Transport Layer

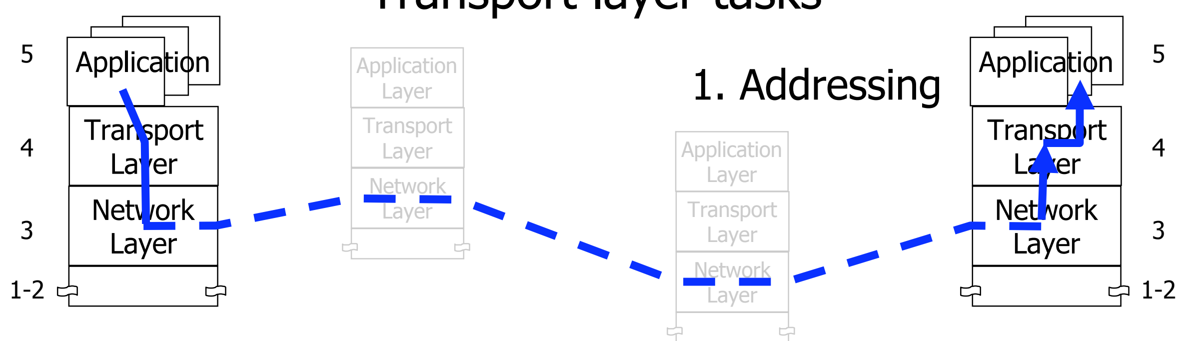
INF3190 / INF4190

Foreleser: Carsten Griwodz
Email: griff@ifi.uio.no

Transport Layer Function

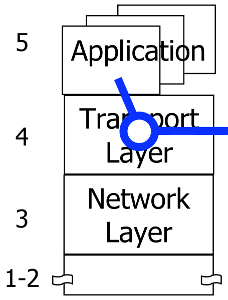
Transport layer tasks

1. Addressing

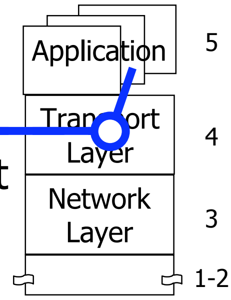


Transport Layer Function

Transport layer tasks

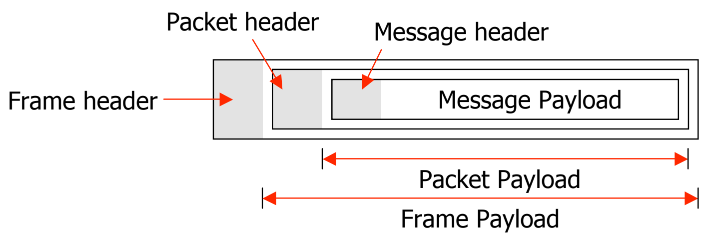


1. Addressing
2. End-to-end connection management
3. Transparent data transfer between end points
4. Quality of service
   - Error recovery
   - Reliability
   - Flow control
   - Congestion control



Transport Service: Terminology

- Nesting of messages, packets, and frames



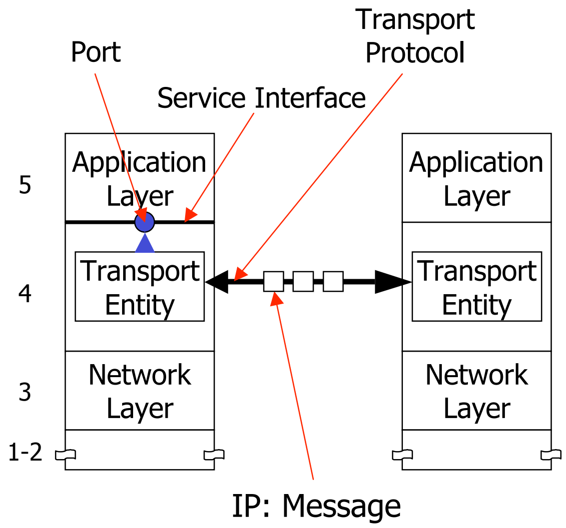
Layer	Data Unit
Transport	Message
Network	Packet
Data link	Frame
Physical	Bit/byte (bitstream)

TCP/IP Message

ISO TPDU (transport protocol data unit)

Transport Service

- Connection oriented service
  - 3 phases
    - connection set-up
    - data transfer
    - disconnect
- Connectionless service
  - Transfer of independent messages
- Realization: transport entity
  - Software and/or hardware
  - Software part usually contained within the kernel (process, library)



TCP/IP Port

ISO TSAP (transport service access point)

Transport Service

- Similar services of
  - Network layer and transport layer
  - Why 2 Layers?
- Network service
  - Not to be self-governed or influenced by the user
  - Independent from application & user
    - enables compatibility between applications
  - Provides for example
    - “only” connection oriented communications
    - or “only” unreliable data transfer
- Transport service
  - To improve the network services that users and higher layers want to get from the network layer, e.g.
    - reliable service
    - necessary time guarantees

Transport Service

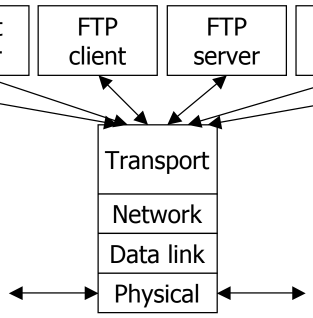
- Transport layer
  - Isolates upper layers from technology, design and imperfections of subnet
- Traditionally distinction made in TCP/IP between
  - Layers 1 – 4
    - transport service provider
  - Layers above 4
    - transport service user
- Transport layer has key role
  - Major boundary between provider and user of reliable data transmission service

