ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Network Protocol Design and Evaluation

02 - Design Principles

Stefan Rührup

University of Freiburg
Computer Networks and Telematics

Summer 2009

CoNe Freiburg

IIF
INSTITUT FÜR INFORMATIK FREIBURG

In the last lecture...

- **Specification: 5 Elements of a protocol**
  - Service
  - Assumptions about the environment
  - Vocabulary of messages
  - Encoding (format) of messages
  - Procedure rules
- **2 Examples for design problems**
  - incomplete specification
  - design flaws

Today

- **Design aspects**
  - What are the properties of a good protocol?

- **Internet design principles**
  - Design goals
  - Development of the Internet protocols

Design Aspects

[Holzmann 1991]

- **Simplicity**
- **Modularity**
- **Well-formedness**
  - neither over- nor under-specified
  - bounded, self-stabilizing, self-adapting
- **Robustness**
- **Consistency**
  - avoidance of deadlocks, livelocks,
    or improper terminations

Simplicity

- **Lean design**
  - A protocol should be built from a small number of elements
  - Each element focuses on one function

- **Think about the next steps...**
  A lean design makes it easier to implement, verify and maintain

Modularity

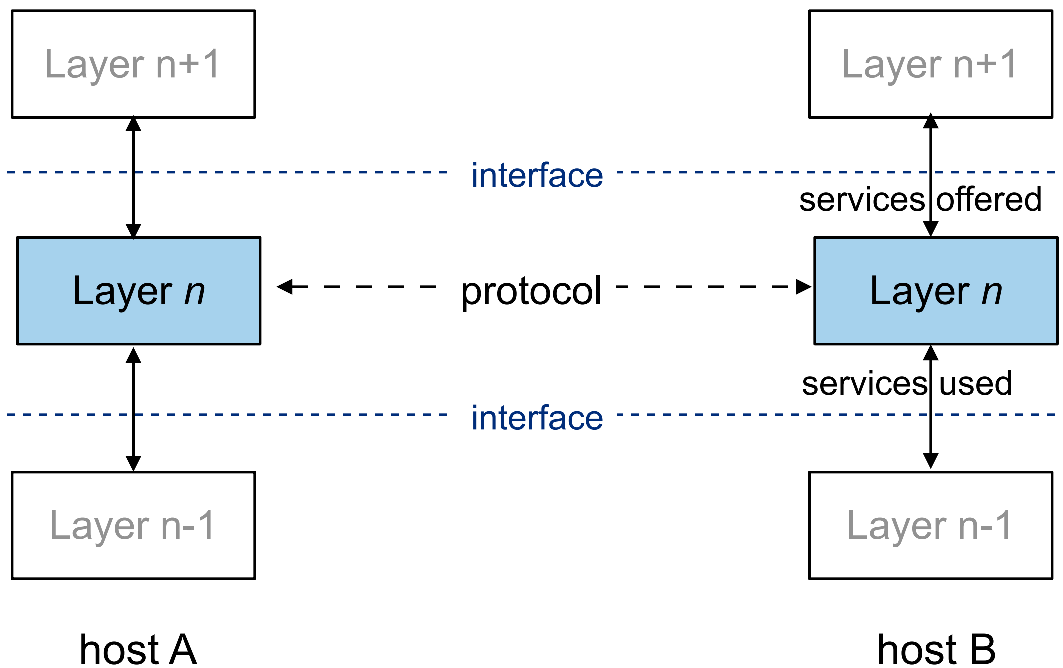
- Complex functions should be built from independent and individual light-weight modules
- Decoupling of orthogonal functions
- No assumptions about other modules
- Main structuring techniques:
  - protocol layering
  - structuring of data

Modularity - Protocol Layering

- **Modularity by layering**
  - separating higher level tasks from lower level details
  - example: OSI model

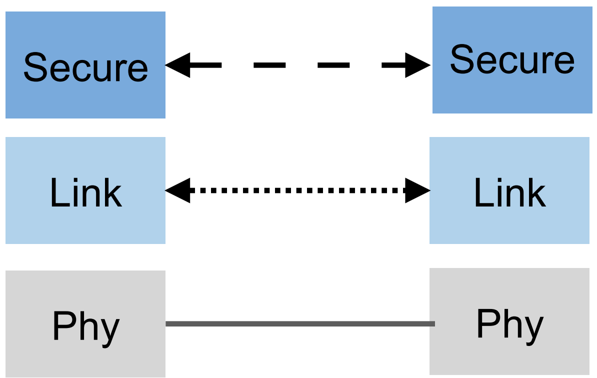
- **Protocol layers**
  - define levels of abstraction
  - should integrate related functions
  - should have small and well-defined interfaces

Protocol Layering - Service Model



Protocol Layering Example

- A protocol for secure data transmission over a raw physical data link:
  - handling of transmission errors
  - flow control
  - key exchange
  - encoding/decoding
- Decoupling of methods for reliable data transmission and security functions
- Link layer and security layer provide independent services



Protocol Layering - the OSI model

layer		data unit
Application	application function calls	data
Presentation	data format conversion	data
Session	session establishment between end systems	data
Transport	end-to-end connection and data transfer	segment
Network	routing and logical addressing	packet
Data Link	medium access, flow control and phys. addressing	frame
Physical	definition of the physical medium	bit

Protocol Layering

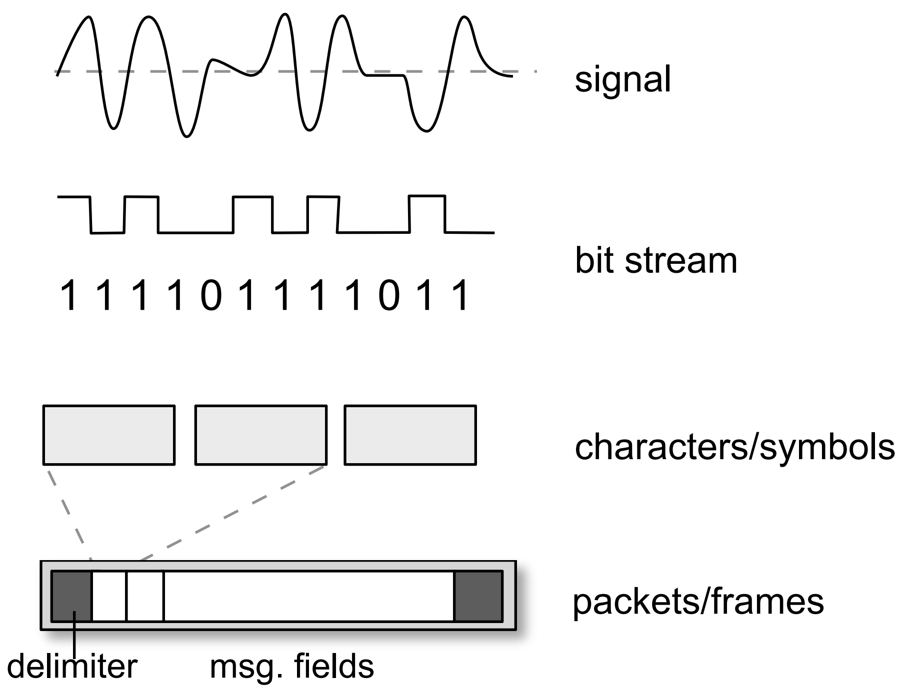
- **Advantages**
  - Layering allows to break complex problems into smaller pieces
  - Implementation as light-weight modules
  - Modules are exchangeable (interface specification is independent from the implementation)
  - Modules are reusable
- **Problems**
  - Information hiding can lead to performance loss

Modularity - Data structuring

- **Low level data formatting:**
  - Bit-oriented, character-oriented, or byte-count oriented
  - frame delimiters: bit sequence, character, or indicated by a counter
- **Higher level formatting**
  - Structured headers and trailers
    - sequence numbers, checksums
    - sender, receiver, priorities, ...

