Operating Systems and Networks

Pierre Kleberger
pierre.kleberger@chalmers.se
Computer Science & Engineering

Adaption of slides by Andreas Larsson and Anders Gidenstam
With selected slides from:
• Kurose & Ross, ”Computer Networking”

Roadmap

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

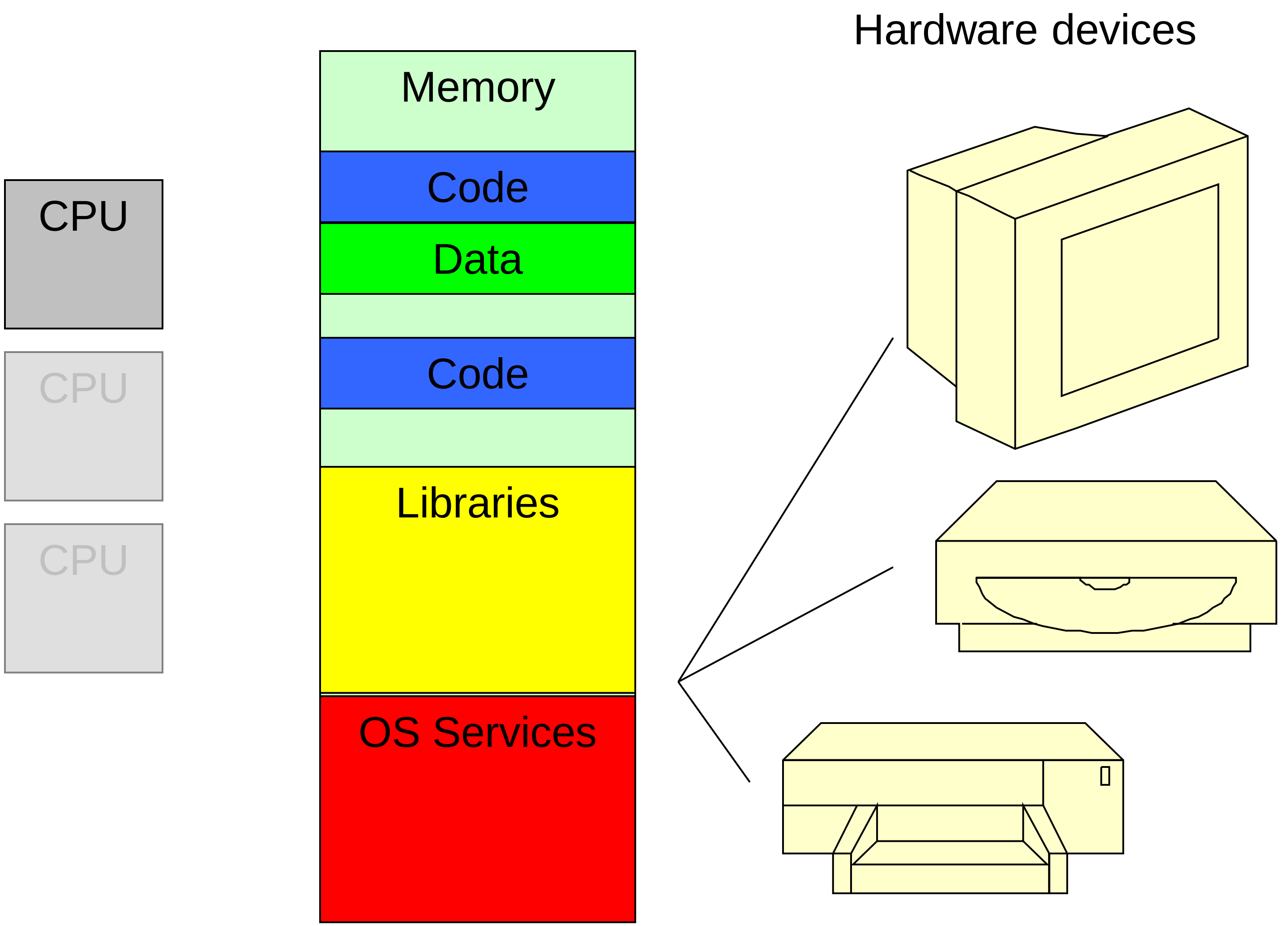
What is an Operating System?

- Intermediary between the user and the hardware.
- Controls the execution of application programs
- Is an interface between applications and hardware

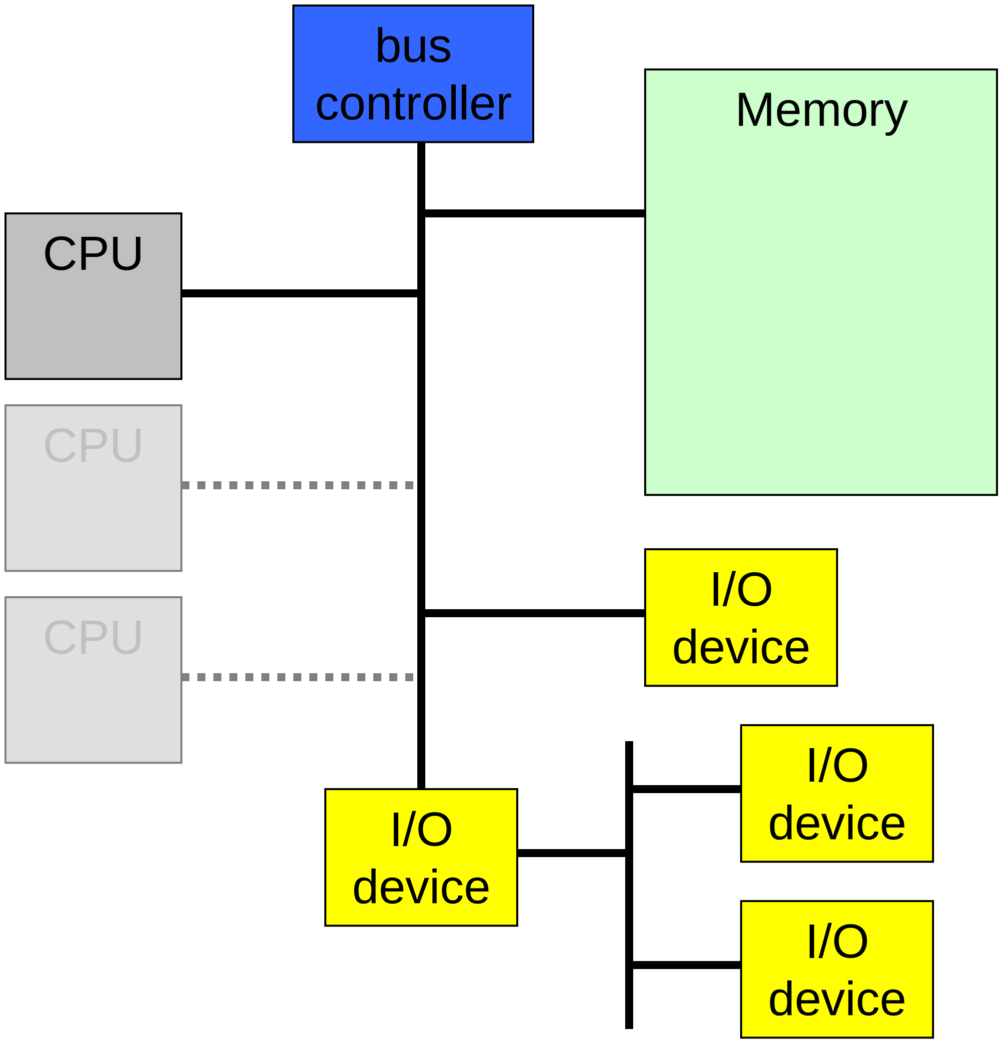
- Operating system goals:
  - Execute user programs
  - Facilitate problem solving for the users
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

The Computer:
End-user's view

The Computer: Application programmer's view

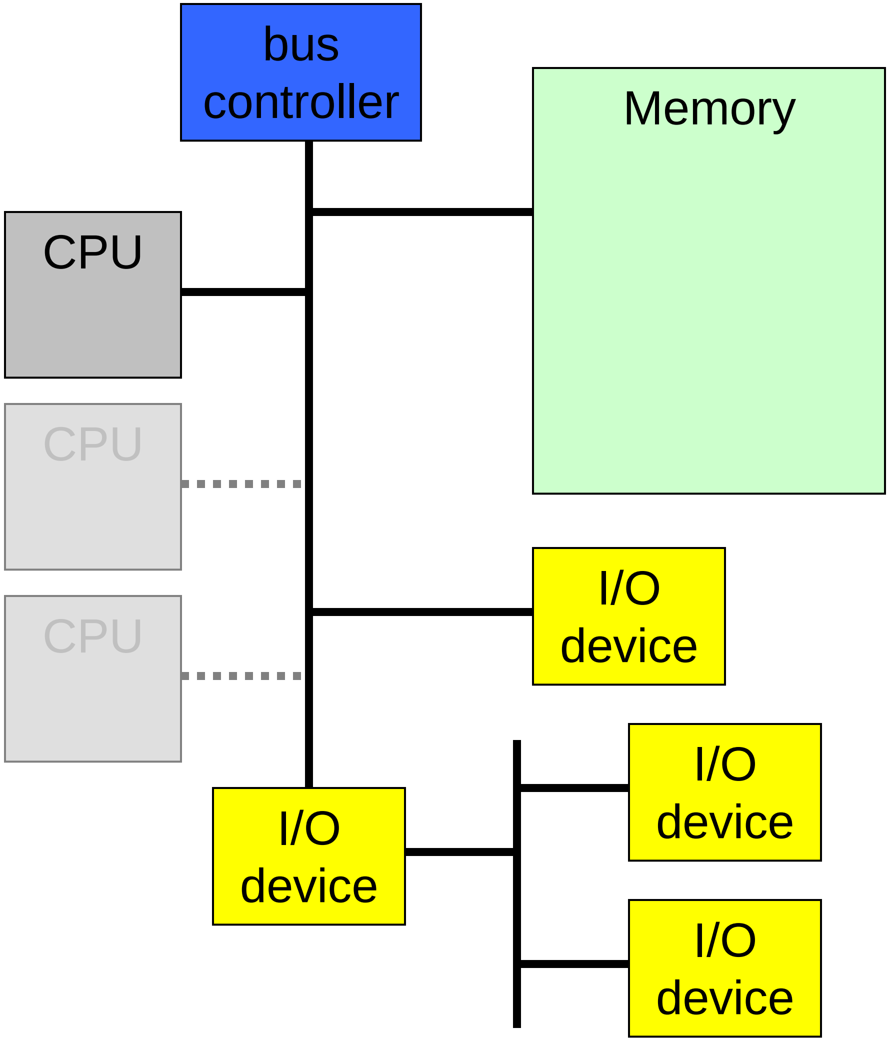


The Computer:
OS programmer's view



Computer Hardware

- Processors
- Main Memory
  - Primary (”real”) memory
  - Volatile
- I/O devices
  - secondary memory devices
  - communications devices
    - Screen, keyboard, network
- System bus
  - communication among processors, memory, and I/O modules

