Chapter 1: roadmap

1.1 What is the Internet?

1.2 Network edge

- end systems, access networks, links

1.3 Network core

- Packet vs Circuit Switching ✔
- Structure: hierarchy of networks ✔

1.4 Performance:

- delay, loss and throughput ✔

1.5 Layered Architecture

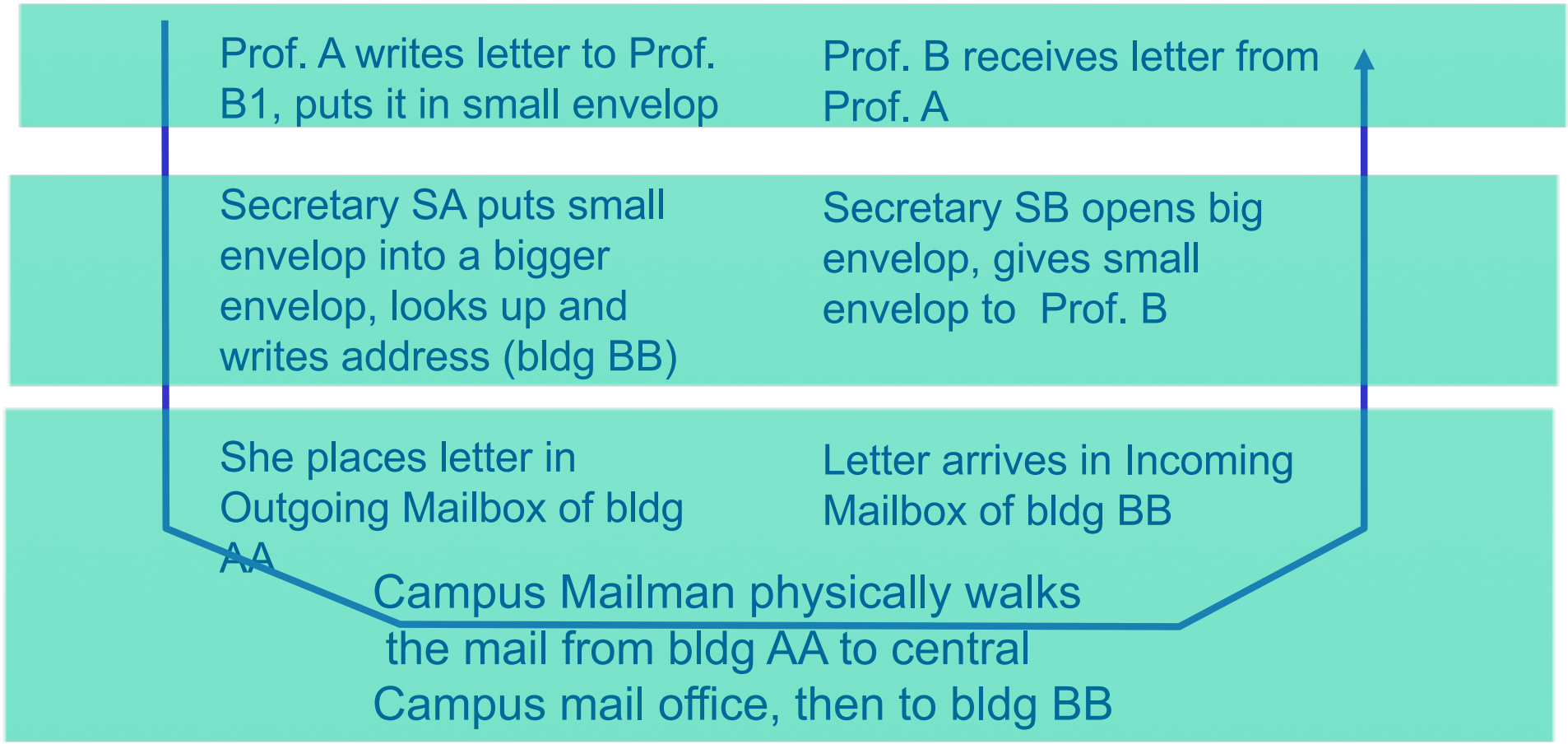
1.6 Networks under attack: security

1.7 History

Dealing with Scale and Complexity

- The Internet is highly complex, in terms of …
  - Size (number of components)
  - Number of tasks it needs to manage: routing, congestion control, packet reordering, connection establishment …
  - Diversity of Components, Topology, Functionality,
- Tasks are structured through modularization
  - Divide and Conquer
  - Break down into smaller pieces
  - Create different functional layers

Layering of Campus Mail



layers: each layer implements a service

- via its own internal-layer actions
- relying on services provided by layer below

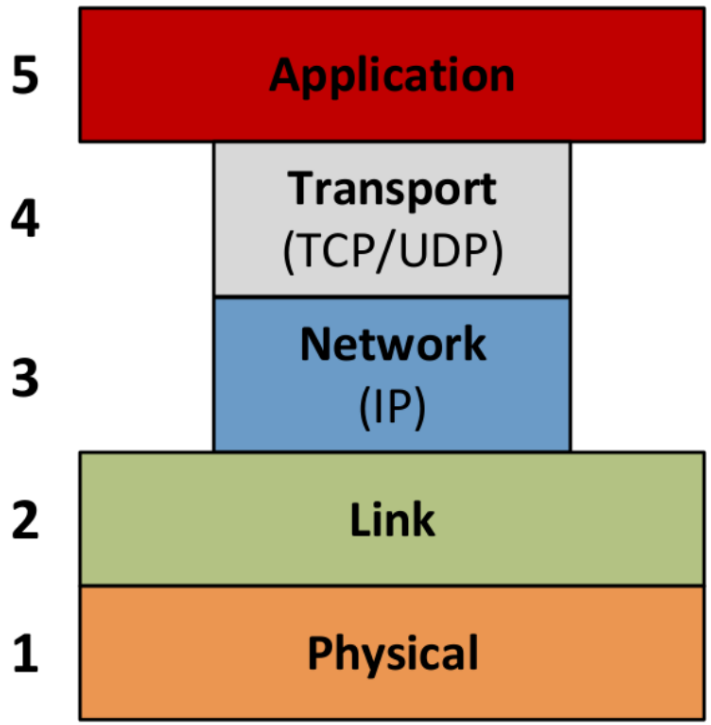
Why layering?