Transport Service

- Transport protocols of TCP/IP protocols
  - Services provided implicitly (ISO protocols offer more choice)

	UDP	DCCP	TCP	SCTP
Connection-oriented service		X	X	X
Connectionless service	X			
Ordered			X	X
Partially Ordered				X
Unordered	X	X		X
Reliable			X	X
Partially Reliable				X
Unreliable	X	X		X
With congestion control		X	X	X
Without congestion control	X			
Multicast support	X	X		
Multihoming support				X

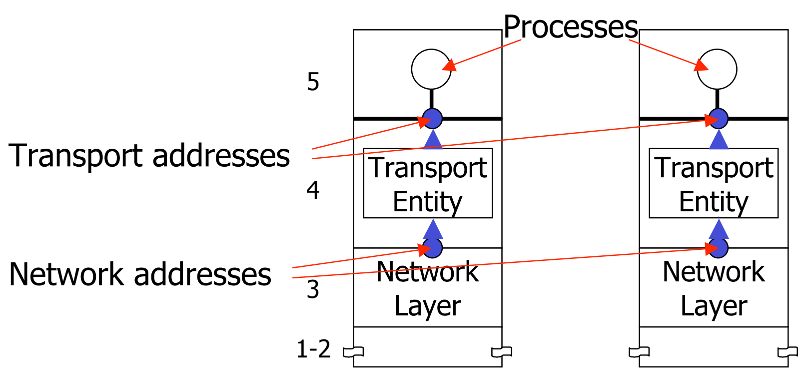
Addressing at the Transport Layer



- Applications …
  - … require communication
  - … communicate
    - locally by interprocess communication
    - between systems via transport services
- Transport layer
  - Interprocess communication via communication networks
- Internet Protocol IP
  - Enables endsystem-to-endsystem communication
  - Not application to application

Addressing at the Transport Layer

- Transport address different from network address
  - Sender process must address receiver process
  - Receiver process can be approached by the sender process



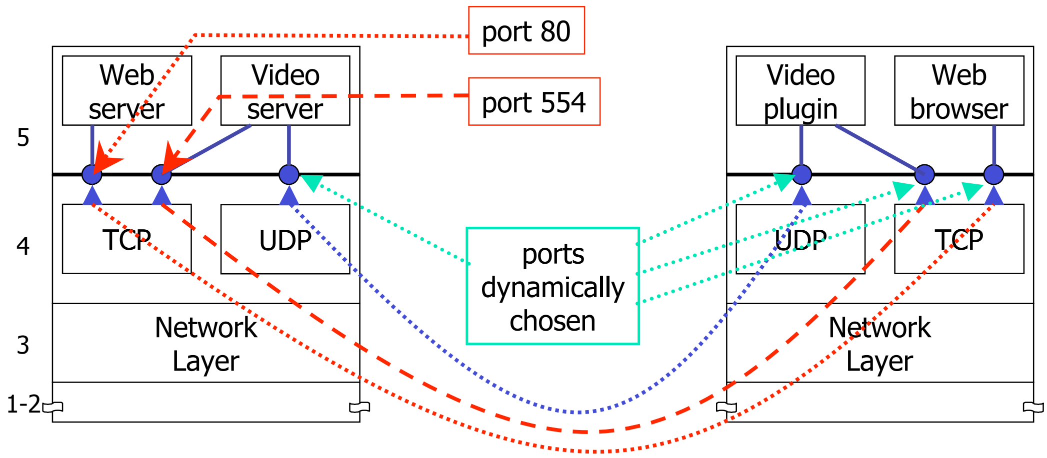
Addressing at the Transport Layer

Decimal	Keyword	UNIX keyword	Description
20	FTP-DATA	ftp-data	File transfer protocol (data)
21	FTP	ftp	File transfer protocol (control)
22	SSH	ssh	Secure shell
23	TELNET	telnet	Terminal Connections
25	SMTP	smtp	Simple mail transfer protocol
37	TIME	time	Time
42	WINS	name	Windows Internet Naming Service
53	DOMAIN	nameserver	Domain Name System
80	HTTP	HTTP	World Wide Web
110	POP3	pop3	Remote Email Access
111	SUN RPC	sunrpc	SUN Remote Procedure Call
119	NNTP	nntp	USENET News Transfer Protocol

- TCP and UDP have their own assignments
  - this table shows some examples for TCP (read /etc/services for more)

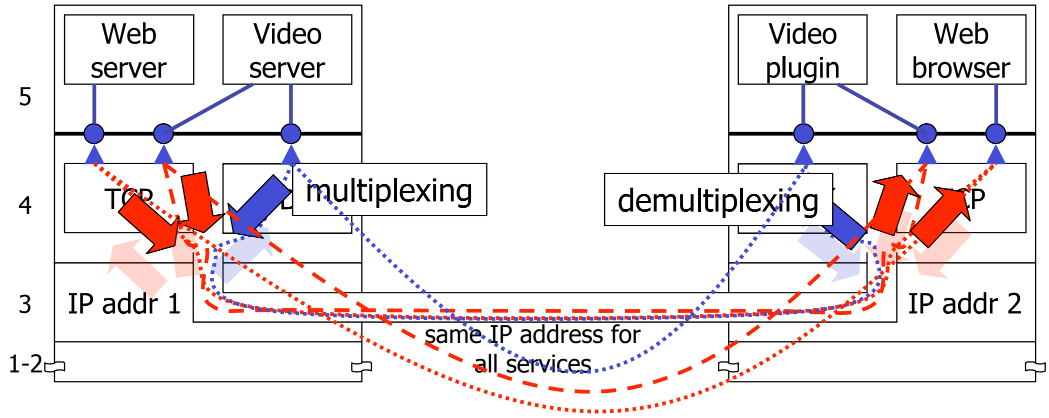
Multiplexing task of the Transport Layer