Modularity - Data structuring

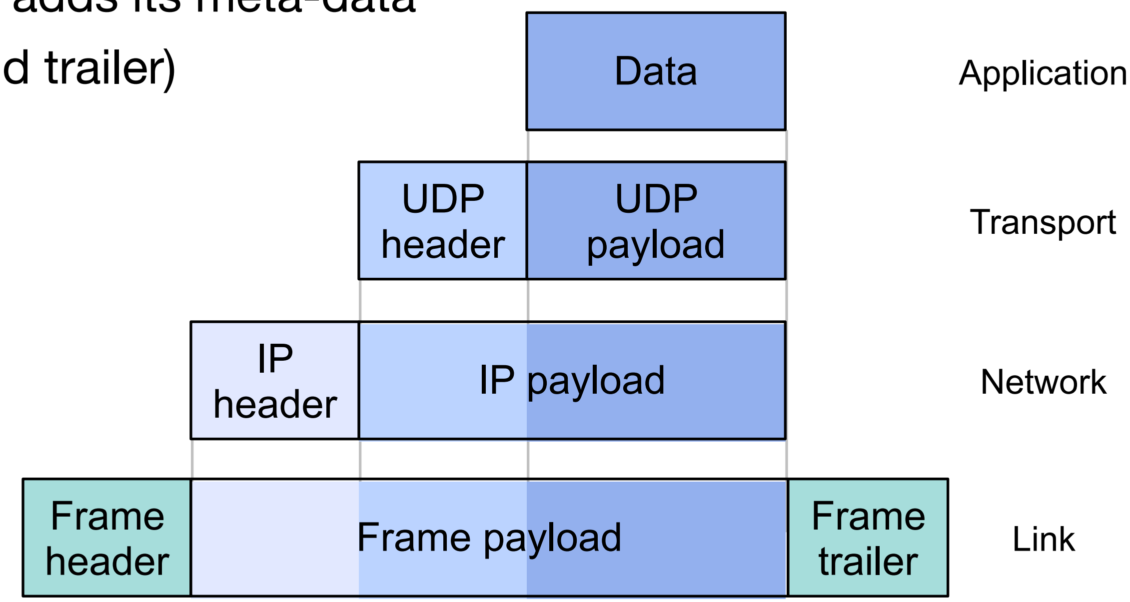
- **Levels of abstraction**



[Holzmann 1991]

Modularity - Data structuring

- Consequence of Layering and Data structuring: **Encapsulation**
- Each layer adds its meta-data (header and trailer)



Well-formedness

- A well-formed protocol is
  - **neither over- nor under-specified** (redundancy or incompleteness)
  - **bounded**: it attends to system limits (memory limits)
  - **self-stabilizing**: it returns to a defined state after a transient error occurred
  - **self-adapting**: it adapts to environmental changes (e.g. flow control)

Robustness

- proper execution under **all** possible conditions

- the protocol should adhere to a **minimal design**
  - minimal assumptions about the environment
  - avoidance of dependencies on other protocol elements, system parameters, etc.

Consistency

- Avoidance of
  - inconsistent states (*deadlocks*)
  - loops in protocol execution without progress (*livelocks*)
  - improper protocol termination

10 Rules of Design

[Holzmann 1991]

1. Make sure that the problem is well designed
2. Define the service first (*what* comes before *how*)
3. Design *external* functionality before the *internal* one
4. Keep it simple
5. Do not connect what is independent
6. Do not impose irrelevant restrictions (extendability)
7. Build a high-level prototype first and validate it
8. Implement the design, evaluate and optimize it
9. Check the equivalence of prototype and implementation
10. Don’t skip rules 1-7

Internet design principles

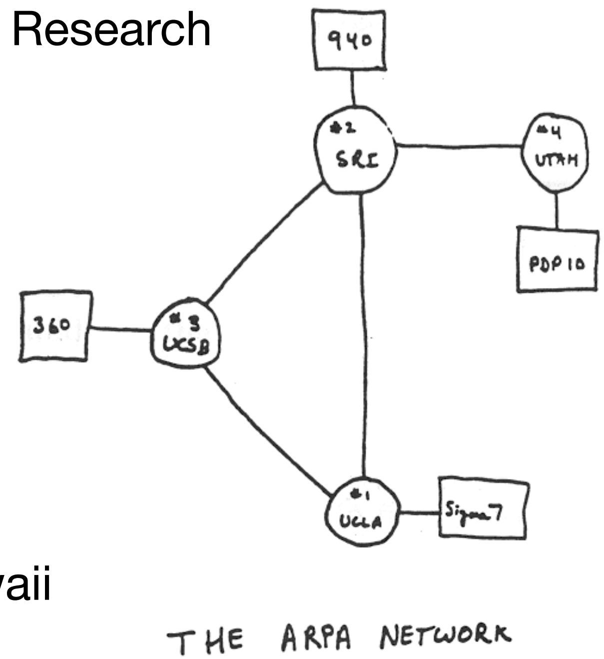
- Internet protocols (TCP/IP) were designed in the 1970s and are still successfully used
- Basic characteristics of Internet communication: packet switching, connectionless services, layered protocols
- What were the ideas behind TCP/IP?
- a little bit of history ...

(see Kurose and Ross, 2007)

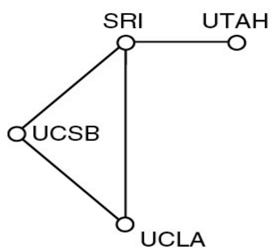
Internet History

1961-1972: Early packet-switching principles

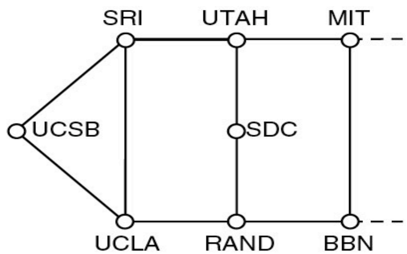
- **1961**: Kleinrock - queueing theory shows effectiveness of packet-switching
- **1964**: Baran - packet-switching for “survivable” networks
- **1967**: ARPANET conceived by Advanced Research Projects Agency
- **1972**:
  - ARPANET public demonstration
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
- **1970**: ALOHAnet satellite network in Hawaii