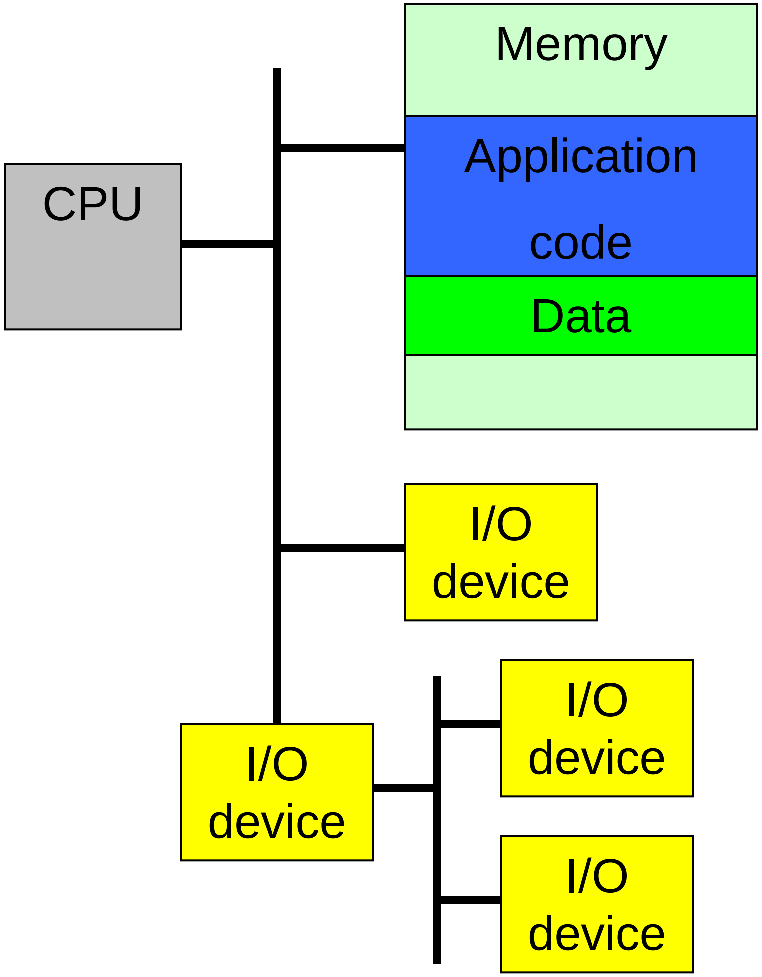


Introduction

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

The evolution of operating systems

- The beginning
  - No OS
  - Every application had to do everything by itself
  - One program at a time

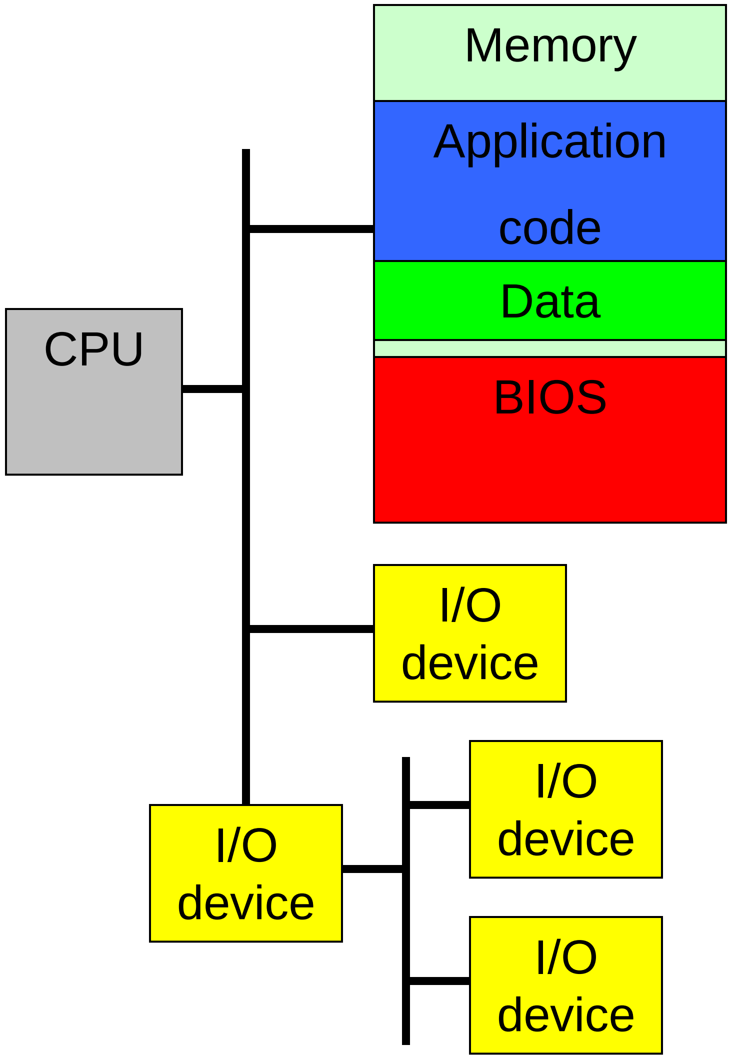


- Surely, still not so today?
  - microcontrollers

The evolution of operating systems

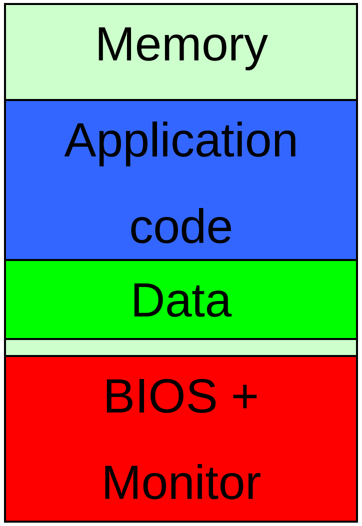
- BIOS
  - Basic Input Output System
  - In Read Only Memory (ROM)
  - Provides interface routines for accessing the hardware

- Still, only one program at a time



Batch processing

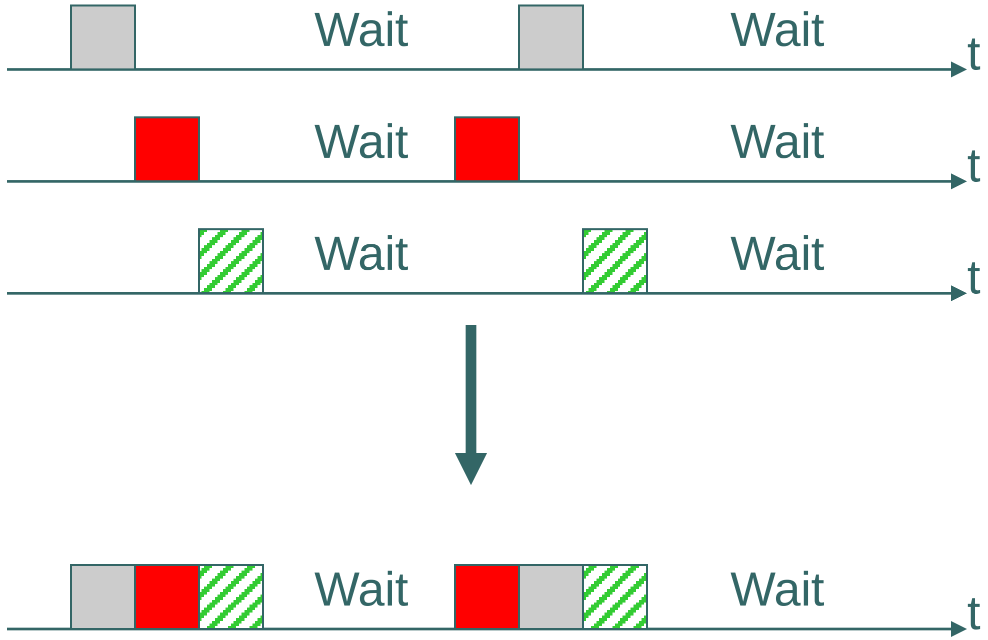
- In the 50s computers were expensive and rare, so efficient utilization was important
- Simple Batch Systems
  - Queue of jobs, run one at the time
  - Monitor
    - Software that controls the running programs
    - Batch jobs together
    - Program branches back to monitor when finished
    - Resident monitor is in main memory and available for execution



Uni- and Multiprogramming (1)
Uniprogramming

- One single program is running
- Processor must wait for I/O operations to complete before proceeding
- Leads to poor processor utilization

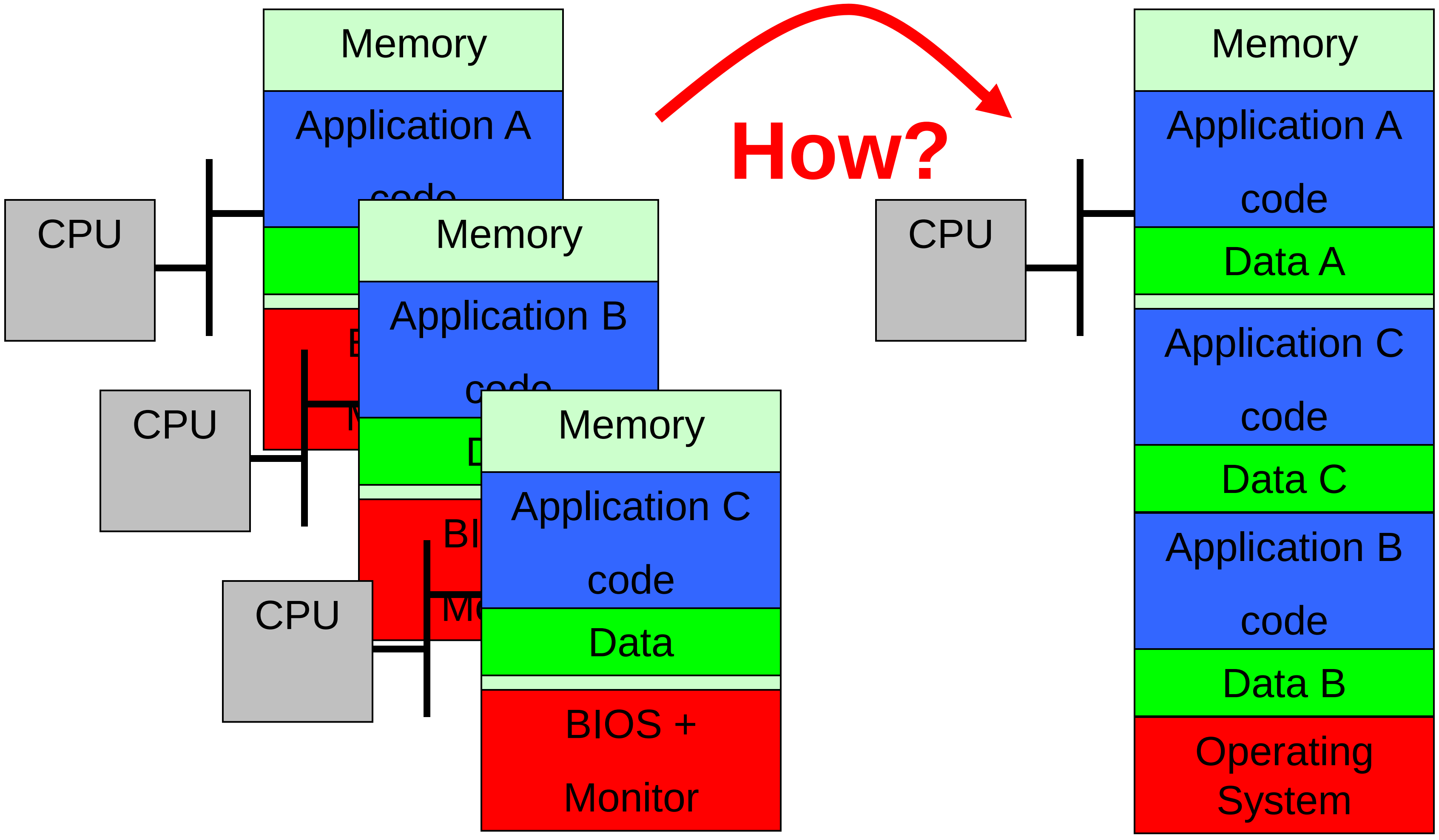
Uni- and Multiprogramming (2)
Multiprogramming