- Multiplexing and demultiplexing task of the transport layer
- Example: accessing a web page with video element
  - Three protocols used (minor simplification)
    - HTTP for web page
    - RTSP for video control
    - RTP for video data



Multiplexing task of the Transport Layer

- Multiplexing and demultiplexing task of the transport layer
- Example: accessing a web page with video element
  - Three protocols used (minor simplification)
    - HTTP for web page
    - RTSP for video control
    - RTP for video data



Transport Service

- Transport protocols of TCP/IP protocols
  - Services provided implicitly (ISO protocols offer more choice)

	UDP	DCCP	TCP	SCTP
Connection-oriented service		X	X	X
Connectionless service	X			
Ordered			X	X
Partially Ordered				X
Unordered	X	X		X
Reliable			X	X
Partially Reliable				X
Unreliable	X	X		X
With congestion control		X	X	X
Without congestion control	X			
Multicast support	X	X		
Multihoming support				X

Transport Protocols: UDP

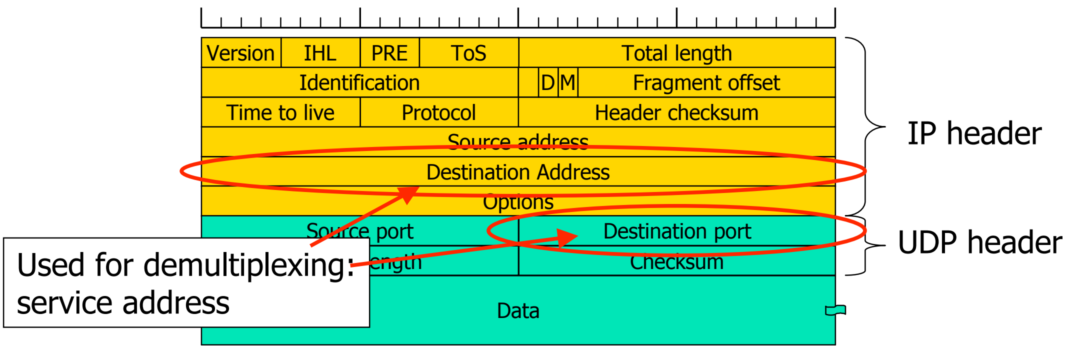
UDP - User Datagram Protocol

- History: IEN 88 (1979), RFC 768 (1980), STD 6
- UDP is a simple transport protocol
  - Unreliable
  - Connectionless
  - Message-oriented
- UDP is mostly IP with short transport header
  - De-/multiplexing
  - Source and destination port
  - Ports allow for dispatching of messages to receiver process

UDP Characteristics

- No flow control
  - Application may transmit as fast as it can / wants and its network card permits
  - Does not care about the network's capacity
- No error control or retransmission
  - No guarantee about packet sequencing
  - Packet delivery to receiver not ensured
  - Possibility of duplicated packets
- May be used with broadcast / multicasting and streaming

UDP: Message Format



- Source port
  - **Optional**
  - 16 bit sender identification
  - Response may be sent there
- Destination port
  - Receiver identification
- Packet length
  - In byte (including UDP header)
  - Minimum: 8 (byte)
    i.e. header without data
- Checksum
  - **Optional** in IPv4
  - Checksum of header and data for error detection

UDP: Message Format – Checksum

- Purpose
  - Error detection (header and data)

- UDP checksum includes
  - UDP header (checksum field initially set to 0)
  - Data
  - Pseudoheader
    - Part of IP header
      - source IP address
      - destination IP address
      - Protocol
      - length of (UDP) data
    - Allows to detect misdelivered UDP messages

Source address		
Destination Address		
00000000	Protocol=17	UDP segment length

- Use of checksum optional
  - i.e., if checksum contains only "0"s, it is not used
    - Transmit 0xFFFF if calculated checksum is 0

UDP: Ranges of Application

- Suitable
  - For simple client-server interactions, i.e. typically
    - 1 request packet from client to server
    - 1 response packet from server to client
  - When delay is worse than packet loss and duplication
    - Video conferencing
    - IP telephony
    - Gaming

- Used by e.g.
  - DNS: Domain Name Service [1]
  - SNMP: Simple Network Management Protocol
  - BOOTP: Bootstrap protocol
  - TFTP: Trivial File Transfer Protocol
  - NFS: Network File System [1]
  - NTP: Network Time Protocol [1]
  - RTP: Real-time Transport Protocol [1]

[1] can also be used with TCP

Transport Protocols: TCP

TCP - Transmission Control Protocol

- TCP is the main transport protocol of the Internet
- History: IEN 112 (1979), RFC 793 (1981), STD 7
- Motivation: network with connectionless service
  - Packets and messages may be
    - duplicated, in wrong order, faulty
    - i.e., with such service only, each application would have to provide recovery
      - error detection and correction
  - Network or service can
    - impose packet length
    - define additional requirements to optimize data transmission
    - i.e., application would have to be adapted
- TCP provides
  - Reliable end-to-end byte stream over an unreliable network service

What is TCP?