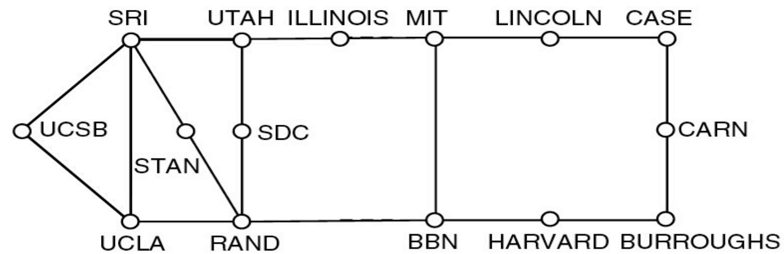
ARPANET Growth



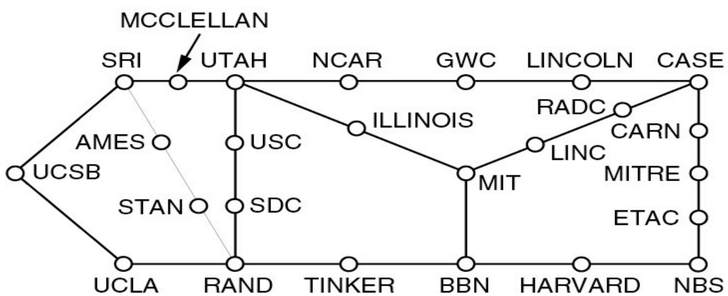
December 1969



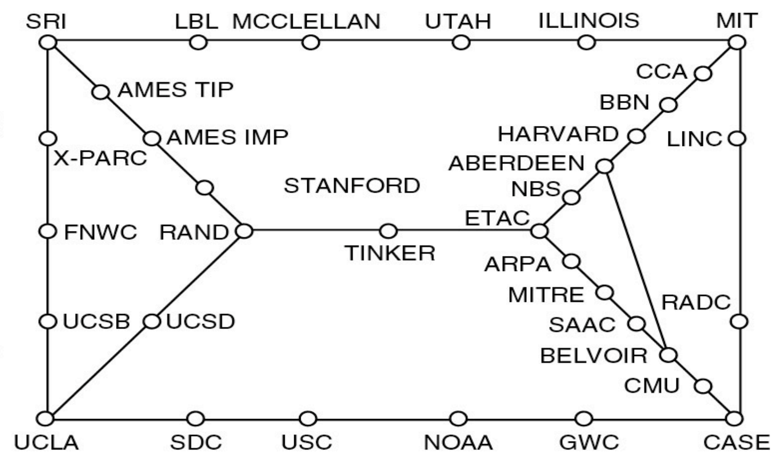
July 1970



March 1971



April 1972



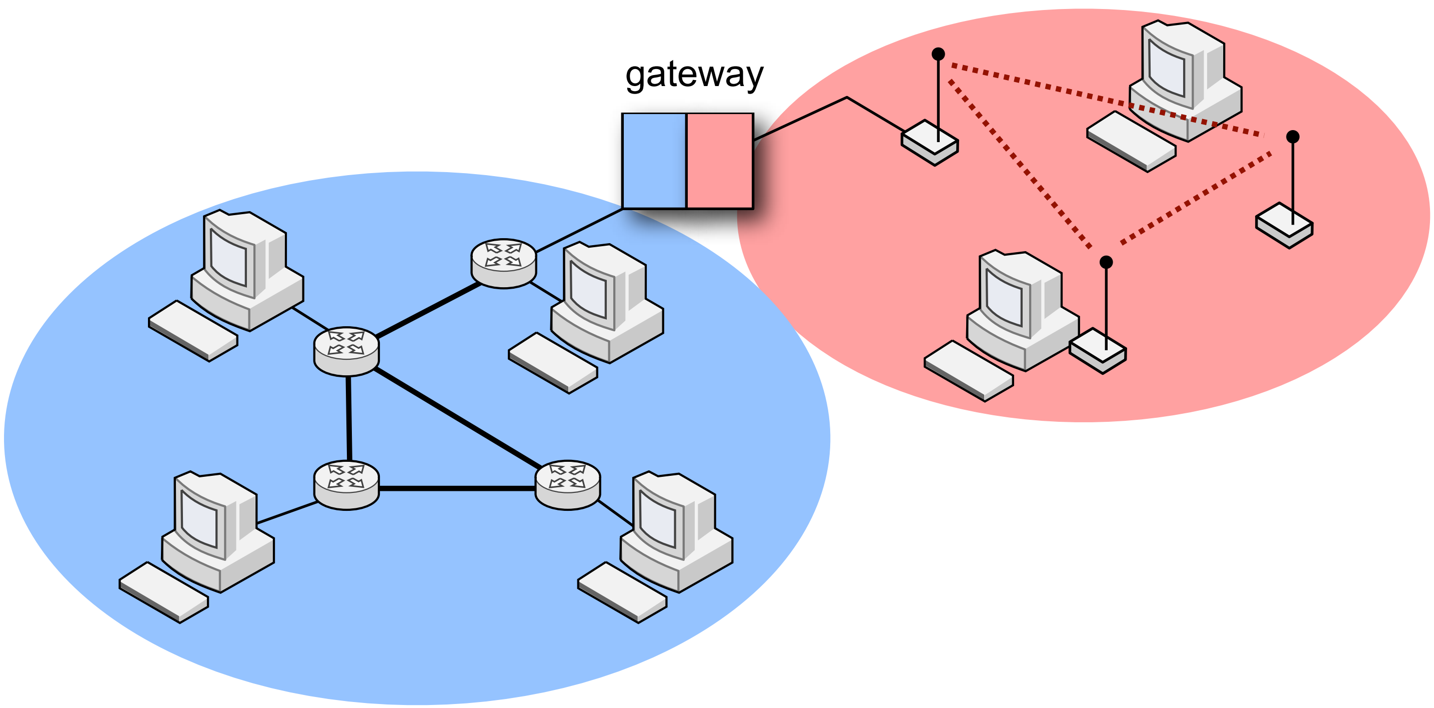
September 1972

[A.S. Tanenbaum, Computer Networks, 4/e, Prentice Hall]

Internet History

Early 1970s: Internetworking

- **1974**: Cerf and Kahn - architecture for interconnecting networks
  Goal: connection of existing networks (ARPA network and packet radio)



Internet History

Early 1970s: Internetworking

- **1974**: Cerf and Kahn - architecture for interconnecting networks

Cerf and Kahn's internetworking principles:
- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control

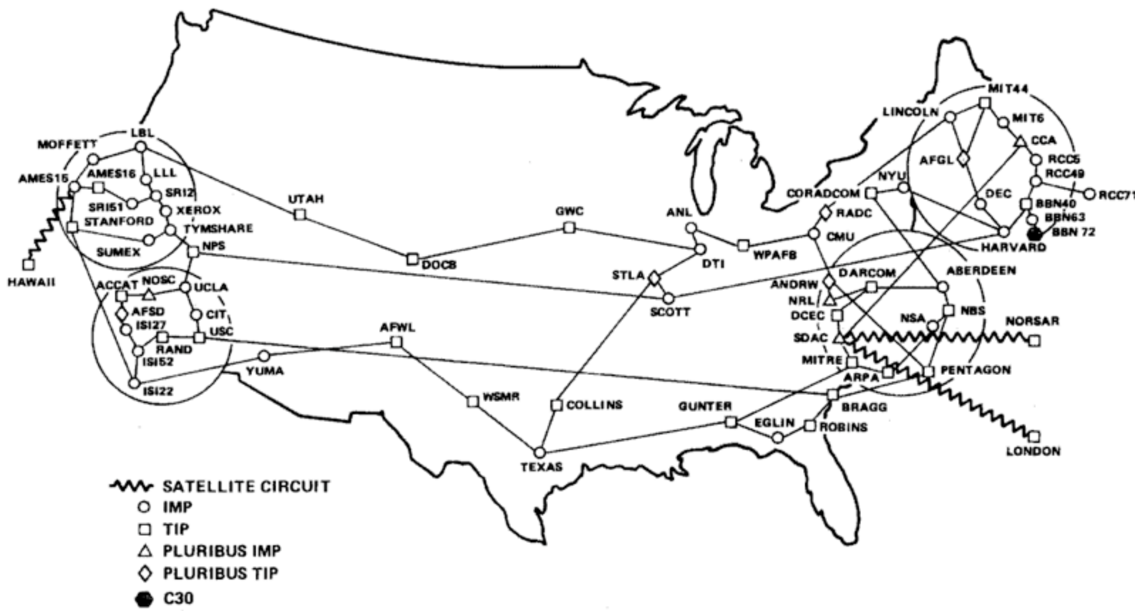
define today's Internet architecture

[V. Cerf, and R. Kahn, "A Protocol for Packet Network Intercommunication", 1974]

Internet History

Late 70s: New and proprietary nets

- **1976**: Ethernet at Xerox PARC
- **late 70's**: proprietary architectures: DECnet, SNA, XNA
- **late 70's**: switching fixed length packets (ATM precursor)
- **1979**: ARPANET has 200 nodes



Internet History

1980-1990: new protocols, a proliferation of networks

- **1983**: deployment of TCP/IP
- **1982**: smtp e-mail protocol defined
- **1983**: DNS defined for name-to-IP-address translation
- **1985**: ftp protocol defined
- **1988**: TCP congestion control
- new national networks: Csnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks

Internet History

1990, 2000's: commercialization, the Web, new apps

- **Early 1990's**: ARPANET decommissioned
- **1991**: NSF lifts restrictions on commercial use of NSFnet (decommissioned, 1995)
- **1990s**: Web
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
  - late 1990's: commercialization of the Web
- **Late 1990's – 2000's**:
  - more killer apps: instant messaging, P2P file sharing
  - network security to forefront
  - est. 50 million host, 100 million+ users
  - backbone links running at Gbps