dealing with complex systems:

- explicit structure allows identification, relationship of complex system's pieces
  - layered *reference model* for discussion
- modularization eases maintenance, updating of system
  - change of implementation of layer's service transparent to rest of system
  - e.g., change in secretary, secretary's routine, mailman's routine, doesn't affect rest of system
- layering considered harmful?
  - duplication of functionality?
  - cross-layer optimization?

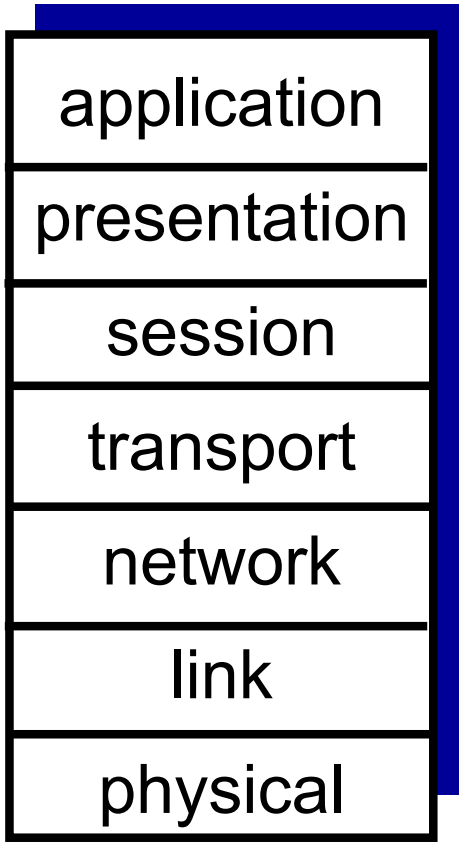
Internet protocol stack

- *Application layer:*
  - What we interact with: HTTP, FTP, SMTP
- *Transport layer:*
  - process-process data transfer
  - end-to-end management: TCP, UDP
- *Network layer:*
  - routing of datagrams from source to destination
  - IP, routing protocols
- *Link Layer:*
  - data transfer between neighboring network elements
  - Ethernet, 802.111 (WiFi), PPP
- *physical:*
  - network medium (fiber, wireless,…)
  - bits on the wire



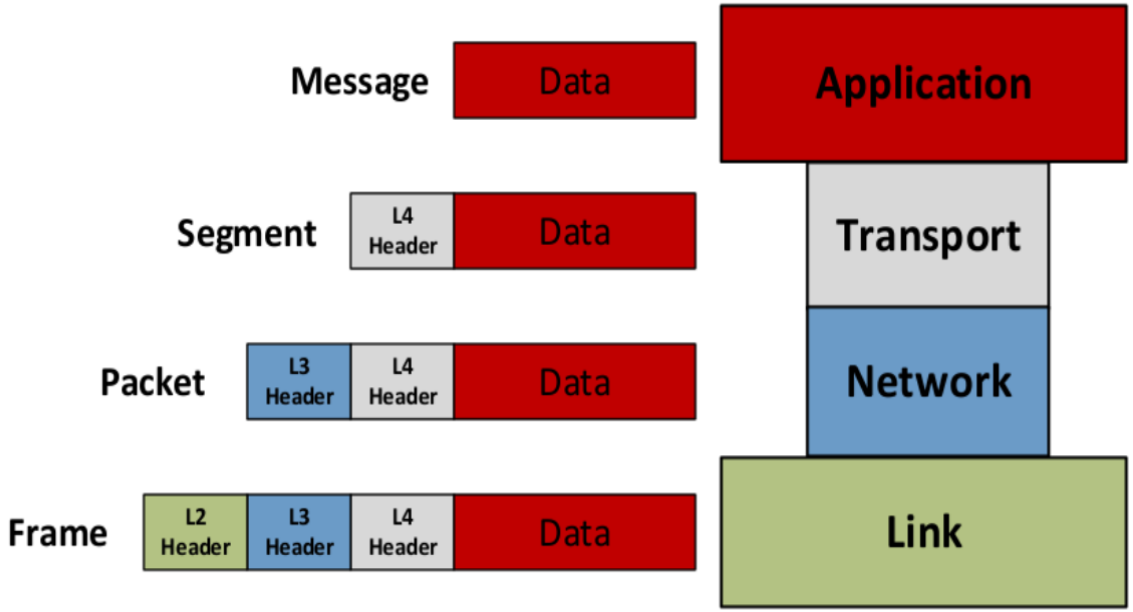
[ISO/OSI reference model]

- *presentation:* allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- *session:* synchronization, checkpointing, recovery of data exchange
- Internet stack "missing" these layers!
  - these services, *if needed,* must be implemented in application



Headers

- Encapsulation: augment data with headers
  - Payload: actual content
  - Header: identification and control information



- This creates overhead. Why bother?
  - Allows us to distinguish messages – essential for packet switching
  - Allows us to implement functionality: e.g.: layer 3 header contains dst IP address, layer 4 header contains seqnum to order packets

Encapsulation