Transport protocol specification

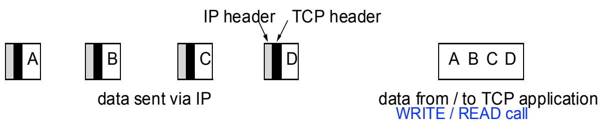
~~Transport protocol implementation~~

- TCP specifies
  - Data and control information formats
  - Procedures for
    - flow control
    - error detection and correction
    - connect and disconnect
  - As a primary abstraction
    - a connection
    - not just the relationships of ports (as a queue, like UDP)

- TCP does not specify
  - The interface to the application (sockets, streams)
  - Interfaces are specified separately: e.g. Berkeley Socket API, WINSOCK

TCP Characteristics

- Data stream oriented
  - TCP transfers serial byte stream
  - Maintains sequential order
- Unstructured byte stream
  - Application often has to transmit more structured data
  - TCP does not support such groupings into (higher) structures within byte stream
- Buffered data transmission
  - Byte stream not message stream: message boundaries are not preserved
    - no way for receiver to detect the unit(s) in which data were written



  - For transmission the sequential data stream is
    - Divided into segments
    - Delayed if necessary (to collect data)

TCP Characteristics

- Virtual connection
  - Connection established between communication parties before data transmission
- Two-way communications (fully duplex)
  - Data may be transmitted simultaneously in both directions over a TCP connection
- Point-to-point
  - Each connection has exactly two endpoints
- Reliable
  - Fully ordered, fully reliable
    - Sequence maintained
    - No data loss, no duplicates, no modified data

TCP Characteristics

- Error detection
  - Through checksum
- Piggybacking
  - Control information and data can be transmitted within the same segment
- Urgent flag
  - Send and transfer data to application immediately
    - example <Crtl C>
      arrival interrupts receiver's application
  - Deliver to receiver's application before data that was sent earlier

TCP Characteristics

- No broadcast
  - No possibility to address all applications
  - With connect, however, not necessarily sensible
- No multicasting
  - Group addressing not possible
- No QoS parameters
  - Not suited for different media characteristics
- No real-time support
  - No correct treatment / communications of audio or video possible
  - E.g. no forward error correction

TCP in Use & Application Areas

Benefits of TCP

- Reliable data transmission
  - Efficient data transmission despite complexity
  - Can be used with LAN and WAN for
    - low data rates (e.g. interactive terminal) and
    - high data rates (e.g. file transfer)

Disadvantages when compared with UDP

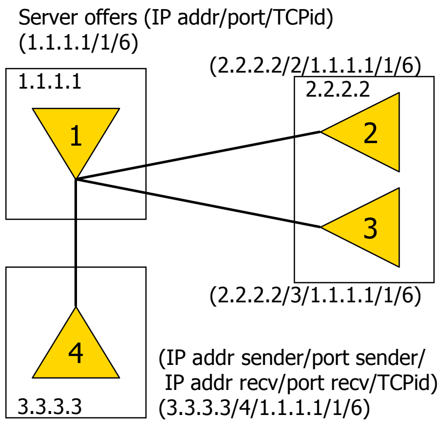
- Higher resource requirements
  - buffering, status information, timer usage
- Connection set-up and disconnect necessary
  - even with short data transmissions

Applications

- File transfer (FTP)
- Interactive terminal (Telnet)
- E-mail (SMTP)
- X-Windows

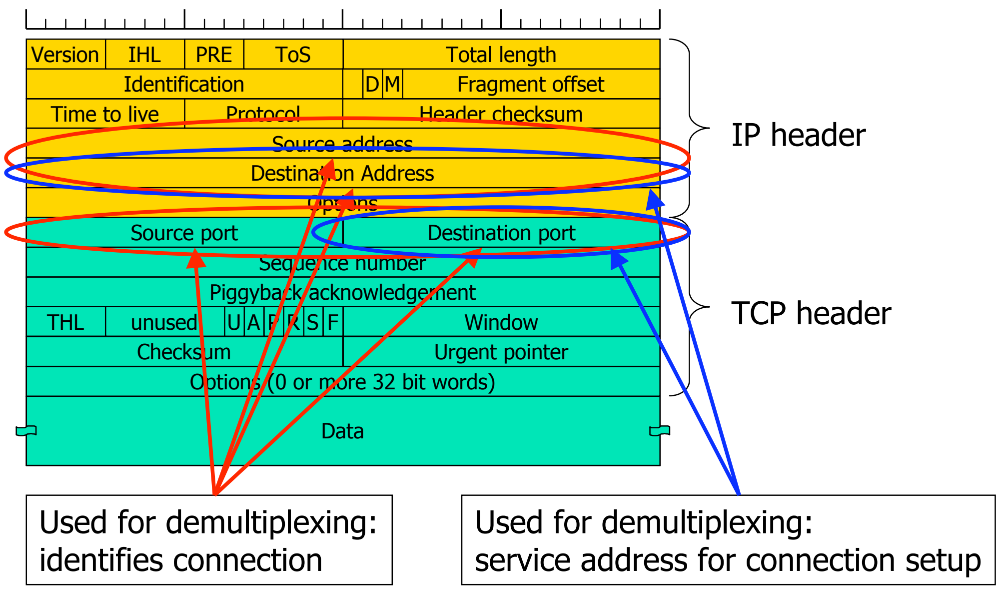
Connection – Addressing

- TCP service obtained via service endpoints on sender and receiver
  - Typically socket
  - Socket number consists of
    - IP address of host and
    - 16-bit local number (port)
- Transport Service Access Point
  - Port
- TCP connection is clearly defined by a ***quintuple*** consisting of
  - IP address of sender and receiver
  - Port address of sender and receiver
  - TCP protocol identifier
- Applications can use the same local ports for several connections