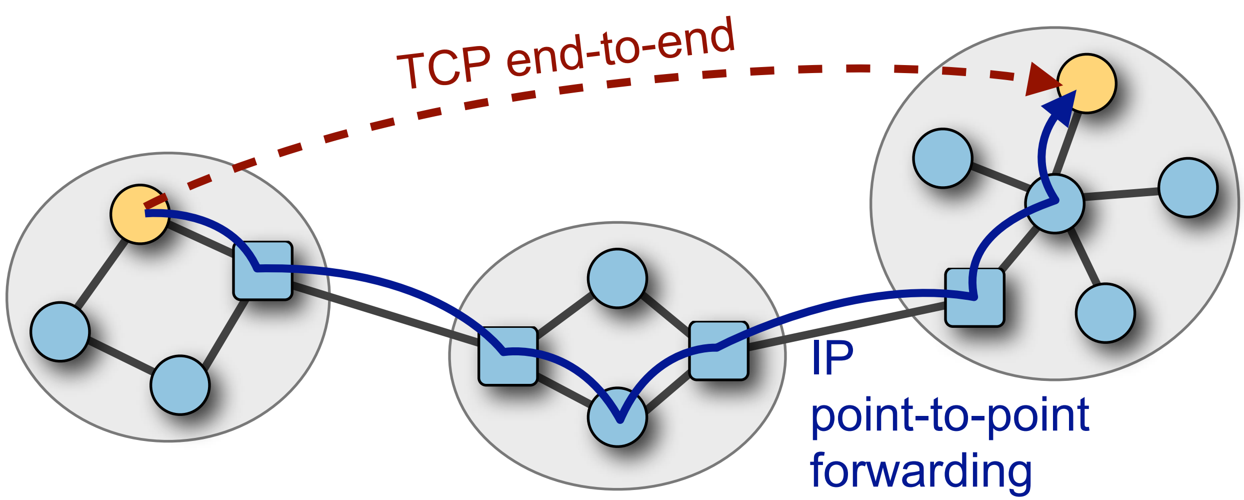
Internet History

- **2007**:
  - ~500 million hosts
  - Voice, Video over IP
  - P2P applications: BitTorrent (file sharing) Skype (VoIP), PPLive (video)
  - more applications: YouTube, gaming
  - wireless, mobility

... and communication still relies mainly on **TCP/IP**

TCP/IP Revisited

- **TCP:** reliable end-to-end transport service, uses IP
- **IP:** connectionless datagram service
- TCP/IP enables end-to-end communication over interconnected networks



Internet design goals

- **Primary goal:**
  Effective multiplexed utilization of existing interconnected networks

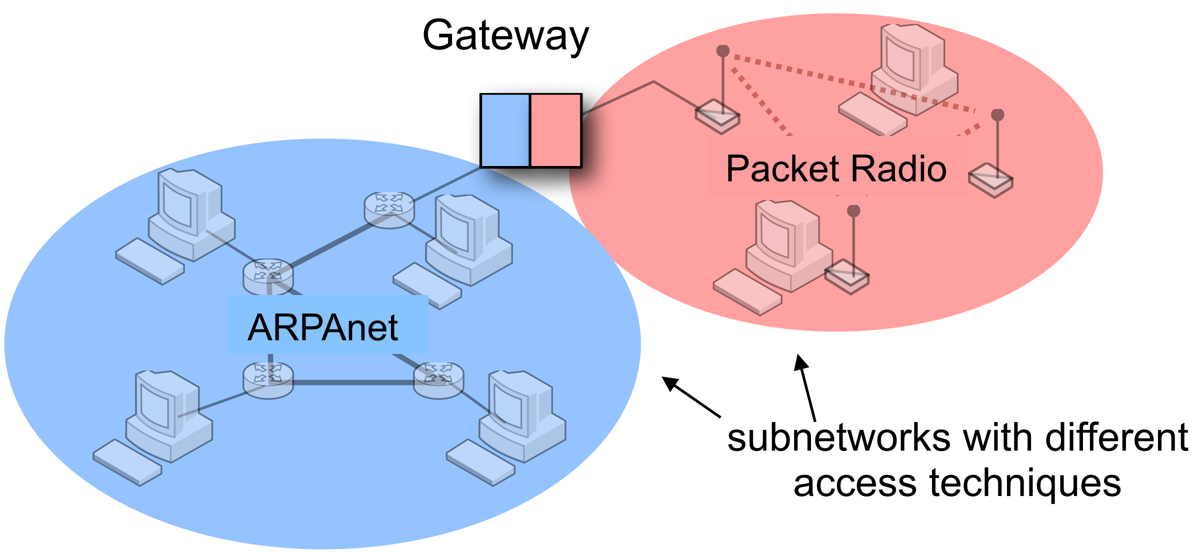
- Components are packet switched networks

- **Design choices:**
  - Multiplexing by **packet switching**
  - **Store and forward** communication through gateways

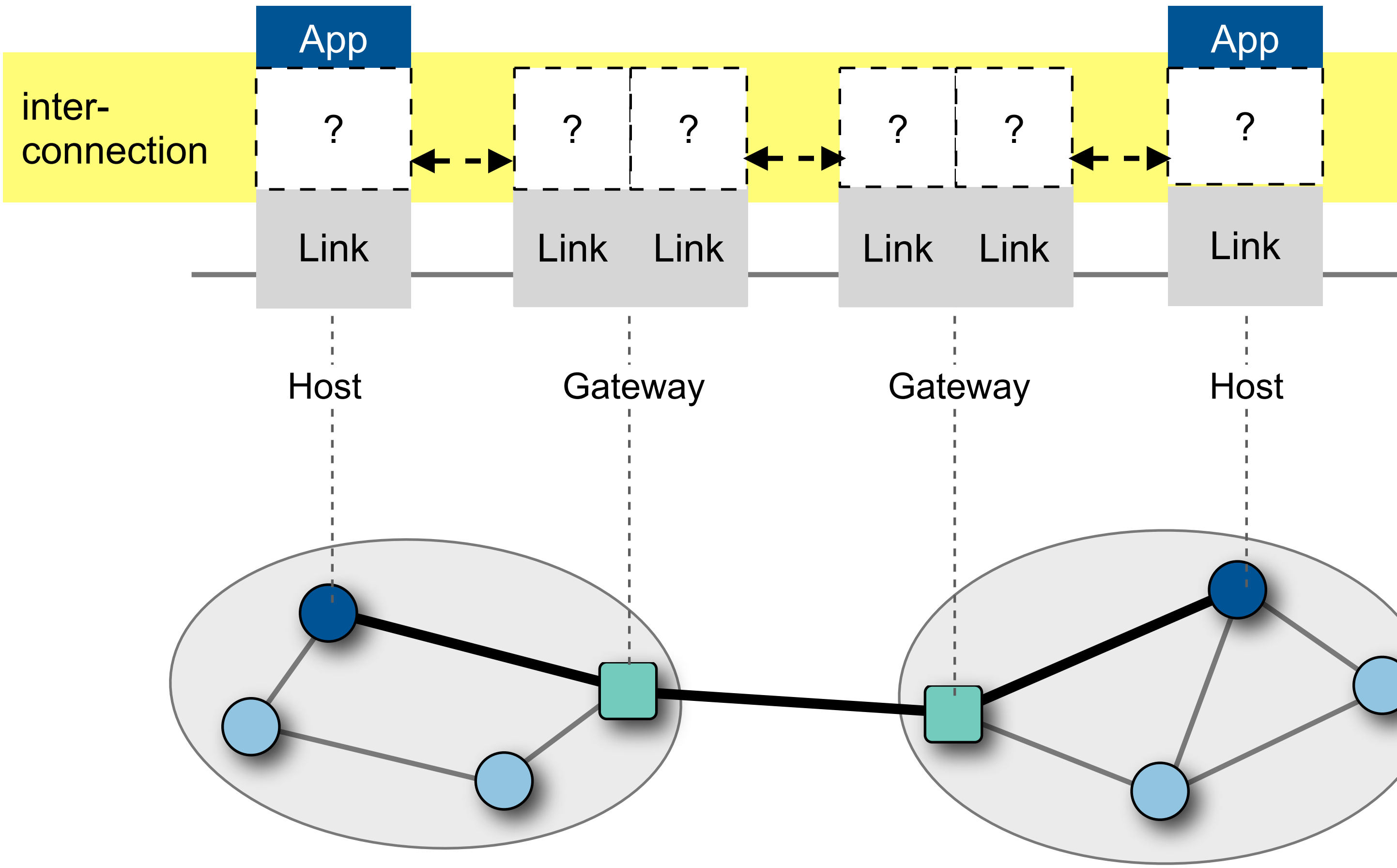
[D. Clark “The Design Philosophy of the DARPA Internet Protocols”, SIGCOMM 1988]

Internet design goals

- **Network Interconnection** (Network of networks)
- Gateways translate between subnetworks



Internet design goals



Internet design goals

Second level goals (in the order of importance)

1. Internet communication must continue despite loss of networks or gateways (survivability).
2. The Internet must support multiple types of communications service.
3. The Internet architecture must accommodate a variety of networks.
4. The Internet architecture must permit distributed management of its resources.
5. The Internet architecture must be cost effective.
6. The Internet architecture must permit host attachment with a low level of effort.
7. The resources used in the internet architecture must be accountable.