Uni- and Multiprogramming (3) Multiprogramming

- Multiprogramming
  - Switch jobs at regular intervals
  - Benefits
    - Many applications running at the same time
    - Allows many simultaneous users
    - Interactive programs
    - “Real-time” interaction with user
    - Parallel/concurrent applications
  - Next step
    - Multiprocessor computers

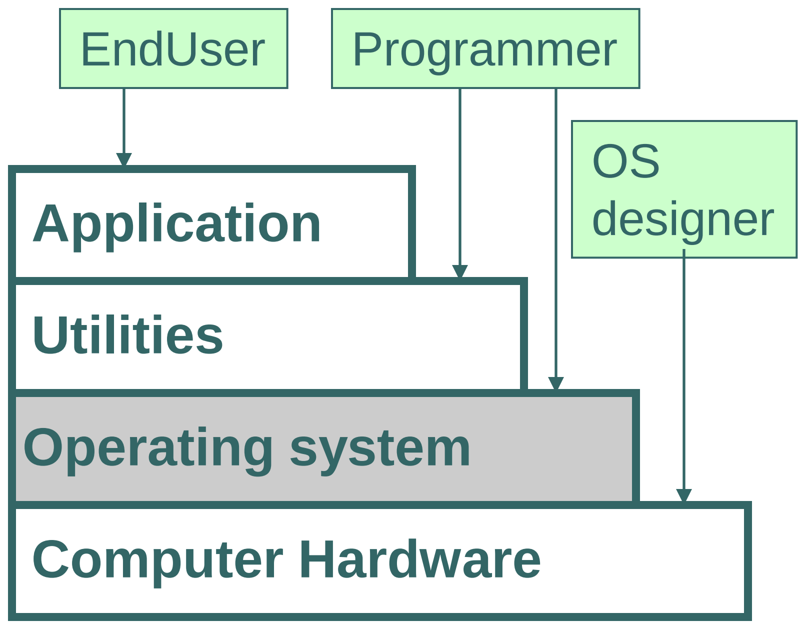
Multiprogramming – The Challenge



Operating System Architecture

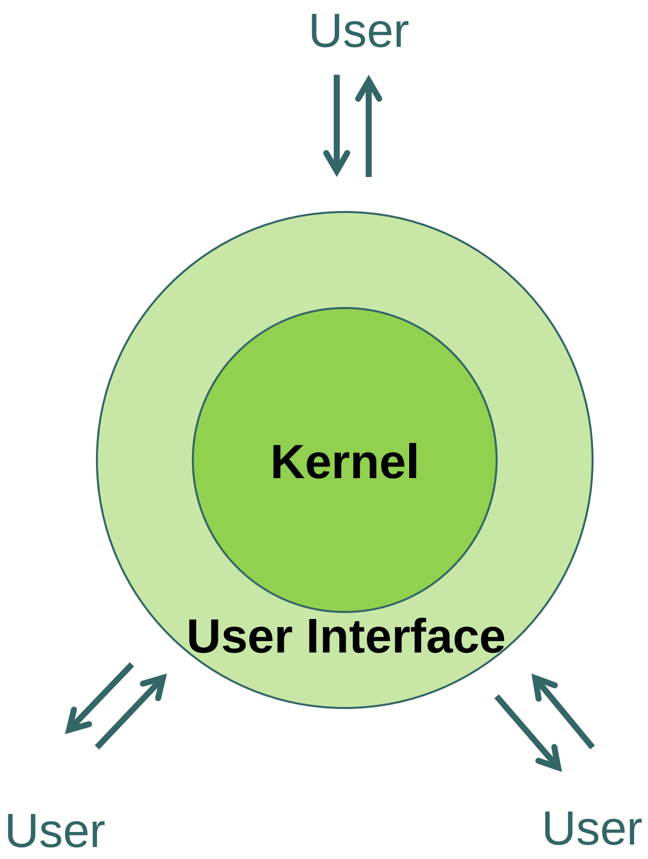
Software in the system

- Applications
- System software
  - Utilities
    - Compilers
    - Interpreters
  - Operating System
    - Shell
      - GUI
      - Command line
    - Kernel
      - The core of the OS



Services provided by the OS

- Program execution
- Access to I/O devices
- Controlled access to files
- Error detection
  - Hardware errors
  - Sofware errors
- Program development

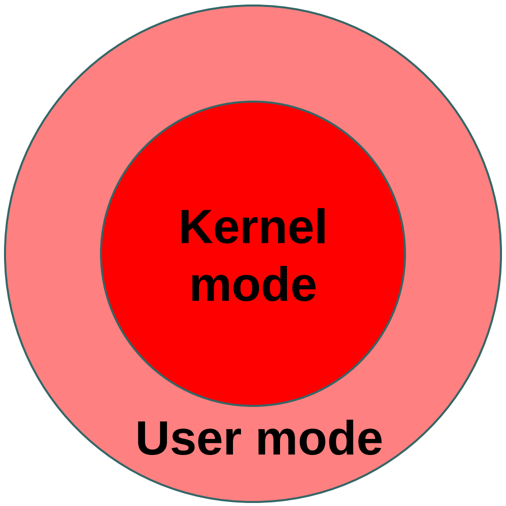


Kernel overview

- Portion of operating system that is in main memory
  - Contains most-frequently used functions
- Resource control
  - CPU Scheduling
  - Memory manager
  - File manager
  - Device drivers
- Bootstrap
  - Get the operating system running at system start

Kernel Security

- Privileged mode (Kernel mode)
  - Allowed to execute all CPU instructions
  - Access to all I/O devices
- Unprivileged mode (User mode)
  - Only a limited number of CPU instructions can be executed
    - e.g. access to memory and I/O devices are restricted

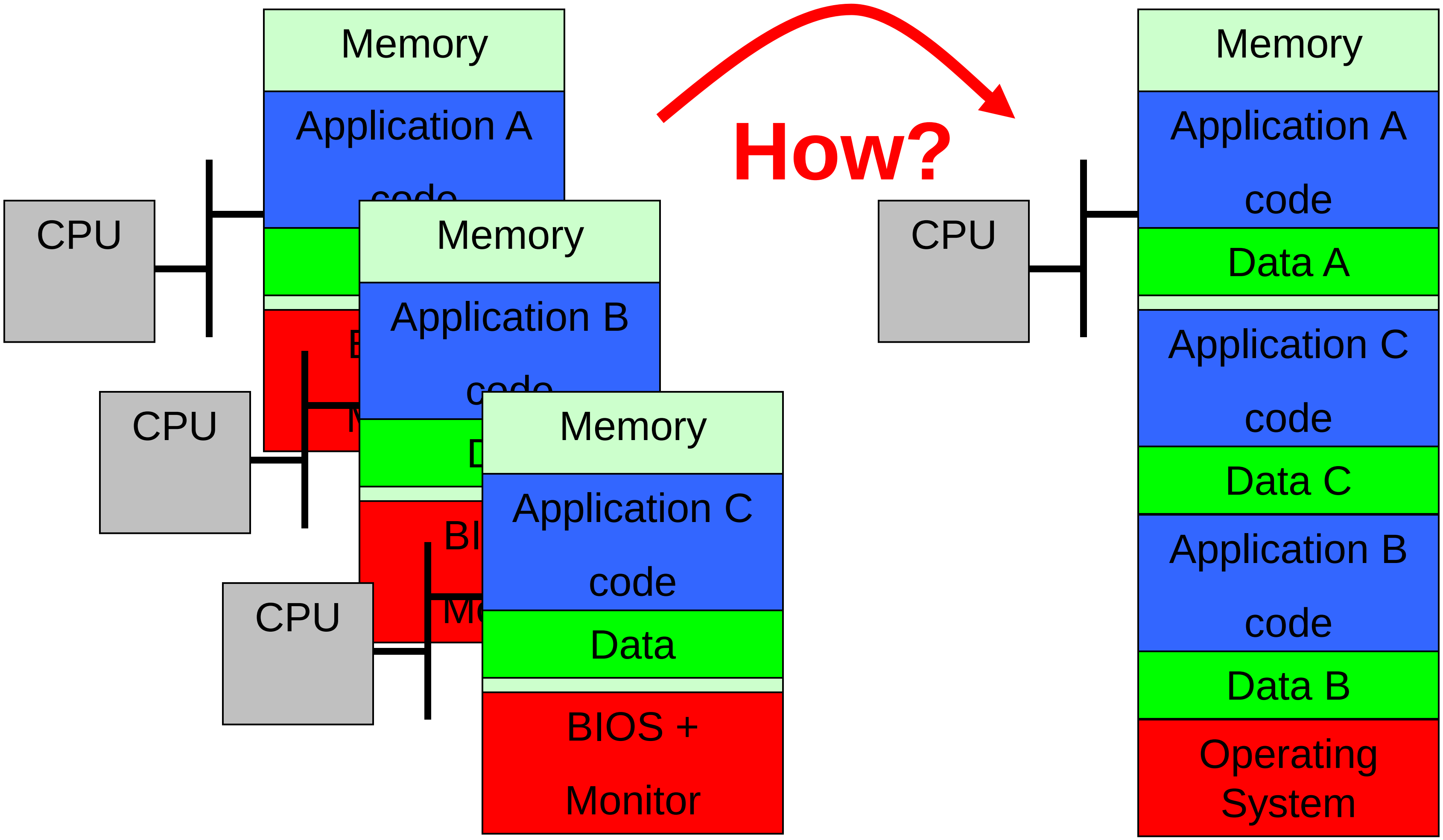


Roadmap

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

Do you remember?