TCP: Message Format

- TCP/IP Header Format

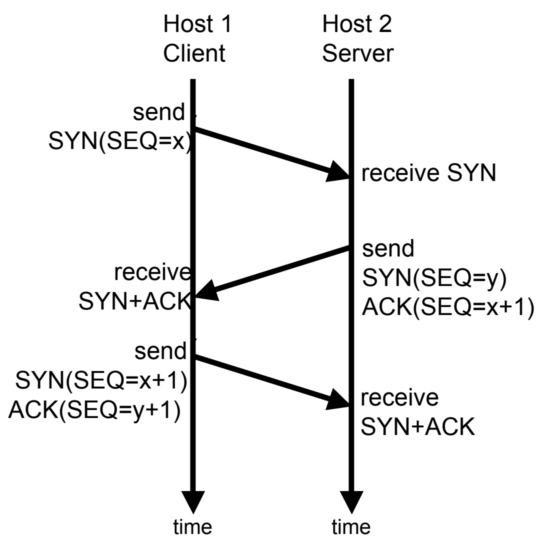


Transport Protocols

Connection Establishment: TCP

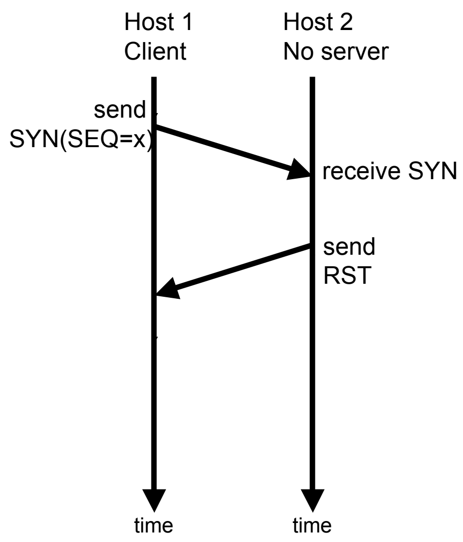
Connection Establishment

- One passive & one active side
  - Server: wait for incoming connection using LISTEN and ACCEPT
  - Client: CONNECT (specifying IP addr. and port, max. TCP segment size)
- Three-Way-Handshake
  - Connecting through 3 packets



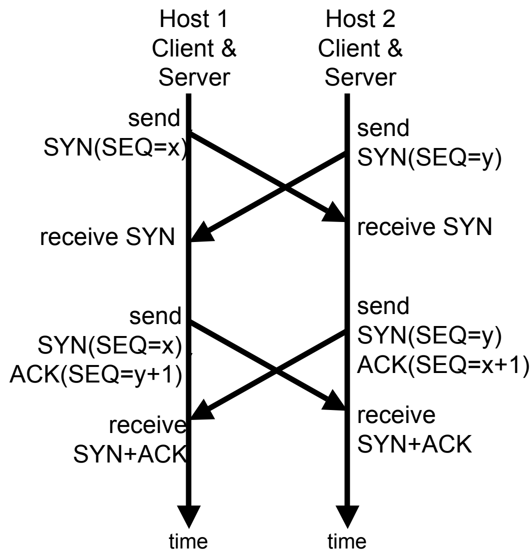
Connection Establishment

- If on server side no process is waiting on port (no process did LISTEN)
  - Reply packet with RST bit set is sent to reject connection attempt
- Process listening on port may accept or reject



Connection Establishment

- Call collision
  - Still only one single connection will be established even when
    - both partners actively try to establish a connection simultaneously

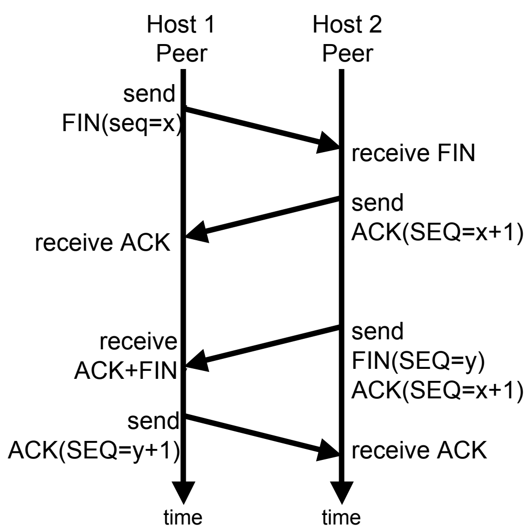


Connection Release

- Connection release for pairs of simplex connections
  - each direction is released independently of the other
- Connection release by either side sending a segment with FIN bit set
  - no more data to be transmitted
  - when FIN is acknowledged, this direction is shut down for new data
- Directions are released independently
  - other direction may still be open
  - full release of connection if both directions have been shut down

Connection Release

- Systematic disconnect by 4 packets
  - between 2nd and 3rd
    - host 2 can still send data to host 1
- 3 packets possible
  - first ACK and second FIN may be contained in same segment
- Connection interrupt: Opposite side cannot transmit data anymore
  - immediate acknowledgement, release of all resources
  - data in transit may be lost