[D. Clark "The Design Philosophy of the DARPA Internet Protocols", SIGCOMM 1988]

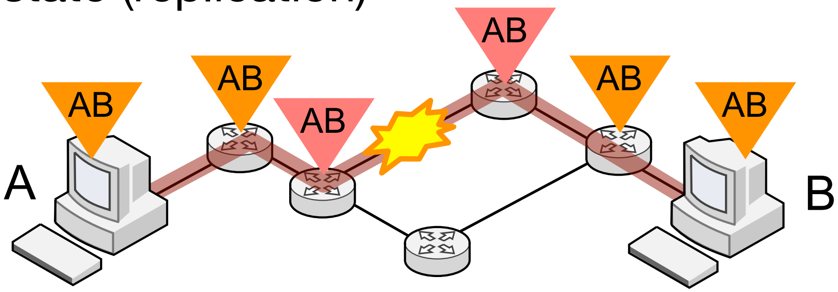
Internet design goals (#1)

- **#1 Survivability in case of failures**
  - On the transport layer, communication should continue despite transient failures
  - ... i.e. on top of the transport layer there are no connection failures other than network partition
- **Protect state information,** do not distribute!
  - State of end-to-end connections is only stored at the endpoints
  - “fate-sharing model” - lose the entity, lose the state

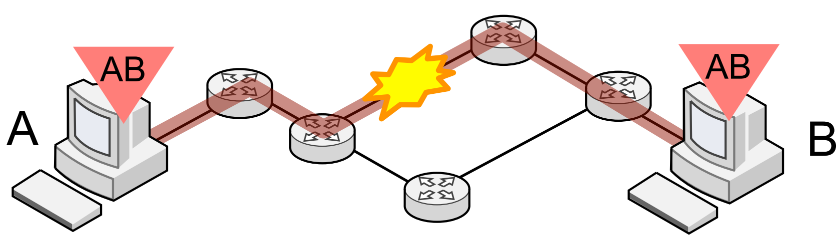
[D. Clark “The Design Philosophy of the DARPA Internet Protocols”, 1988]

End-to-end vs. distributed state

- Distributed state (replication)



- End-to-end communication state (fate sharing)
  - No state information is stored *in* the network
  - State is lost only if the endpoint fails



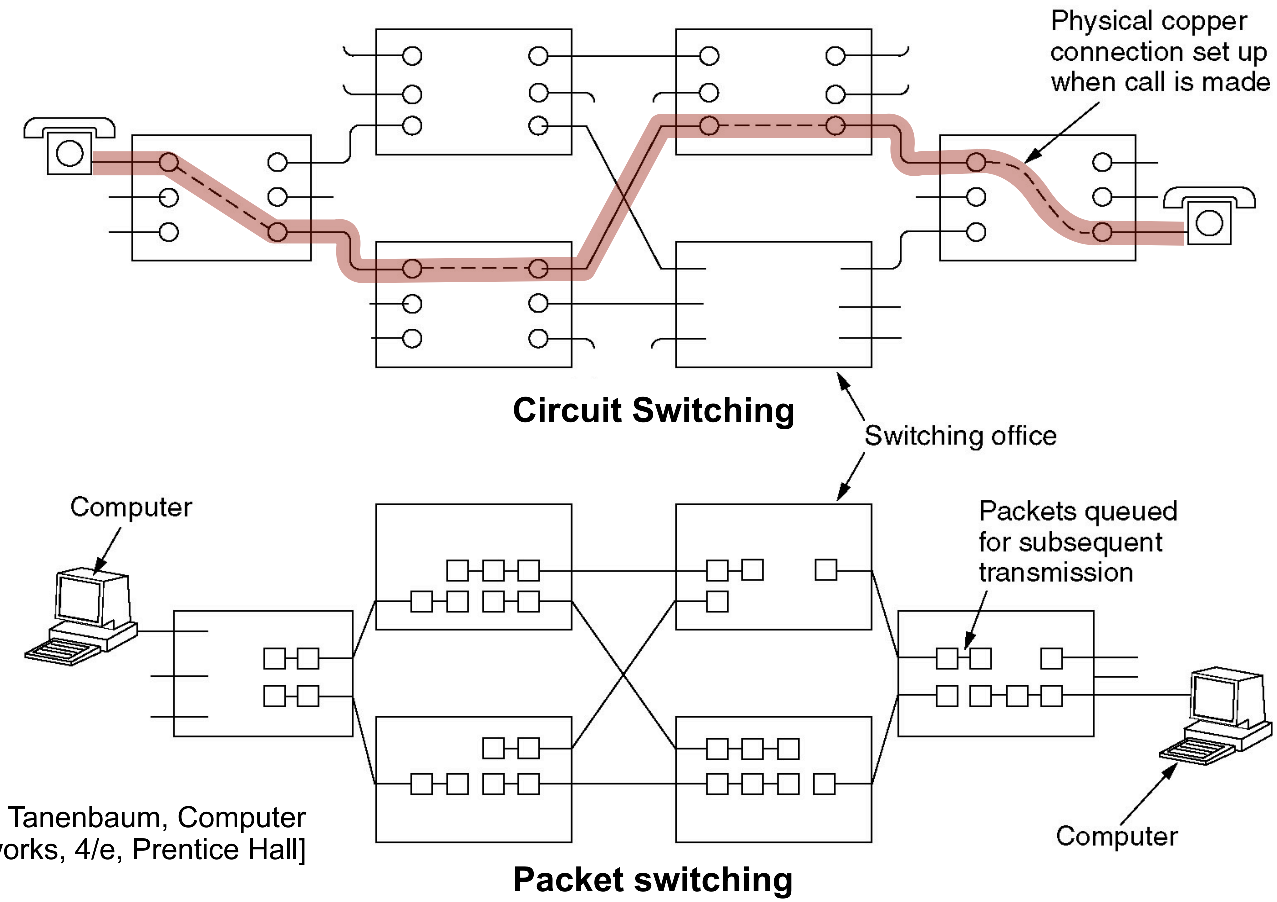
End-to-end vs. distributed state

- Distributed state (replication) prone to failures

- Advantages of the end-to-end (fate sharing) principle:
  - protection from any intermediate failure
  - easier to design and implement
  - allows network interconnection with few assumptions
- Consequence: packet switching better than virtual circuits
- Drawback: end-to-end error correction not always efficient

[B. Carpenter "Architectural Principles of the Internet", RFC1958, 1996]

Packet switching vs. Circuit switching

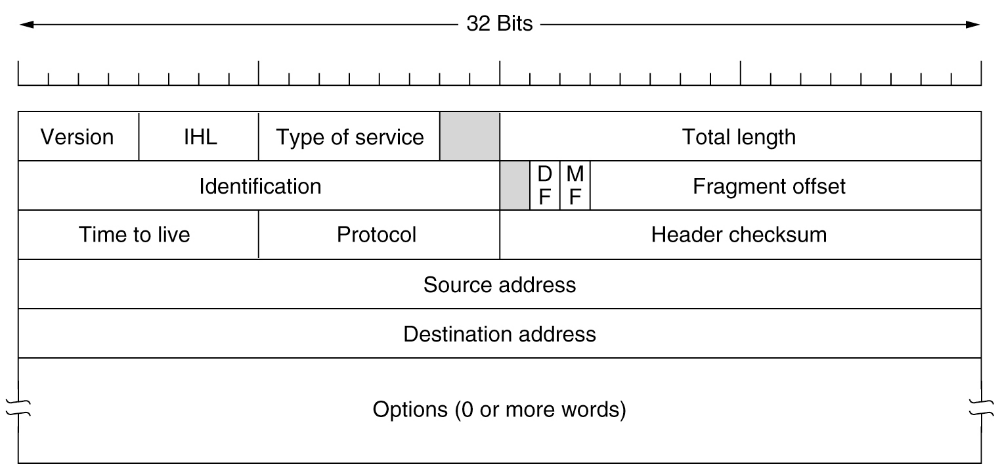


[A.S. Tanenbaum, Computer Networks, 4/e, Prentice Hall]