Multiprogramming – The Challenge

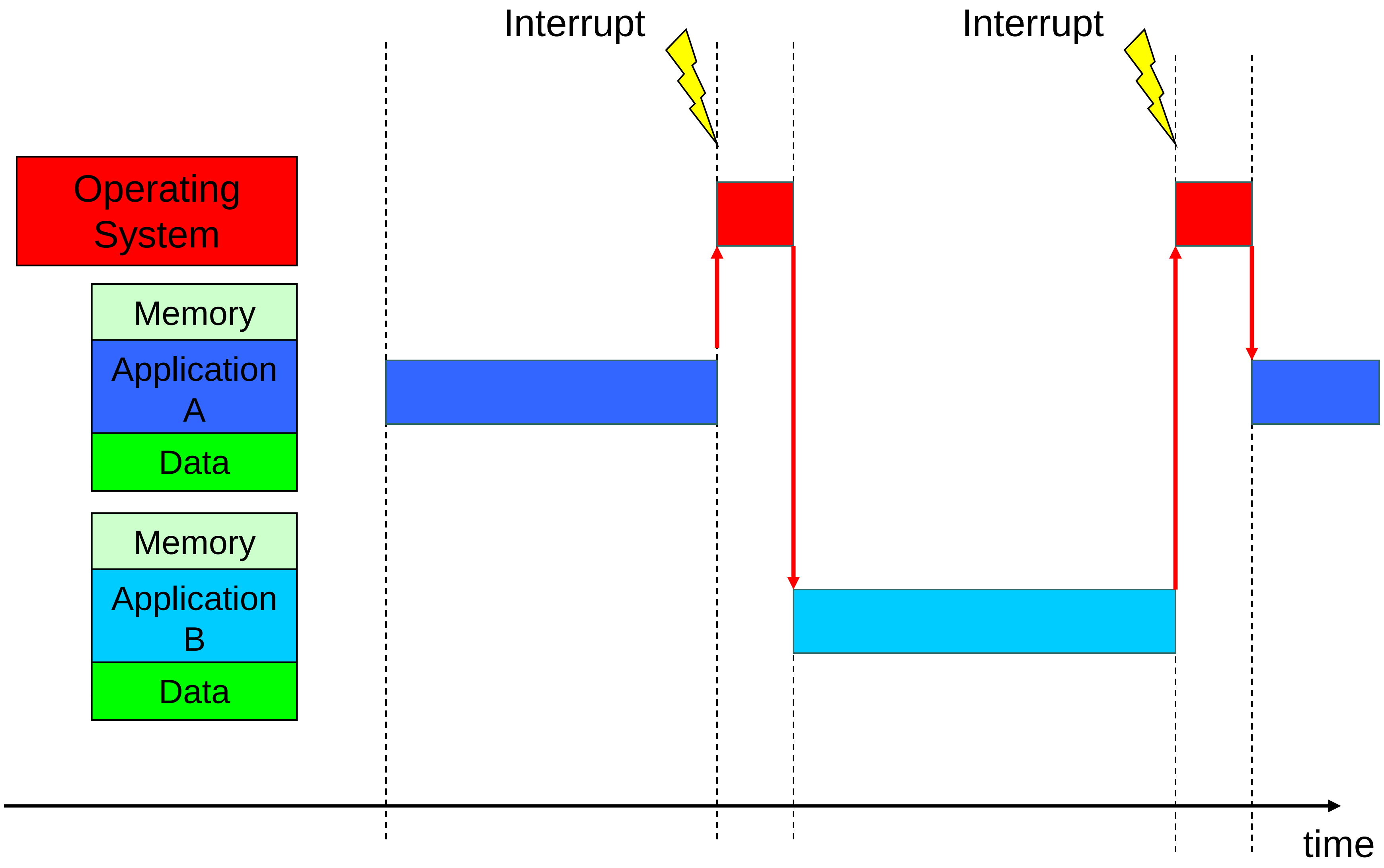


The answer:
Processes

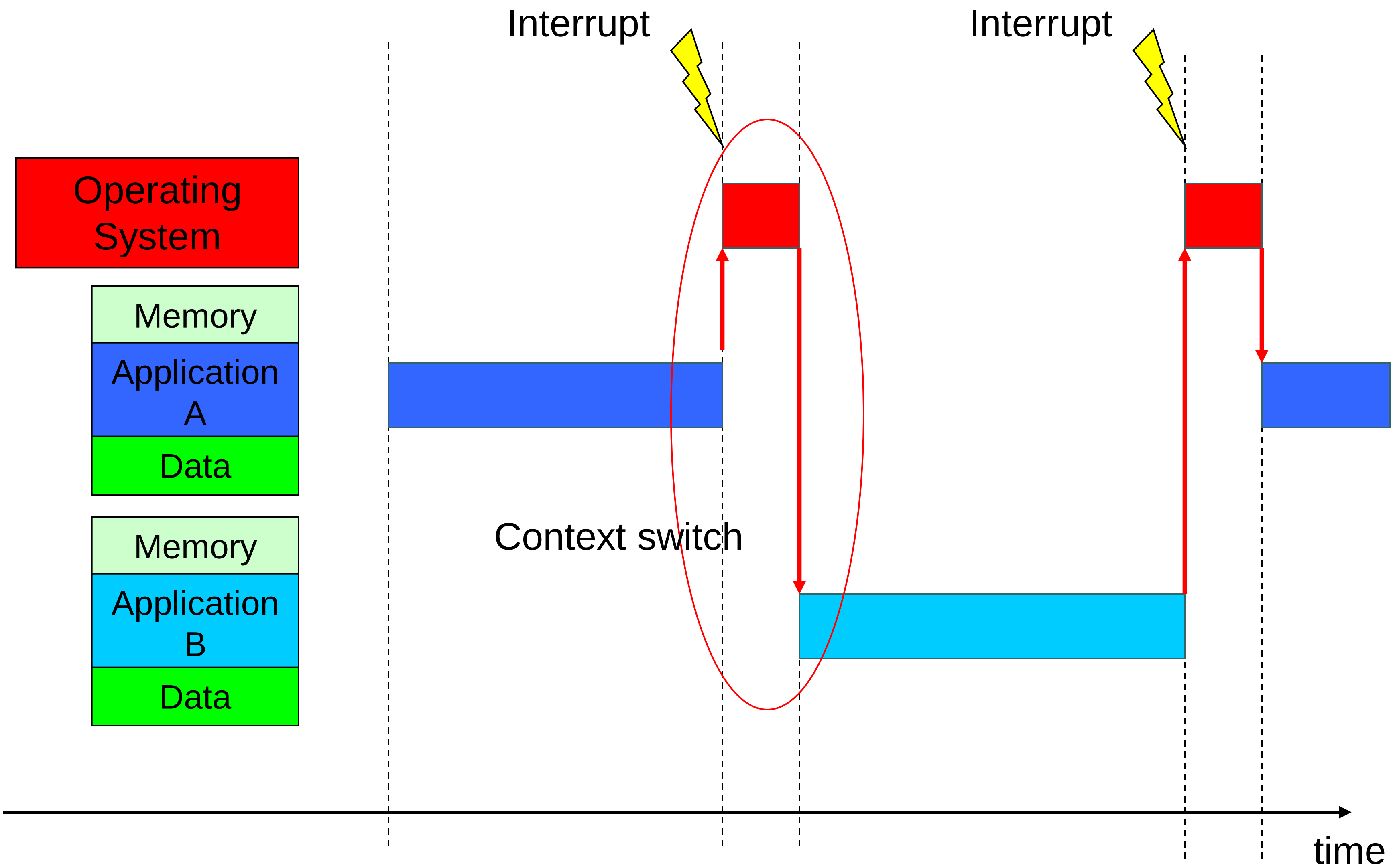
- Process
  - A program in execution
  - The OS presents a simpler ”virtual” computer for exclusive use to the program/programmer
- A process includes:
  - Program code
  - Program data and stack
    - The variables
  - State
    - for context switches

