bottleneck.
    - The DNS removed this bottleneck by partitioning the name table between servers located throughout the Internet.

DNS Server



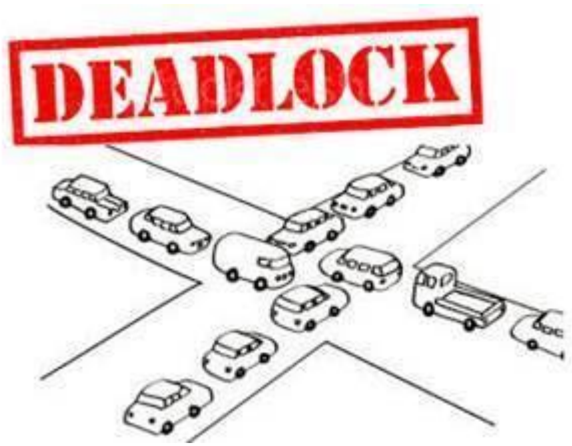
Ref: http://teachweb.milin.cc/datacommunicatie/tcp_osi_model/application_layer/dns.htm

Challenges V

- What if
  - Two customers want to order the same item at the same time, or to bid on the same item in eBay?
  - **Concurrency**
  - avoid
    - Racing Condition
    - deadlock

Thread A		Thread B		
Instruction	Register	Instruction	Register	Count
		LOAD Count	10	10
LOAD Count	10			10
ADD #1	11			10
STORE Count	11			11
		SUB #1	9	11
		STORE Count	9	9

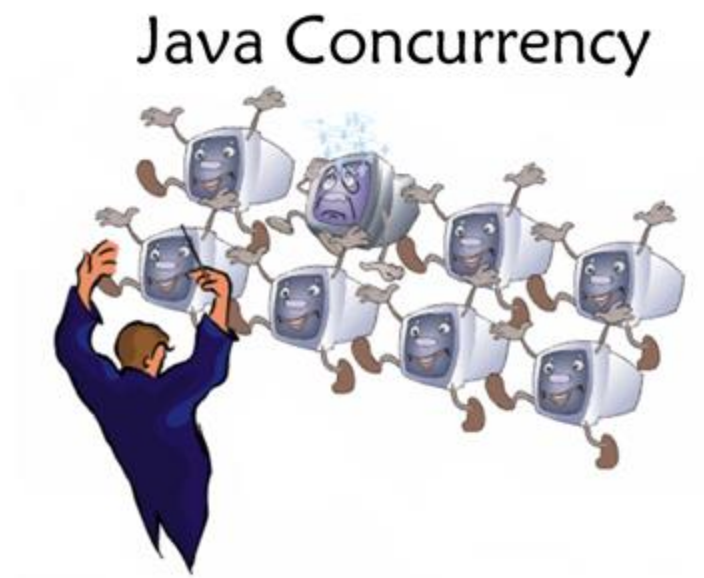
https://pages.mtu.edu/~shene/NSF-3/e-Book/RACE/overview.html



https://robertostefanettinavblog.com/2018/04/11/detect-deadlocks-in-old-navs-2017/

Concurrency

- Provide and manage concurrent access to shared resources:
  - Fair scheduling
  - Preserve dependencies (e.g. distributed transactions)
  - Avoid deadlocks
  - Object locking, data consistency, semaphores

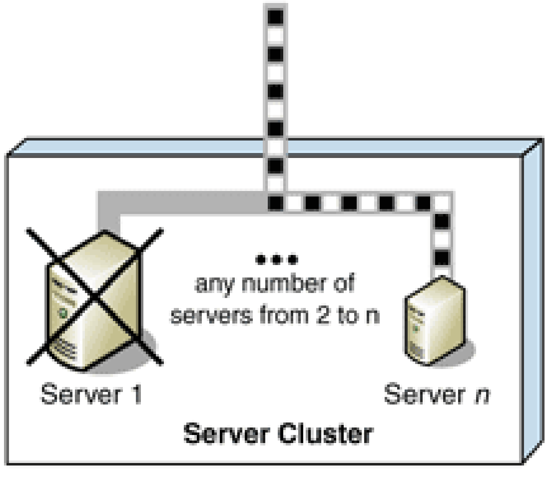


Challenges VI

- What if
  - The server or database with your inventory information crashes?
  - Your customer's computer crashes in the middle of an order?
  - The order is lost or corrupted during transmission?
  - **Fault tolerance**

Fault Tolerance

- Failure: an offered service no longer complies with its specification
- Fault: cause of a failure (e.g. crash of a component)
- Fault tolerance: no failure despite faults



Fault Tolerance Mechanisms

- Fault detection
  - Checksums, heartbeat (originator sends messages periodically), …
- Fault masking
  - Retransmission of corrupted messages, redundancy, …
  - Redundancy in
    - hardware, routing path, Domain Name System, databases, etc.
- Fault toleration
  - Exception handling, timeouts, …
- Fault recovery
  - Rollback mechanisms, …

Challenges VII

- What if
  - You want to move your business and servers to a different city (because of the weather)?
  - **Distribution or mobility transparency**
  - Transparency is defined as the concealment from the user and application programmer of the separation of components in a distributed system.
- Eight forms of transparency (as identified by Advanced Networked Systems Architecture ANSA, an industry standard)
  - Access transparency enables local and remote resources to be accessed using identical operations.
  - Location transparency enables resources to be accessed without knowledge of their physical or network location.

Challenges VI

- Eight forms of transparency (cont.)
  - Concurrency transparency enables several processes to operate concurrently using shared resources without interference between them.
  - Replication transparency enables multiple instances of resources to be used to increase reliability and performance without knowledge of the replicas by users or application programmers.
  - Failure transparency enables the concealment of faults.
  - Mobility transparency allows the movement of resources and clients within a system without affecting the operation of users or programs.
  - Performance transparency allows the system to be reconfigured to improve performance as loads vary.
  - Scaling transparency allows the system and applications to expand in scale without change to the system structure or the application algorithms.

Summary

- Distributed Systems are everywhere
- The Internet enables users throughout the world to access its services wherever they are located
- Resource sharing is the main motivating factor for constructing distributed systems
- Construction of DS produces many challenges:
  - Heterogeneity, Openness, Security, Scalability, Failure handling, Concurrency, and Transparency

Sample questions

- Give examples of heterogeneous computing devices in DS.
- Give a characteristic of open system.
- Give an example of security requirement in DS.
- Why scalability has been a challenge to DS?
- Explain a mechanism to handle failures in DS.
- What are challenges in concurrent computation in DS.
- Explain X transparency.