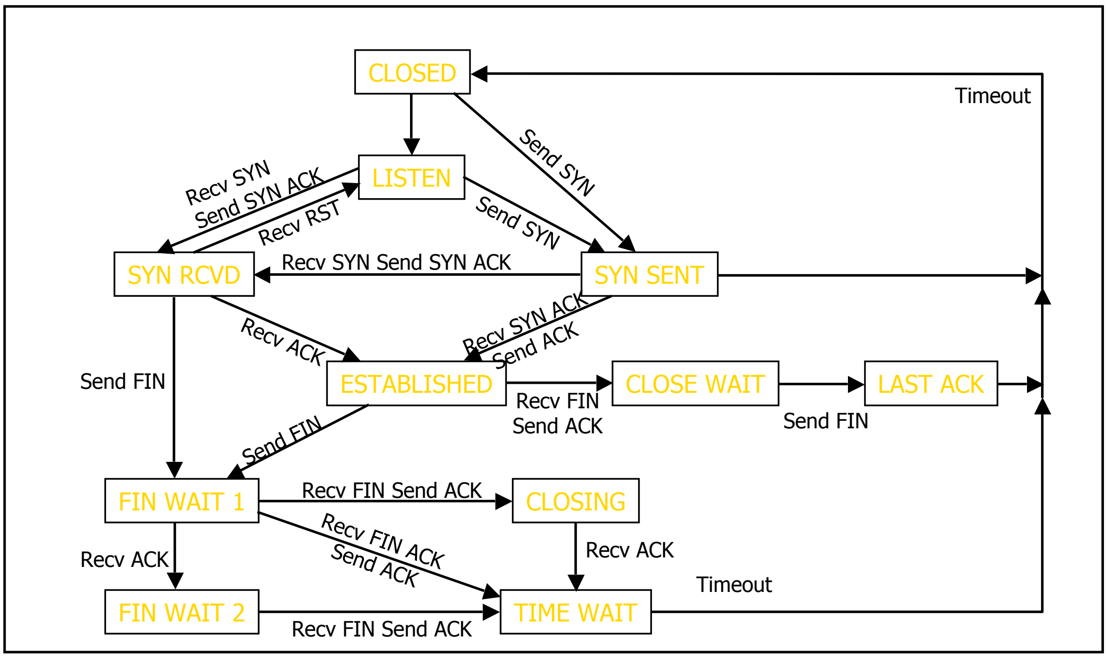


Connection Management Modelling

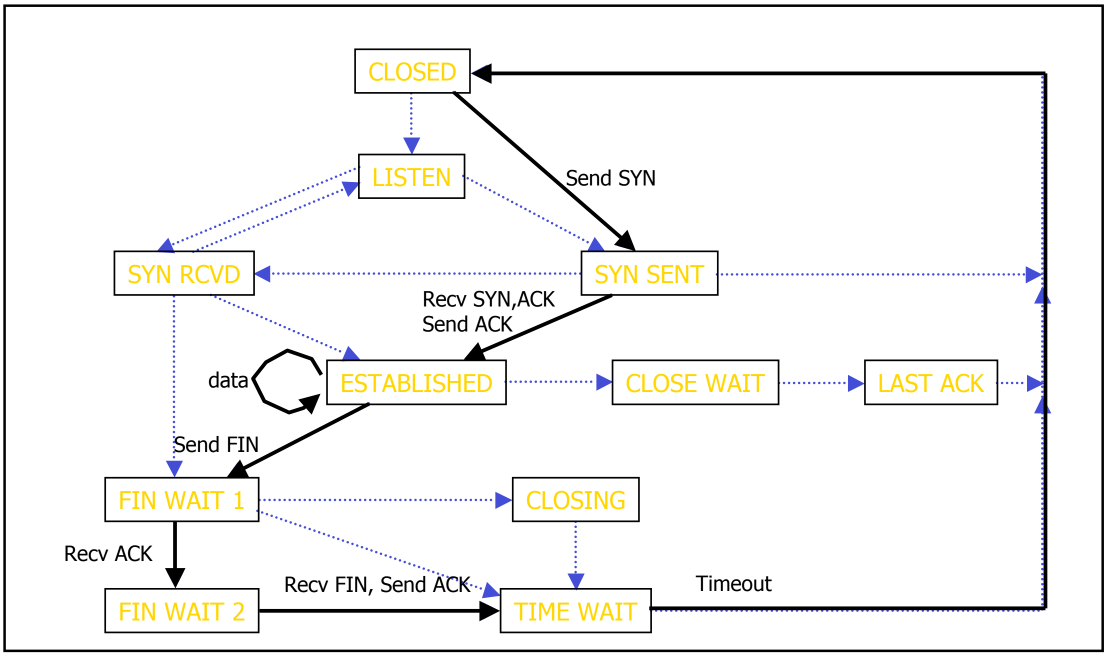
- States

State	Description
CLOSED	No connection is active or pending
LISTEN	Server is waiting for an incoming call
SYN RCVD	Connection request has arrived, wait for ACK
SYN SENT	Application has started to open a connection
ESTABLISHED	Normal data transfer state
FIN WAIT 1	Application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