Snapshot of the state of the program in execution

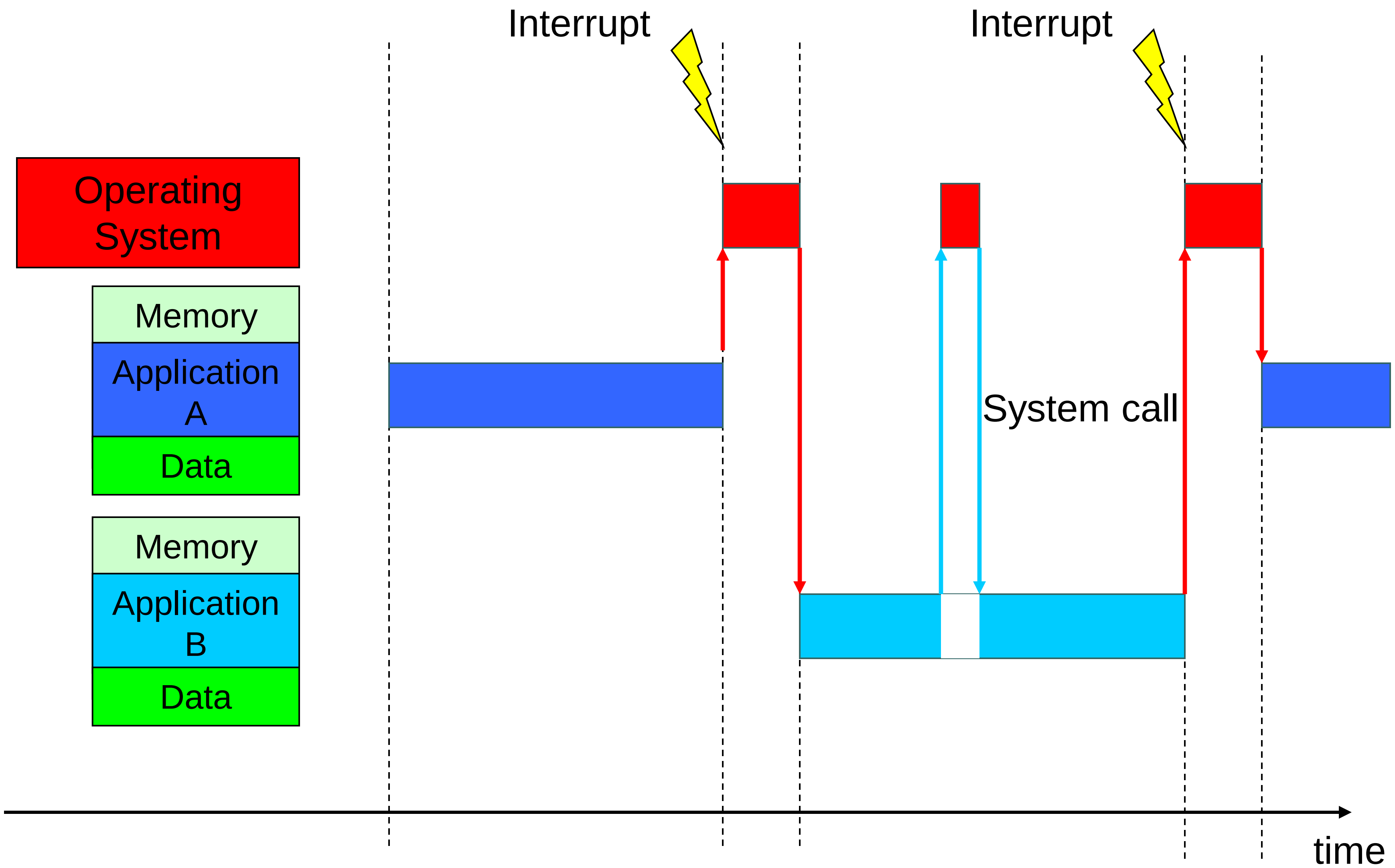
Sharing the processor



Sharing the processor



Sharing the processor



Context switch

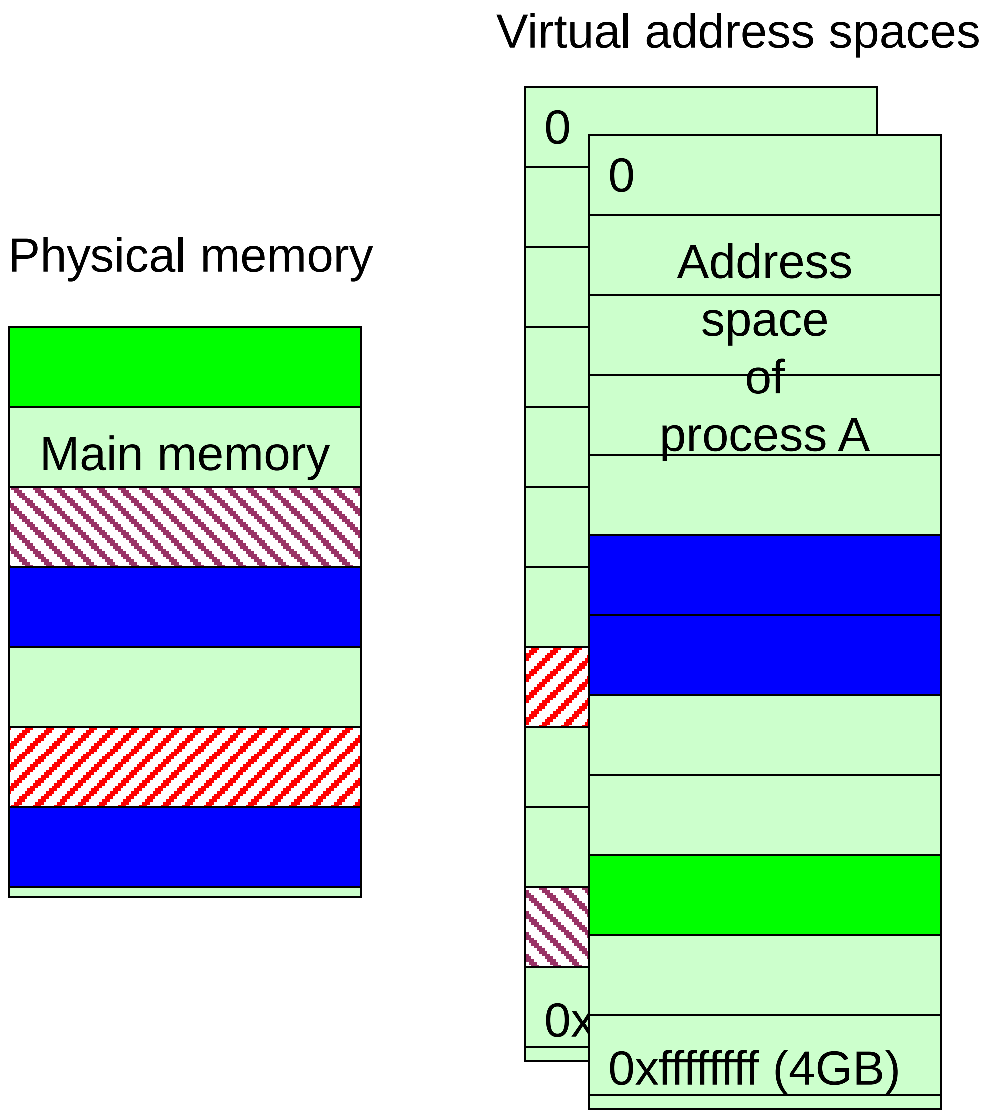
- When switching to another process
  - Save state of old process
  - Load state of new process
- Reasons for switch
  - Interrupts
  - Blocking operations
    - I/O
    - Process synchronization
- Scheduling
  - Choosing the process to switch to

Virtual Memory

- The illusion of having
  - As much memory as are addressable
    - Probably more that the available physical memory.
      - 64-bit adress => 16.8 million TB
  - All the memory by itself
- Allows the OS to move parts of processes on secondary memory
- Provides protection from other processes

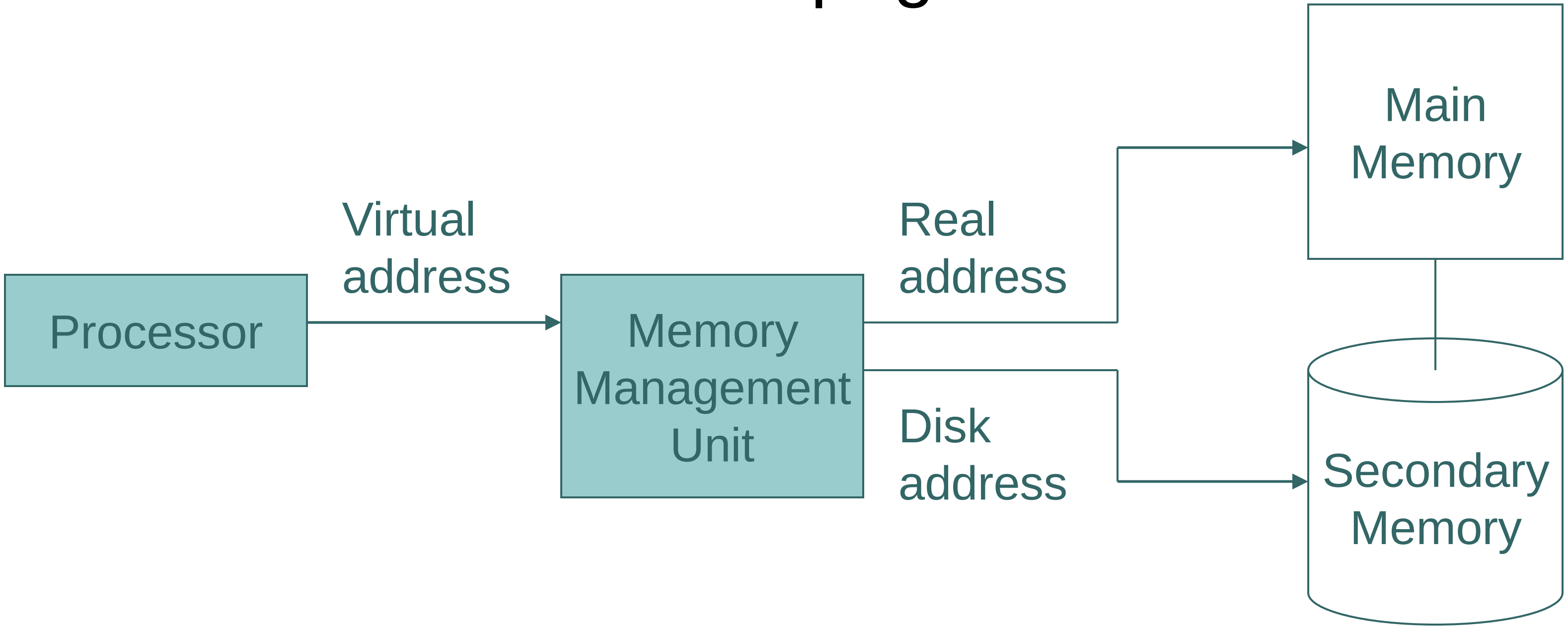
Paging

- Address space and main (physical) memory is divided into fixed-sized pages
- Each page may be located anywhere in main memory



Virtual memory addressing

- A virtual address is the combination of
  - A page number
  - An offset within the page



When physically memory filled

- Pages can be written to disk
  - They are ”paged out”
  - Memory becomes available for other pages
  - Chosen pages should be seldomly used
  - If a process uses paged out memory
    - Needs to be read back into main memory
    - Probably ends up in another location

Competition for resources

- Processes uses different resources
- Many resources cannot be shared at the same time between processes
  - Synchronization between processes are needed so it is used by one at a time
  - Locks are one tool to do this
- Problems can arise
  - Many processes wants to use many resources at the same time
  - The order of aquiring them becomes important

Deadlock

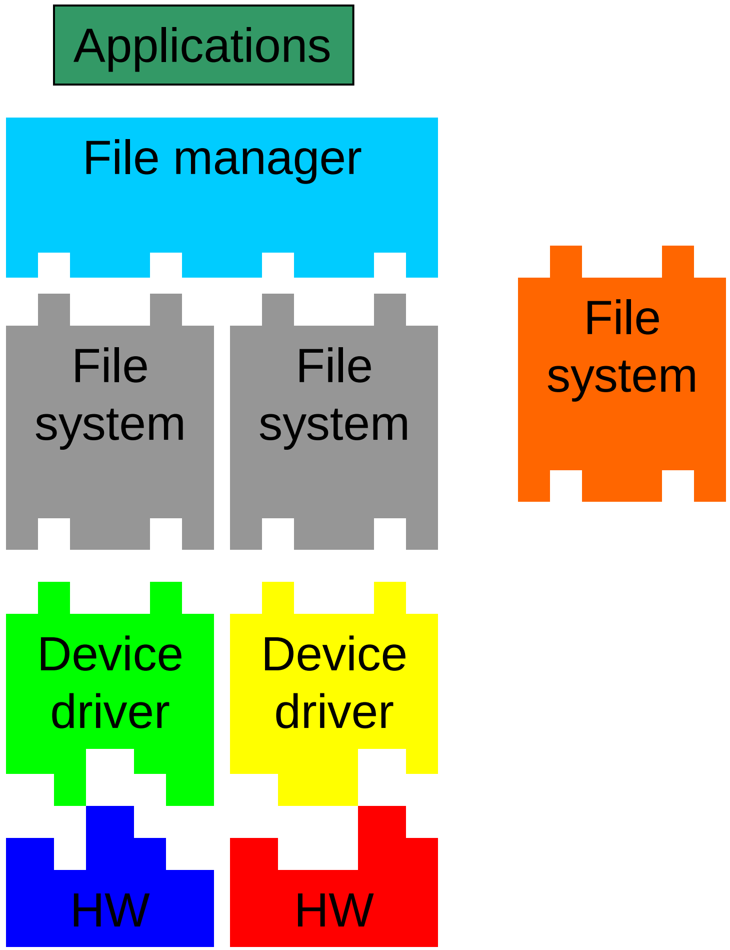
- If a set of processes are all waiting on some other process in the set a deadlock has occured.
- Example:
  - Two processes wants to transfer money between the same two accounts
  - Process A has got account 1
  - Process B has got account 2
  - Process A needs account 2 to do its transfer
  - Process B needs account 1 to do its transfer

Deadlock: Solution and Starvation

- Solve the deadlock: one process backs off
  - Release the resource
  - The other one can complete
  - Try again to acquire both resourses
- Starvation can occur
  - If the same process backs of every time it might never finish
- Other solutions can avoid this

File systems

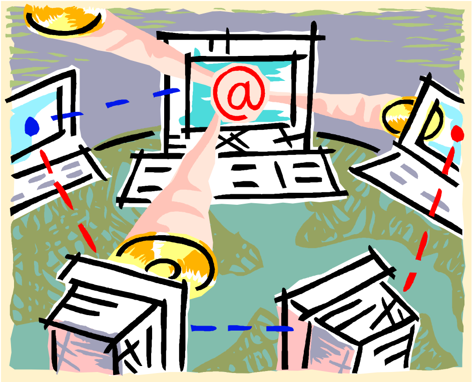
- An abstraction that provides
  - Long-term information storage in named files and directories
  - Allows hierarchical organization of data
  - Standard interface for applications
- The implementation is layered
  - Storage device
  - Device driver
  - File system implementation
  - OS file manager and application interface