Packet switching

- **Packets** as data carriers
  - basic structure: meta-data (header) + payload
  - meta-data is *self-descriptive*

- **Example: IPv4 data format**



Packet switching

- **Store and forward communication**
  - less expensive to operate
- **Problems**:
  - uncertain delay
  - switches and routers need buffering capacities
  - packet loss in case of buffer overflow
    (if links are used simultaneously)

[Keshav 1997]

Internet design goals (#2)

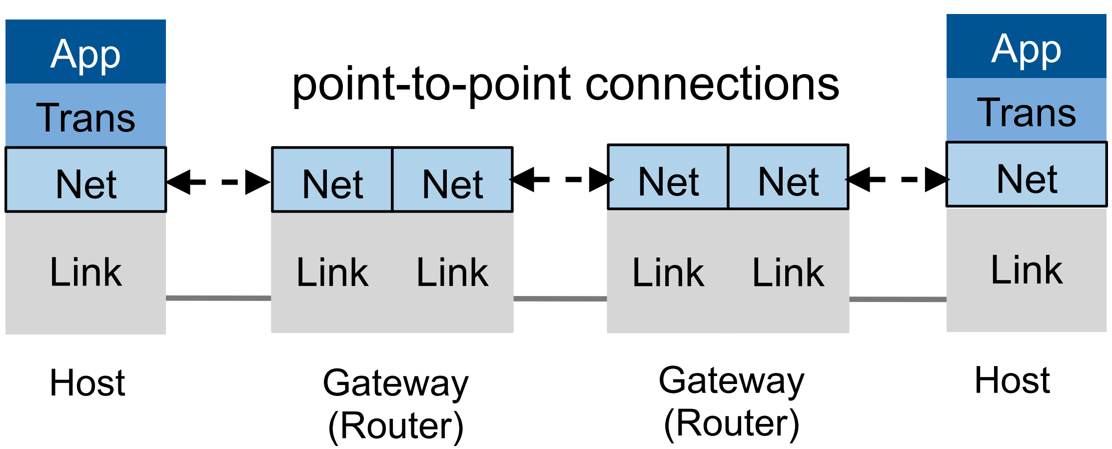
- **#2 Support of various transport services**, e.g.
  - remote login (requires low delay)
  - file transfer (requires high throughput)
  - other services that do not fit TCP

- **Multiple transport services**
  - consequence: Modularity by **protocol layering**
  - separation of TCP and IP
  - IP provides a basic datagram delivery service,
    TCP a reliable data stream on top of IP

[D. Clark “The Design Philosophy of the DARPA Internet Protocols”, 1988]

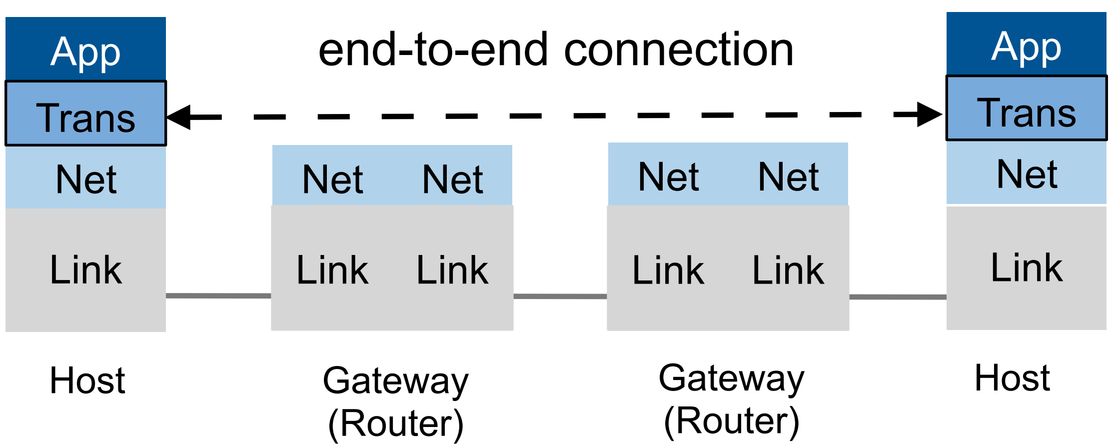
IP Datagram Service

- **IP (Internet Protocol)**
  - delivery of datagrams
  - connection-less and unreliable
  - building block for higher layer services



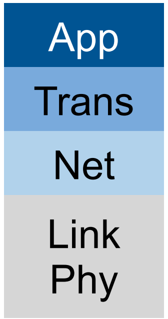
Transport Services

- **TCP (Transmission Control Protocol)**
  - connection-oriented
  - delivers a stream of bytes
  - reliable and ordered
- **UDP (User Datagram Protocol)**
  - application layer interface to IP



Internet Protocol Layers

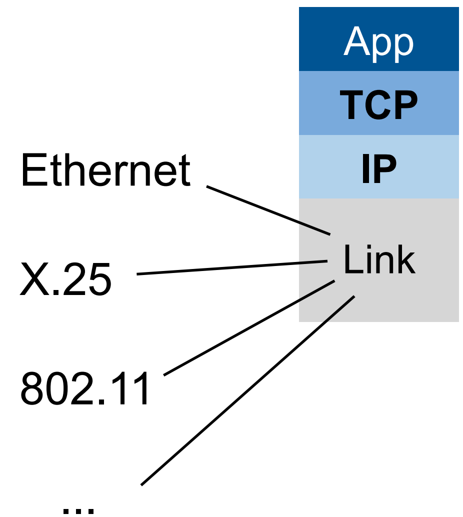
- **Why separating TCP and IP?**
  - routing requires hop-by-hop actions
  - flow control is better done end-to-end
- **...and why do we need more layers?**
  - TCP and IP have to be independent from the actual links
  - 5 layers are sufficient, as session and presentation layer tasks can be provided by the application



[Keshav 1997]

TCP/IP Architecture

- The TCP/IP architecture should enable various services **without support from the underlying networks**
- **Problem:** underlying networks are designed for a certain type of service and not flexible enough
- Internet is operational though



[D. Clark "The Design Philosophy of the DARPA Internet Protocols", 1988]

Internet design goals (#3)

- **#3 Variety of networks**
  - Interconnection of long distance networks, local area networks, satellite connections, packet radio networks
  - large range of bandwidths
  - requires flexibility

- **Minimum set of assumptions**
  - requirement: network can transport a datagram/packet
  - reasonable packet size, reasonable reliability
  - suitable addressing scheme

[D. Clark "The Design Philosophy of the DARPA Internet Protocols", 1988]

Internet design goals (#3, contd.)

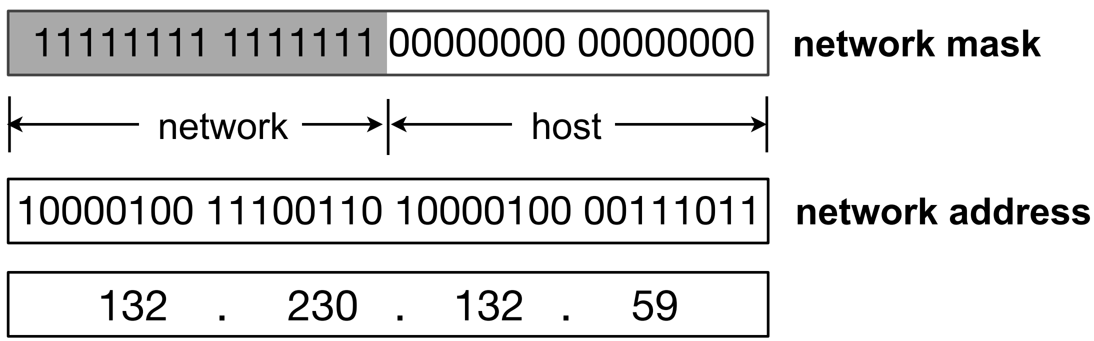
- **What is *not* assumed from a network:**
  - reliable delivery
  - sequenced delivery
  - broadcast or multicast
  - priorities
  - knowledge of failures, speed, delay