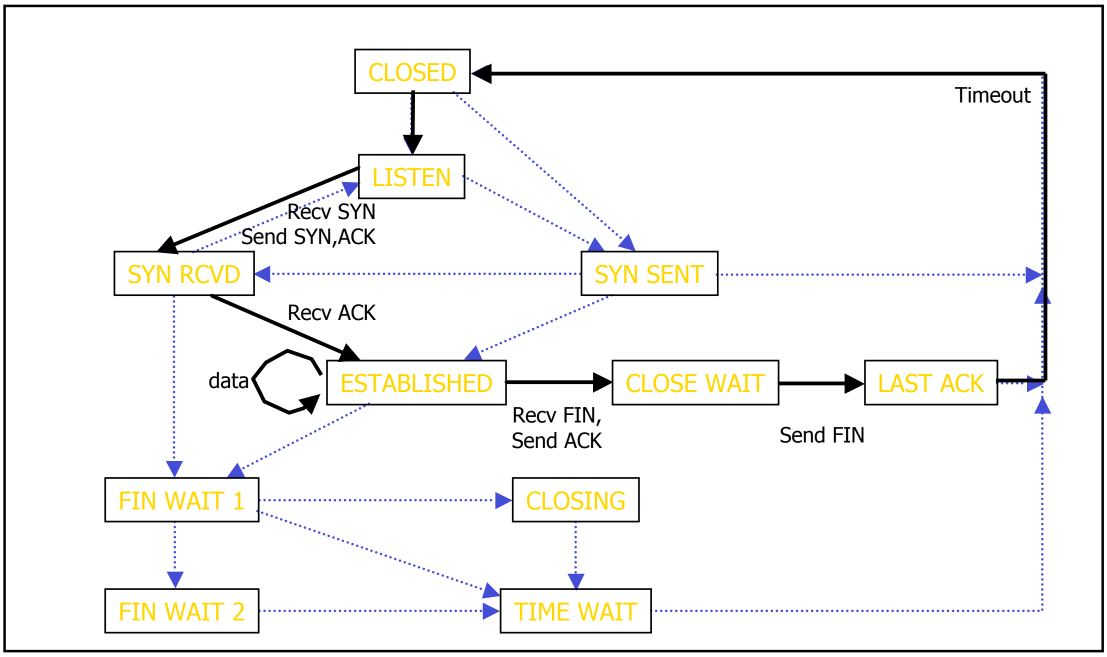
States



Typical State Sequence of a TCP Client



Typical State Sequence of a TCP Server



Transport layer

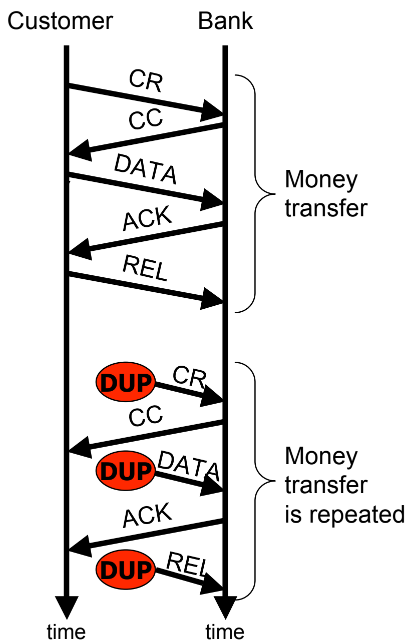
Reliability and Ordering: Generic approaches

Reliability and Ordering

- Transport layer must handle
  - Packet loss
  - Packet duplication
  - Multiplexing and demultiplexing of connections
- Packet loss
  - Retransmission
    - Used with various ACK and NACK schemes
  - Forward error correction
    - Not typically used by the transport layer

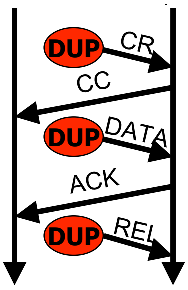
Duplicates

- Initial Situation: Problem
  - Network has
    - Varying transit times for packets
    - Certain loss rate
    - Storage capabilities
  - Packets can be
    - Manipulated
    - Duplicated
    - Resent by the original system after timeout
- In the following, uniform term: "Duplicate"
  - A duplicate originates due to one of the above mentioned reasons and
  - Is at a later (undesired) point in time passed to the receiver



Duplicates

- Possible error causes and consequences
  - Cause
    - Network capabilities
      - Flood-and-prune approach to routing in wireless networks
      - All acknowledgements lost
  - Consequence
    - Duplication of sender's packets
    - Duplicates arrive in the same order as originals
  - Cause
    - Man-in-the-middle attack
      - Packets are captured and replayed
  - Consequence
    - Controlled duplication of sender's packets
    - Duplicates arrive in an order expected by the application



- Result
  - Without additional means
    - Receiver cannot differentiate between correct data and duplicated data
  - Would re-execute the transaction

Duplicates: Problematic Issues

- 3 somehow disjoint problems
  - How to handle duplicates within a connection?

  - What characteristics have to be taken into account regarding …
    - Consecutive connections
      or
    - Connections which are being re-established after a crash?

  - What can be done to ensure that a connection has been established?
    - Has actually been initiated by
      and
    - With the knowledge of both communicating parties?

Duplicates: Methods of Resolution

- Using temporarily valid ports

- Method
  - Port valid for one connection only
  - Generate always new port

- Evaluation
  - In general not applicable:
    process server addressing method not possible, because
    - Server is reached via a designated port
    - Some ports always exist as "well known‟

Duplicates: Methods of Resolution

- Identify connections individually

- Method
  - Each individual connection is assigned a new sequence number and
  - End-systems remember already assigned sequence numbers

- Evaluation
  - End-systems must be capable of storing this information
  - Prerequisite
    - Connection oriented system
  - End-systems, however, will be switched off and
    it is necessary that the information is reliably available whenever needed

Duplicates: Methods of Resolution

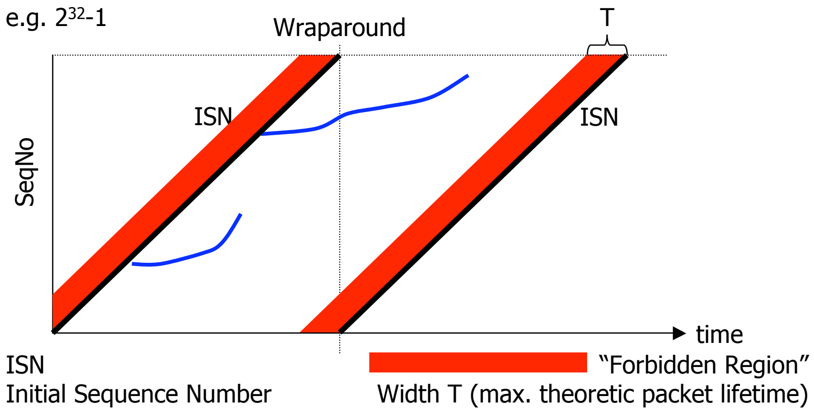
- Identify packets individually
  - Individual sequential numbers for each packet