Roadmap

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

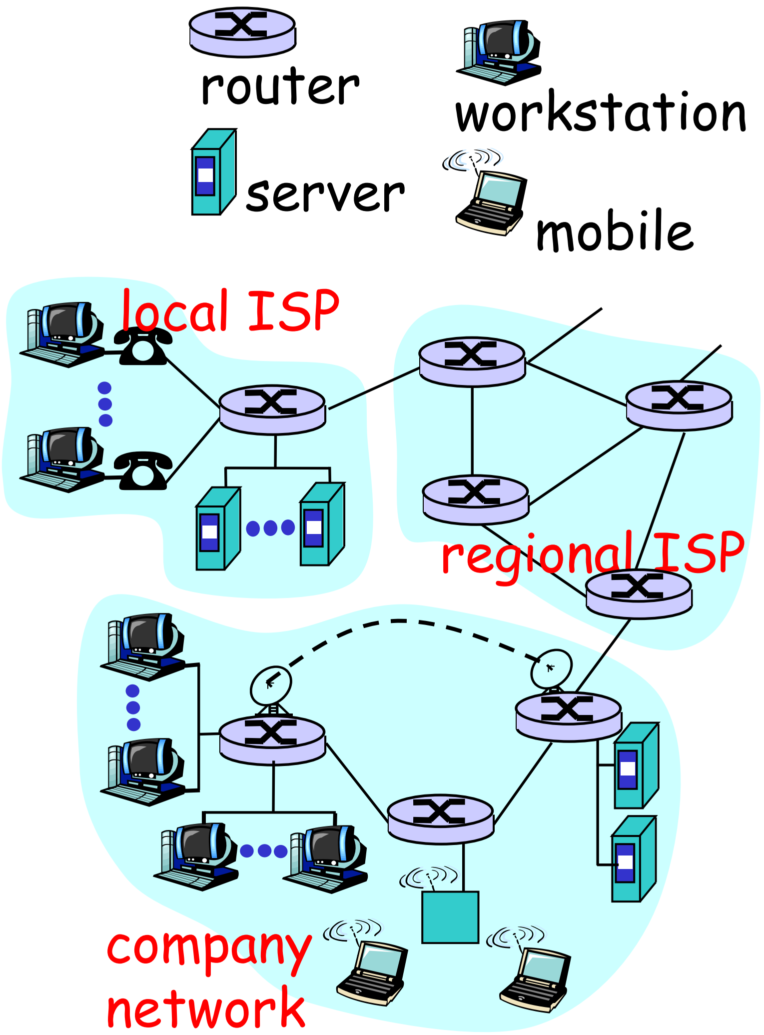
Networking



- Purpose
  - Allow applications (on different computers) to talk to each other

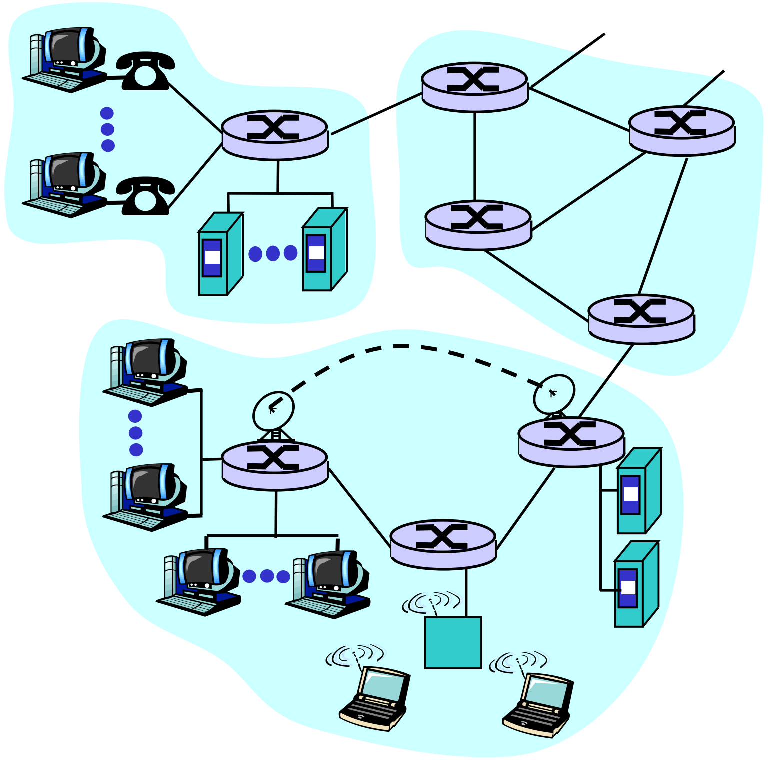
What's the Internet: "nuts and bolts" view

- millions of connected devices: *hosts, end-systems* running *network apps*
- *communication links*
- "network of networks"
  - connecting "devices" : *hubs, bridges, routers*
- *protocols*: control sending, receiving of msgs
- Internet standards
  - RFC: Request for comments
  - IETF: Internet Engineering Task Force



What's the Internet: a service view

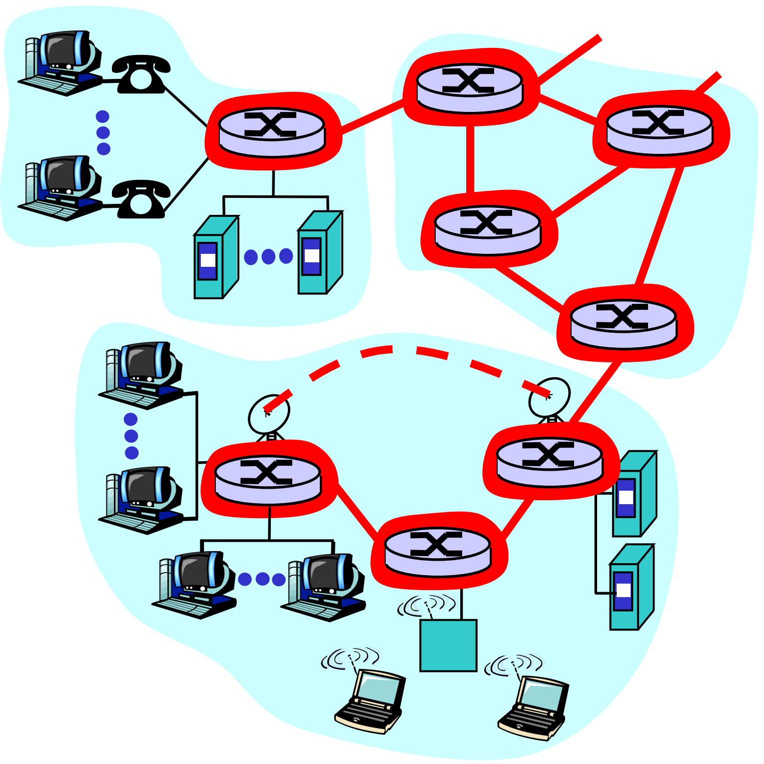
- communication *infrastructure* enables distributed applications:
- communication services: various types are provided/possible



A closer look at network structure:

The network core:

- mesh of interconnected routers
- fundamental question: how is data transferred through net?
  - circuit switching: dedicated circuit per call: telephone net
  - packet-switching: data sent thru net in discrete "chunks"
  - hybrid form: virtual circuits



Network Core: Packet Switching

each end-end data stream divided into *packets*

- user packets *share* network resources
- resources used *as needed*

store and forward:

- packets move one hop at a time
  - transmit over link
  - wait turn at next link

resource contention:

- aggregate resource demand can exceed amount available
- congestion: packets queue, wait for link use

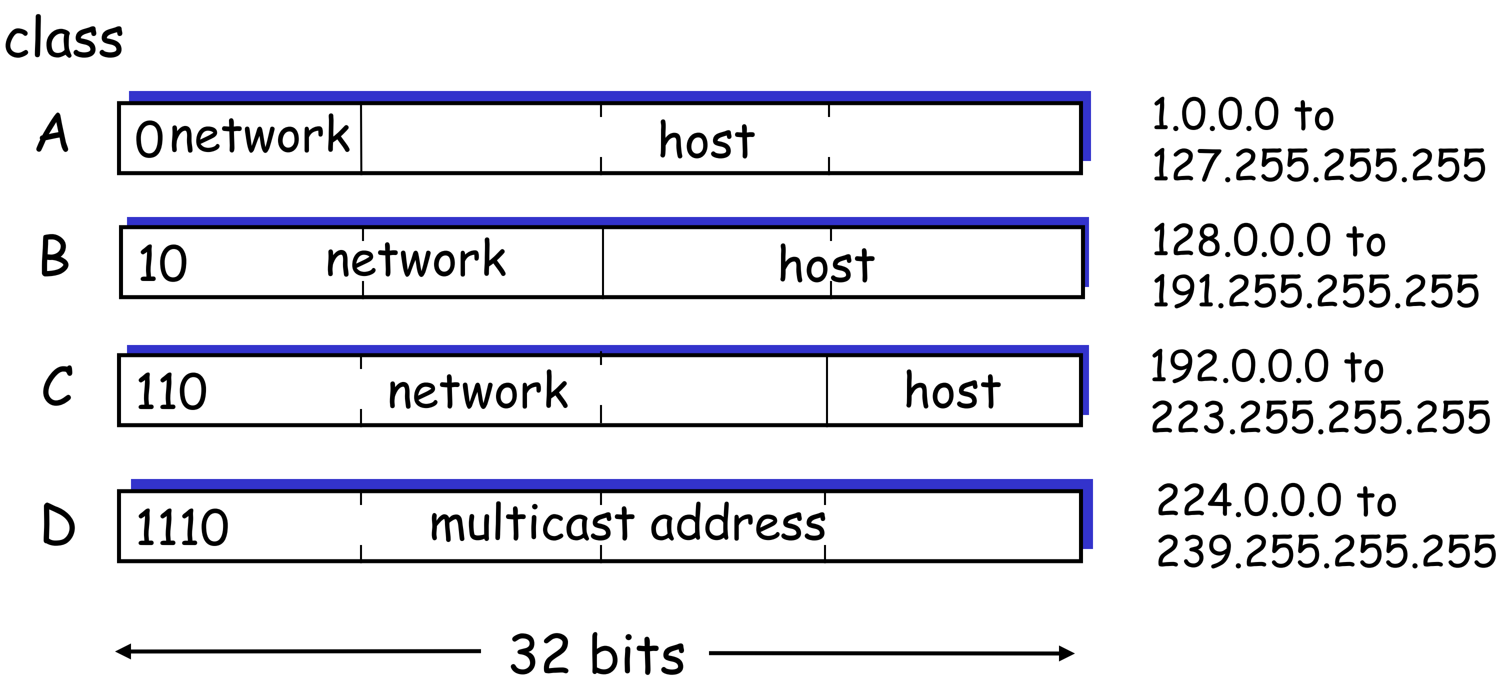
Internet Addressing

- An Internet host is identified by
  - IP-address (IPv4)
    - A unique 32 bit id number
    - Hierarchical w.r.t. routing
  - Domain Name Service (DNS) name
    - Human readable name
    - Hierarchical w.r.t. country, organization etc.
    - Eg. zsh.chalmers.se
      www.chalmers.se

IPv4 Addresses

"class-full" addressing:

given notion of "network", re-examine IP addresses:



IPv6

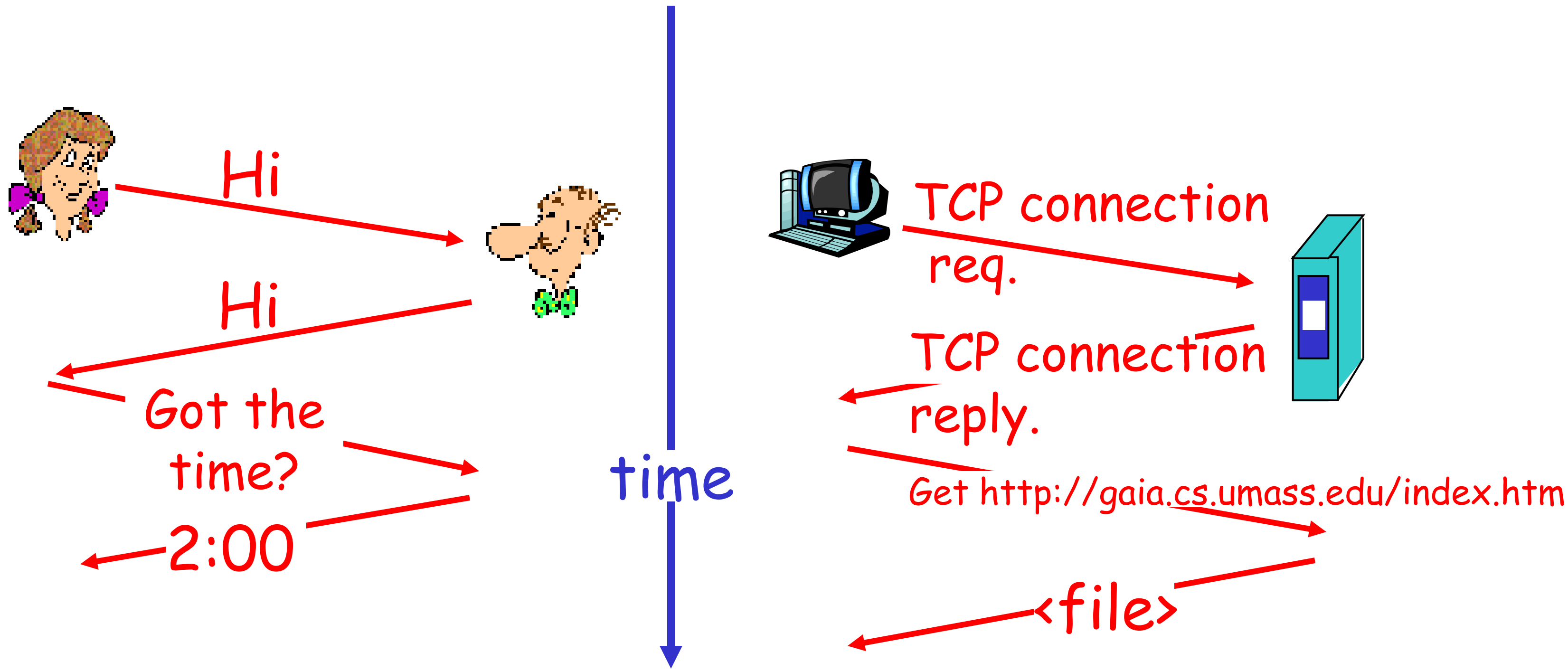
- “Normal” IP have 4 billion addresses
  - A lot of them are wasted
  - Chalmers alone have 65535 addresses
- We are running out of addresses
- Solution: IPv6
  - 128 bit addresses
    - 50 billion billion billion addresses / person
  - Allows solutions that is more hierarchical
    - We can waste address space freely

Roadmap

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

What's a protocol?

a human protocol and a computer network protocol:



protocols define format, order of msgs sent and received among network entities and actions taken on msg transmission, receipt

Protocol "Layers"

Networks are complex!

- many "pieces":
  - hosts
  - routers
  - links of various media
  - applications
  - protocols
  - hardware, software

Question:

Is there any hope of *organizing* structure of network?

Or at least our discussion of networks

Why layering?

Dealing with complex systems:

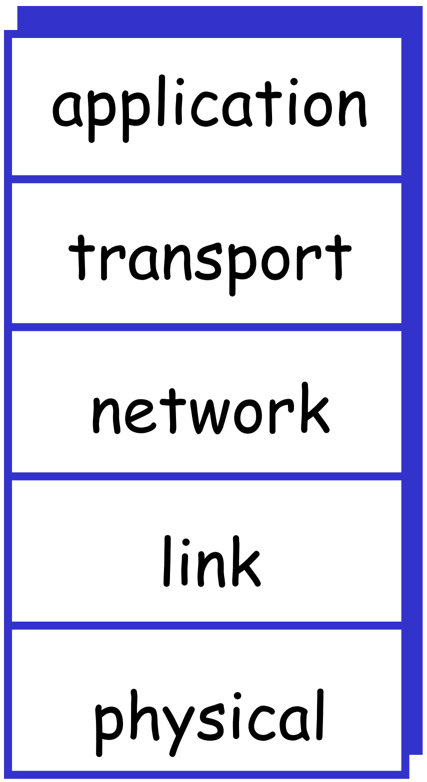
- explicit structure allows identification, relationship of complex system's pieces
  - layered reference model for discussion
- modularization eases maintenance
  - change of implementation of layer's service transparent to rest of system
  - e.g., change in gate procedure doesn't affect rest of system

Terminology: Protocols, Interfaces

- Each layer offers **services** to the upper layers (shielding from the details how the services are implemented)
  - service interface: across layers in same host
- Layer n on a host carries a conversation with layer n on another host (data are not sent directly)
  - host-to-host (aka peer-to-peer) interface: defines messages exchanged with peer entity
- Interfaces must be clean
  - min info exchange
  - make it simple for protocol replacements
- Network architecture (set of layers, interfaces) vs protocol stack (protocol implementation)

Internet protocol stack

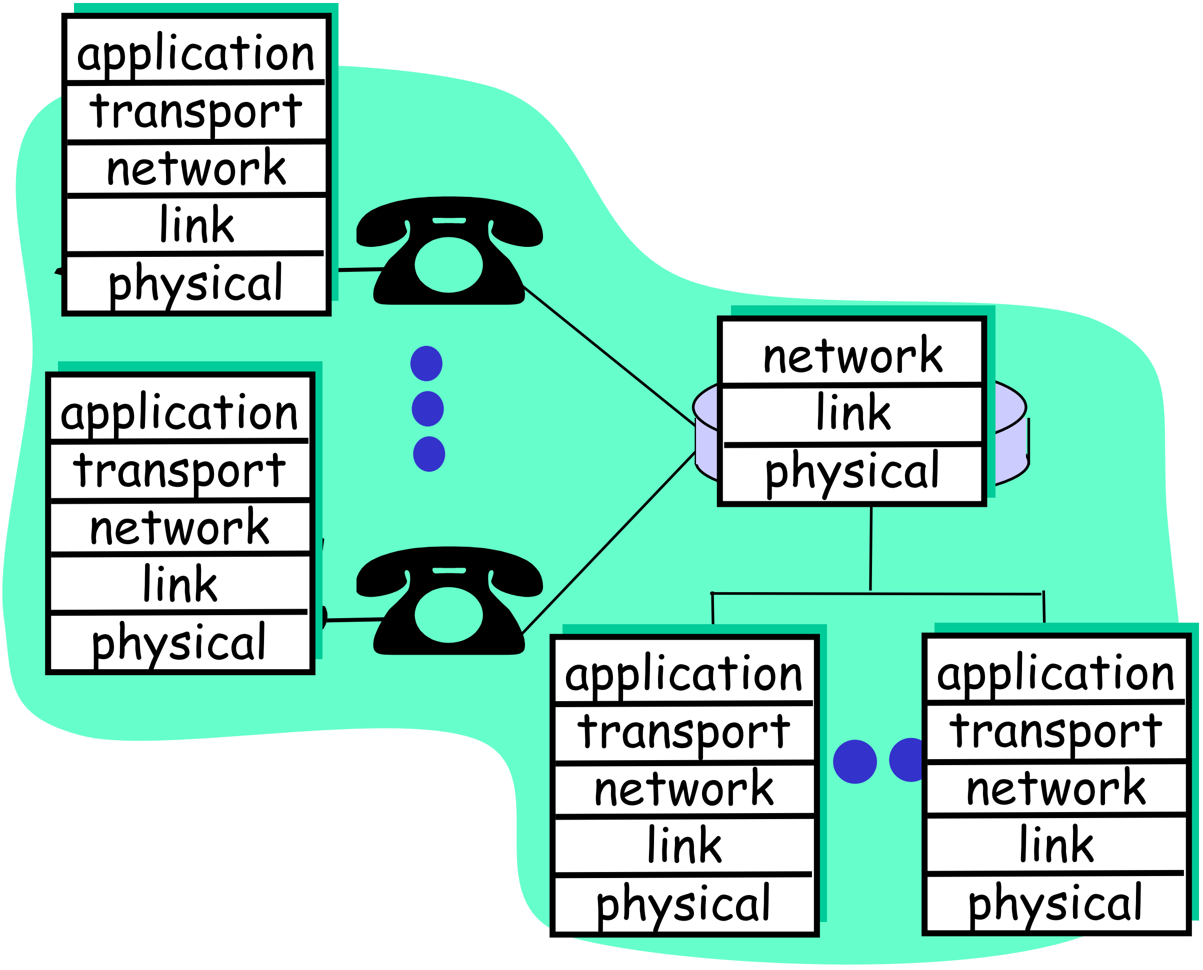
- application: ftp, smtp, http, etc
- transport: tcp, udp, …
- network: routing of datagrams from source to destination
  - ip, routing protocols
- link: data transfer between neighboring network elements
  - ppp, ethernet
- physical: bits "on the wire"



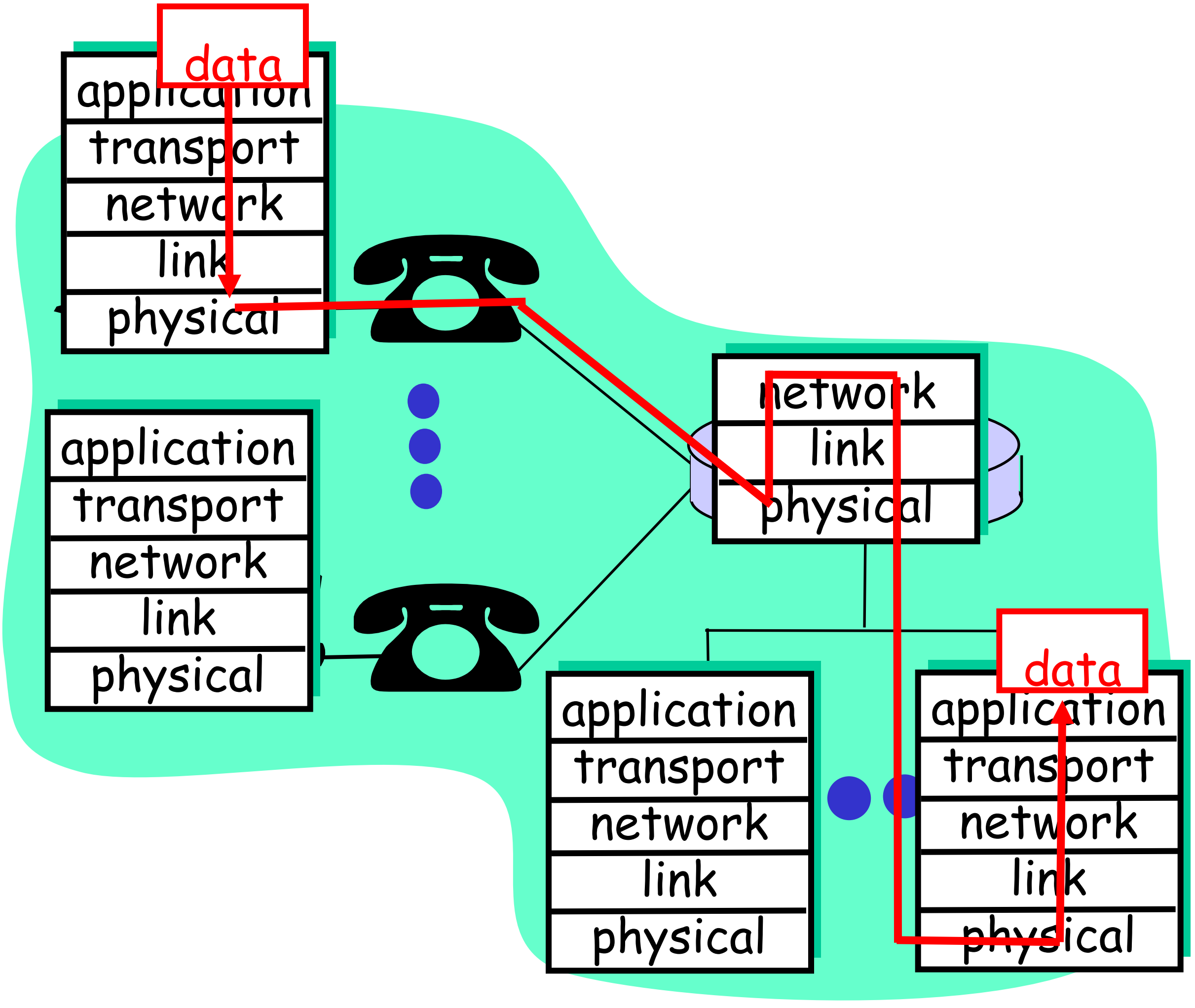
Layering: logical communication

Each layer:

- distributed
- "entities" implement layer functions at each node
- entities perform actions, exchange messages with peers

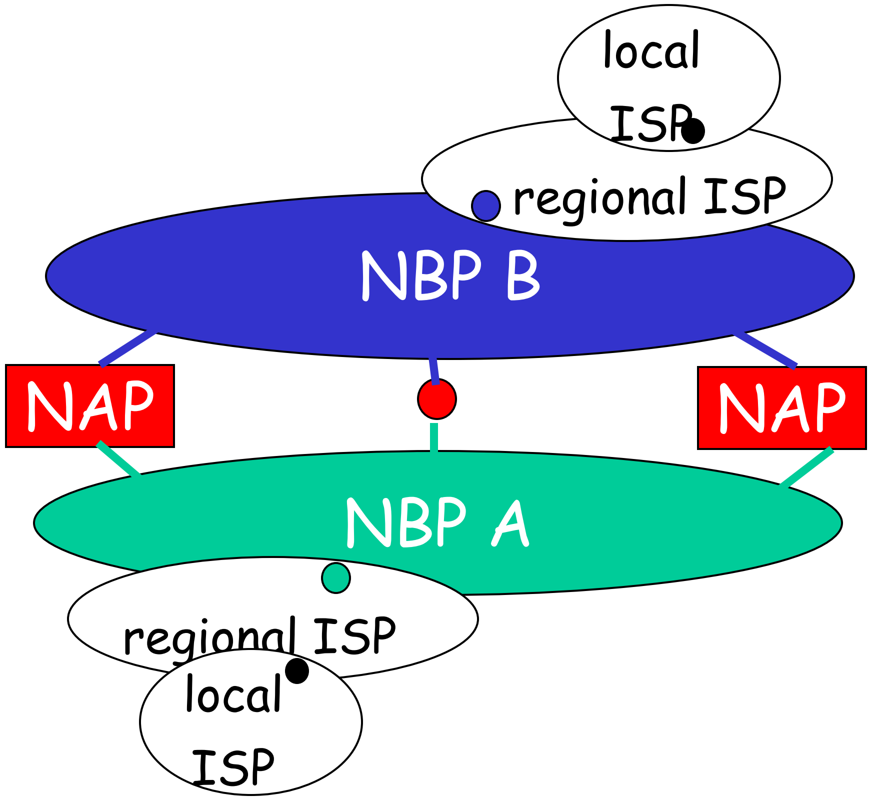


Layering: physical communication



Internet structure: network of networks

- roughly hierarchical
- national/international backbone providers (NBPs)
  - e.g. BBN/GTE, Sprint, AT&T, IBM, UUNet
  - interconnect (peer) with each other privately, or at public Network Access Point (NAPs: routers or (ATM) NWs of routers)
- regional ISPs
  - connect into NBPs
- local ISP, company
  - connect into regional ISPs



Roadmap

- Operating Systems
  - What is an Operating System
  - OS evolution
  - OS details
- Networking
  - The Internet
  - Network protocols
  - Security

What is network security?

Confidentiality: only sender, intended receiver should "understand" msg contents

- sender encrypts msg
- receiver decrypts msg

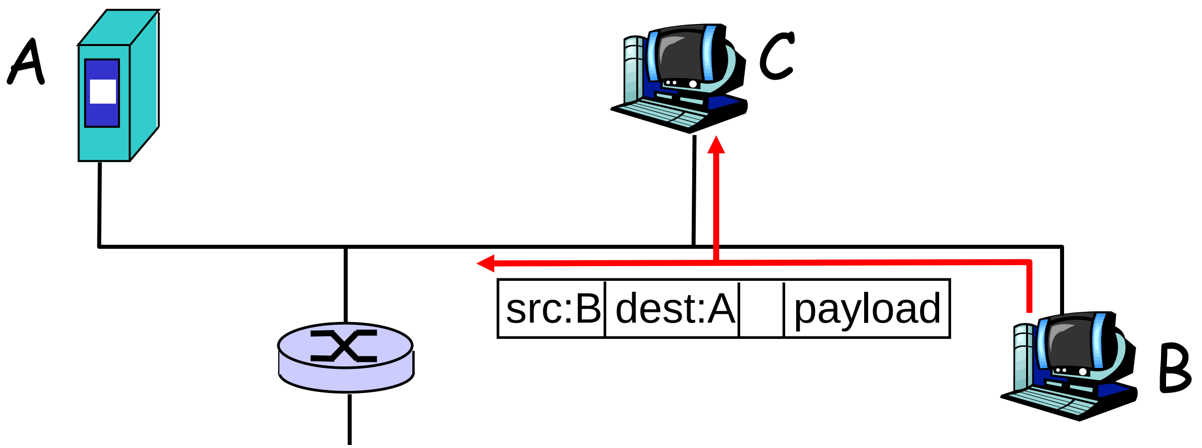
Integrity of Messages: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

Authentication: sender, receiver want to confirm identity of each other

Internet security threats

Packet sniffing:

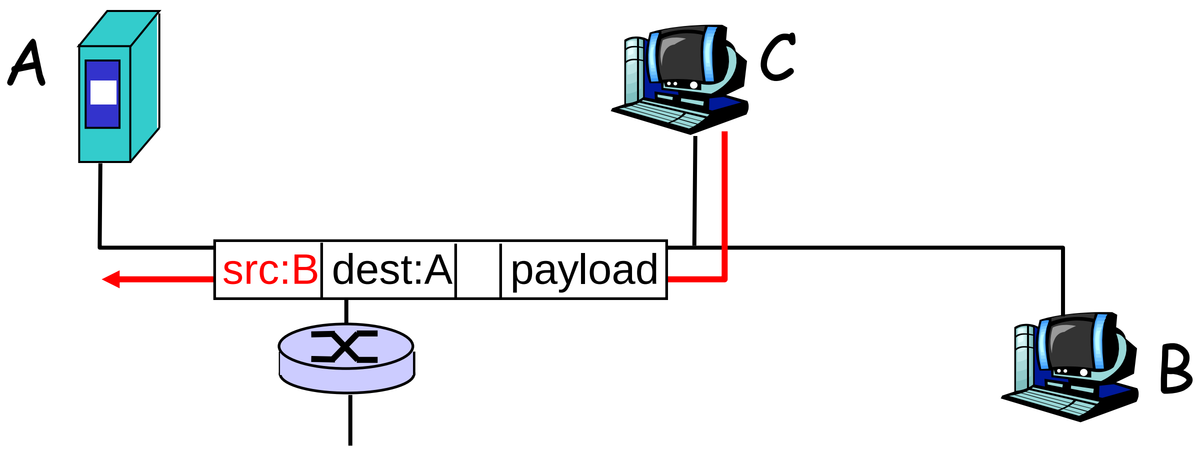
- broadcast media
- promiscuous NIC reads all packets passing by
- can read all unencrypted data (e.g. passwords)
- e.g.: C sniffs B's packets



Internet security threats

IP Spoofing:

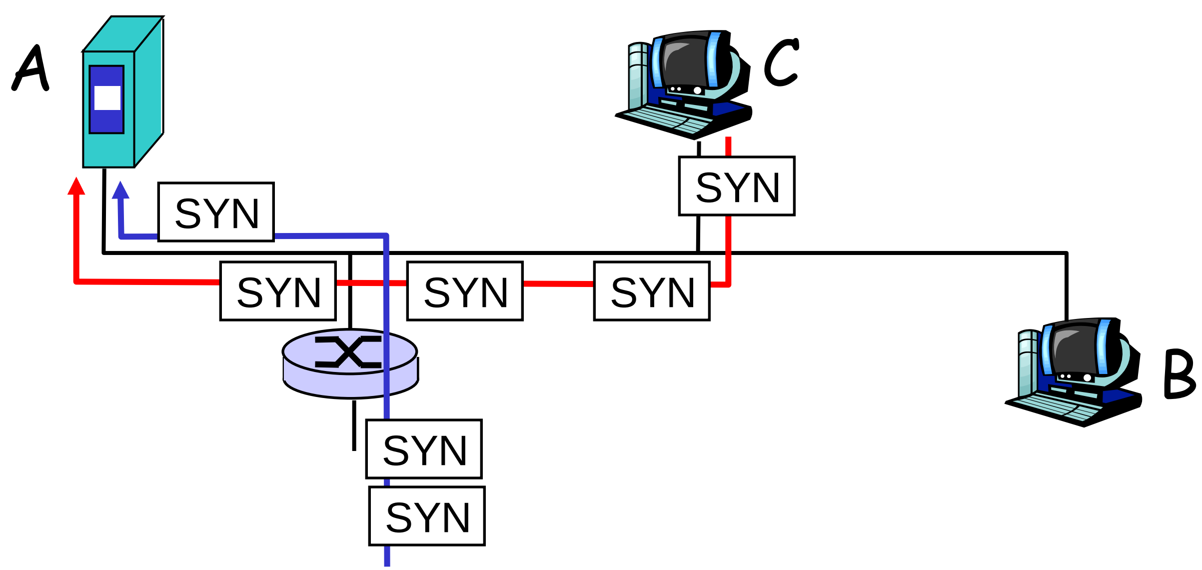
- can generate "raw" IP packets directly from application, putting any value into IP source address field
- receiver can't tell if source is spoofed
- e.g.: C pretends to be B



Internet security threats

Denial of service (DOS):

- flood of maliciously generated packets "swamp" receiver
- Distributed DOS (DDOS): multiple coordinated sources (or, rather, spoofed packets) swamp receiver
- e.g., C and remote host SYN-attack A



Encryption

- Symmetric
  - Encryption/Decryption with the same key
  - Key distribution a problem
- Public key encryption
  - Encryption with public key
  - Decryption only with private key
  - Key distribution still a problem

Viruses and Worms

- Virus
  - Malicious code fragment
  - Attaches itself to other programs on the host computer
- Worms
  - “Virus” that infect computers through the network.
  - The host has to run some vulnerable network service
- Trojan
  - Program with a hidden malicious agenda

Firewalls

firewall

isolates organization's internal net from larger Internet, allowing some packets to pass, blocking others.

Two firewall types:

- packet filter
- application gateways

To prevent denial of service attacks:

- SYN flooding: attacker establishes many bogus TCP connections. Attacked host alloc's TCP buffers for bogus connections, none left for "real" connections.

To prevent illegal modification of internal data.

- e.g., attacker replaces CIA's homepage with something else

To prevent intruders from obtaining secret info.

Questions?

- Contact Information:
  - Address:

    Pierre Kleberger
    Computer Science & Engineering
    Chalmers University of Technology
  - Email:

    pierre.kleberger@chalmers.se
  - Web:

    http://www.chalmers.se/cse/EN/people/kleberger-pierre