- **Consequence**:
  - Services have to be provided by higher layers
  - Interfaces to subnetworks remain simple

[D. Clark “The Design Philosophy of the DARPA Internet Protocols”, 1988]

Addressing

- One address per host interface (not per endpoint)
- Internet addresses used by IP
- IP addresses with 2-part hierarchy to address networks and hosts inside the networks:



(class B address)

- IP addresses in packets remain unchanged

Internet design goals (#4-7)

- **Further design goals not always met effectively**
  - distributed management is possible and takes place, but not always effectively (conflicting routing policies)
  - not always cost- or resource-efficient, e.g.
    - remote login, only few characters per IP packet
    - end-to-end retransmission after packet loss
  - host attachment requires implementation effort
  - accountability was discussed but not considered in the original design

[D. Clark "The Design Philosophy of the DARPA Internet Protocols", 1988]

TCP/IP Review

- **Implemented design principles**:
  Simplicity and modularity by separating datagram (IP) and transport service (TCP)

- **Problems and challenges**
  - end-to-end retransmissions inefficient
    trade-off between efficiency and survivability (goal #1)
  - lack of end-to-end flow control led to the congestion problem in 1986
  - routing can be non-optimal (result of decentralized control, goal #4)
  - IP address shortage and the NAT problem

IP address shortage

- **Scalability problem**
  - short term solutions: variable length subnet masking by CIDR (Classless Inter-Domain Routing), **NAT**
  - long-term solution: IPv6

IPv4 address classes (RFC 791)

only octets

Class	**Leading bits**	**Network address length**	**Host address length**	**Number of networks**	**Number of hosts**
Class A	0	8	24	128	16 777 214
Class B	10	16	16	16,384	65,534
Class C	110	24	8	2 097 152	254
Class D	1110	*undefined*			
Class E	1111	*undefined*			

Violation of Design Principles

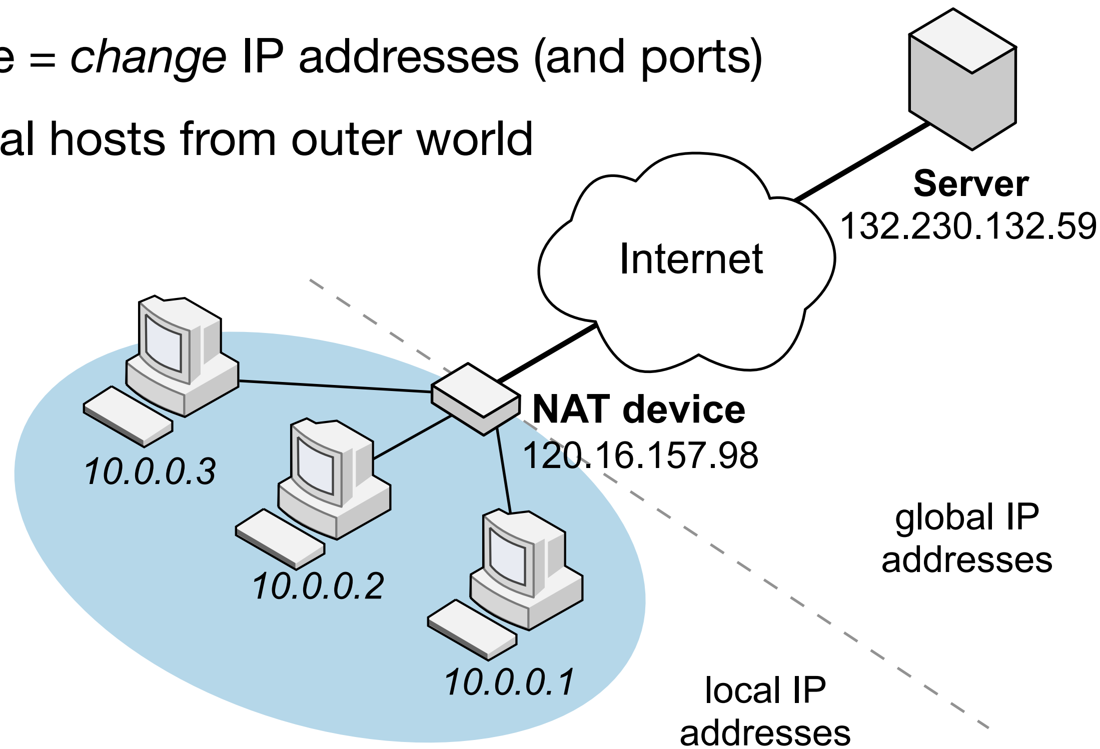
- **NAT (Network Address Translation)**
  - support local IP address assignment
    (solve IP address shortage problem)
  - security feature
  - modification of addresses and ports

- **Firewalls**
  - protection of sub-networks

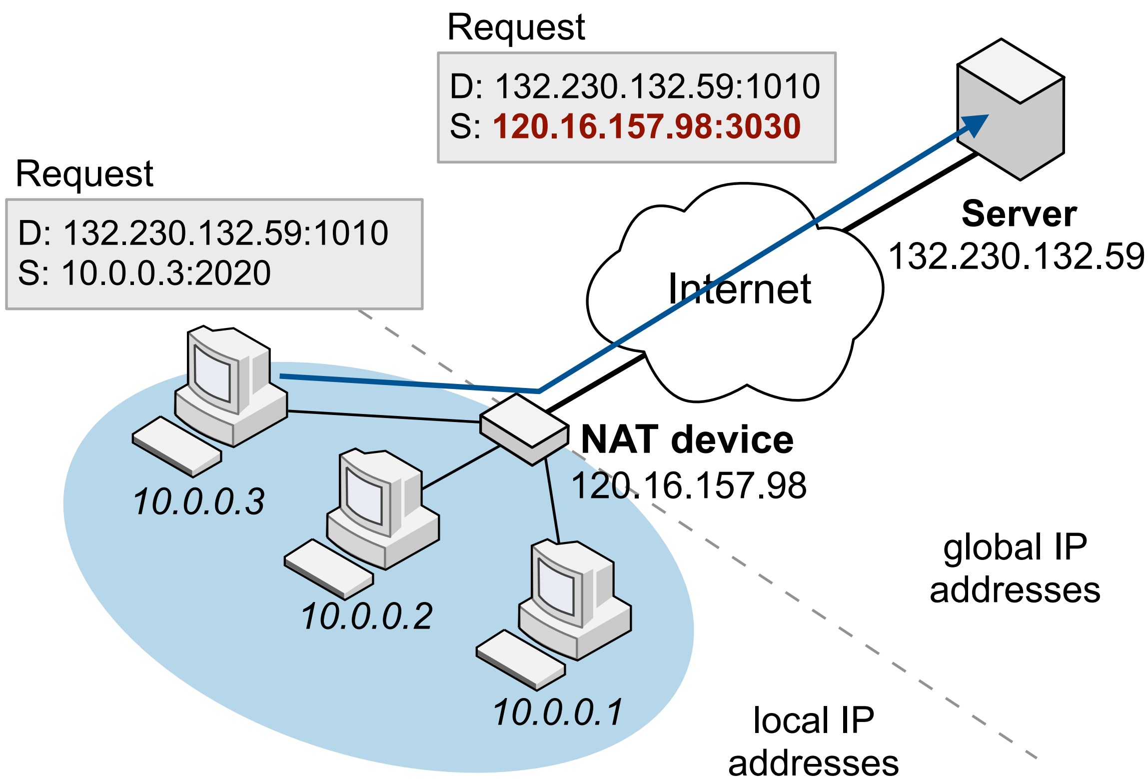
- **both violate the end-to-end principle**

Violation of Design Principles: NAT

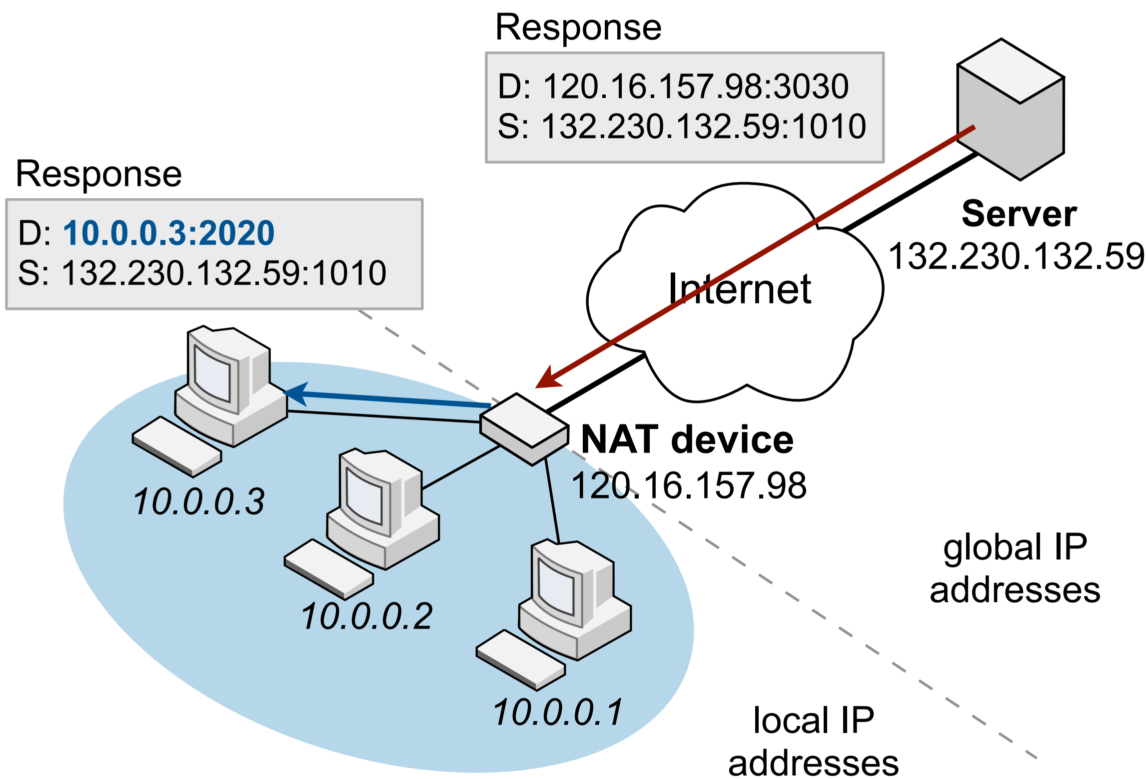
- **NA(P)T devices**
  - translate = *change* IP addresses (and ports)
  - hide local hosts from outer world



NAT Example (1)

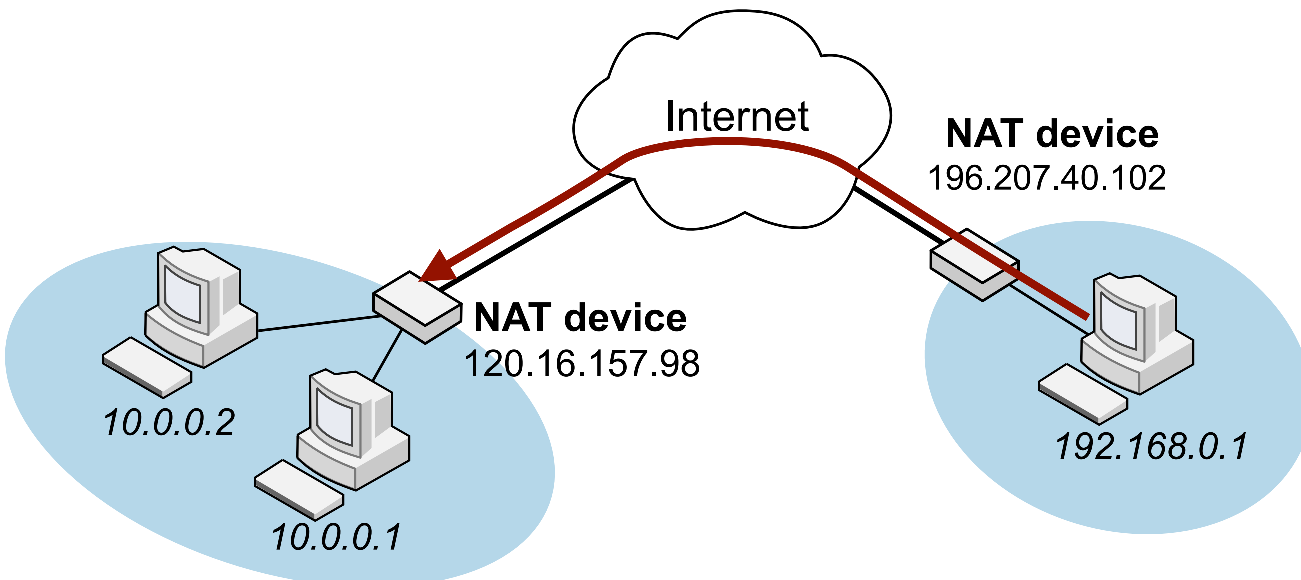


NAT Example (2)



NAT Problem

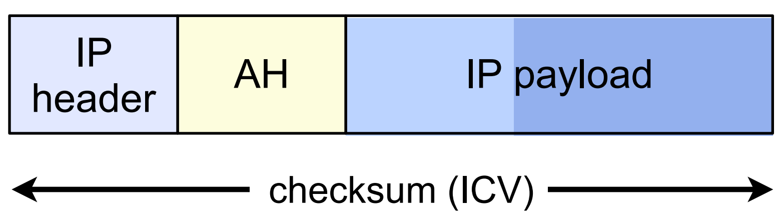
- **How to address a host behind a NA(P)T device?**



NAT and IPsec

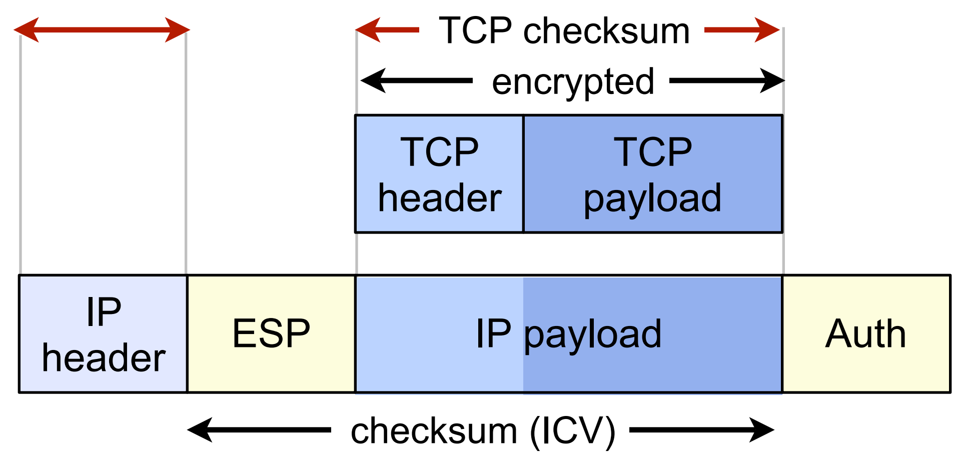
- **IPsec - Layer 3 security** (RFC 4301)
  - encryption and authentication of IP packets

- **Authentication Header**
  - Integrity check value: checksum over the full IP packet including address fields.
  - Altering the address fields by NAT invalidates the packet



NAT and IPsec

- **Encapsulating Security Payload**
  - integrity check value covers payload, but not IP header
  - problem: TCP checksum covers IP address
  - altering TCP checksum not possible or violates ICV



Violation of Design Principles: NAT

- Implications:
  - hosts are not any longer directly addressable
  - shift towards a client-server model
  - workarounds needed for P2P communication (“hole punching”)
  - IPsec fails (NAT device cannot access data or corrupts authentication packets)

- *“an example of an ‘effective’ solution to a point problem greatly restricting generality and future usability”*
  [Braden et al. “Developing a Next-Generation Internet Architecture”, 2000]

Lessons learned

- Layering is an effective concept for modularization
- End-to-end (fate sharing) principle ensures robustness
- Design goals can be conflicting
  - then trade-offs are unavoidable
    (there is no all-purpose all-in-one solution)
- Punctual solutions may have great negative impacts