- Method
  - Sequence number basically never gets reset
  - e.g. 48 bit at 1000 msg/sec: reiteration after ~8930 years

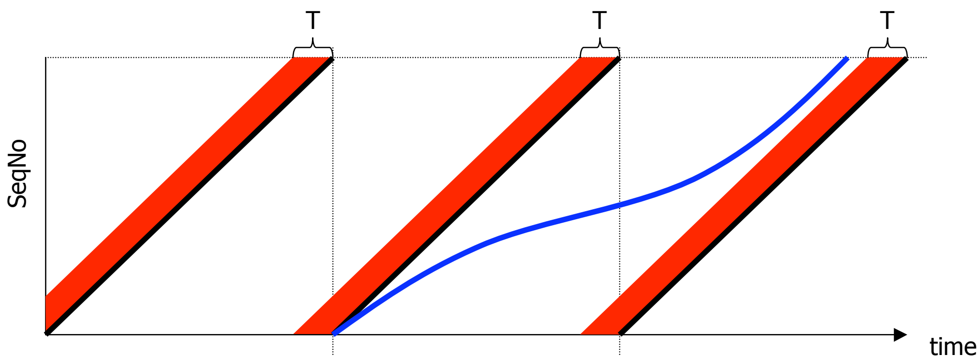
- Evaluation
  - Higher usage of bandwidth and memory
  - Sensible choice of the sequential number range depends on
    - The packet rate
    - A packet's probable "lifetime" within the network
  - Discussed in more detail in the following

Handling of Consecutive Connections

- Method
  - End-systems timer continues to run at switch-off / system crash
  - Allocation of initial sequence number (ISN) depends on
    - time markers (linear or stepwise curve because of discrete time)
  - Sequence numbers can be allocated consecutively within a connection (steadily growing curve)

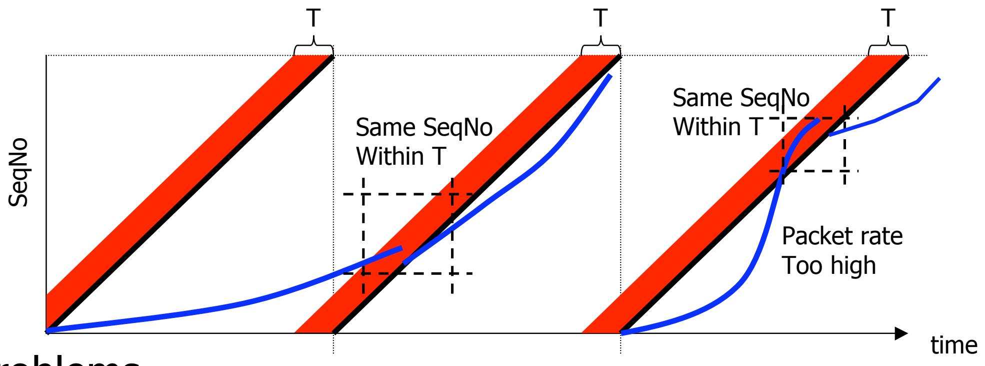


Handling of Consecutive Connections



- No problem, if
  - "Normal lived" session (shorter than wrap-around time) with data rate smaller than ISN rate (ascending curve less steep)
- Then, after crash
  - Reliable continuation of work always ensured
  - System clock may be used to continue with correct ISN

Handling of Consecutive Connections



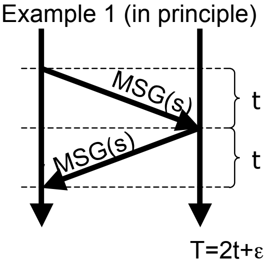
- Problems
  - "Long-lived", "slow" session (longer than wrap-around time)
    - Sequence number is used within time period T before it is used as initial sequence number
    - ⇒ "Forbidden Region" - begins T before ISN is generated
  - High data rate
    - Curve of the consecutively allocated sequence numbers steeper than ISN curve (enters from underneath)

Duplicates: Limiting Packet Lifetime

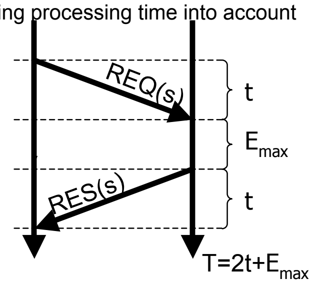
- Enabling the above Method 3
  Identify packets individually: individual sequential numbers for each packet
  - Sequence number only reissued if
    - All packets with this sequence number or references to this sequence number are extinct
  - i.e., ACK (N-ACK) have to be included
    - Otherwise new packet may be wrongfully confirmed or non-confirmed by delayed ACK (N-ACK)

- Mandatory prerequisite for this solution
  - Limited packet lifetime
  - I.e. introduction of a respective parameter *T*

Example 1 (in principle)



Example 2: Request/response
Taking processing time into account



Duplicates: Limiting Packet Lifetime

- Limitation by appropriate network design
  - Inhibit loops
  - Limitation of delays in subsystems & adjacent systems
- Hop-counter / time-to-live in each packet
  - Counts traversed systems
  - If counter exceeds maximum value
    - ⇒ packet is discarded
  - Requirement: known maximal time for one hop
- Time marker in each packet
  - Packet exceeds maximum configurable lifetime
    - ⇒ packet is discarded
  - Requirement: "consistent" network time

Duplicates: Limiting Packet Lifetime

- Determining maximum time T, which a packet may remain in the network
  - T is a small multiple of the (real maximal) packet lifetime t
  - T time units after sending a packet
    - The packet itself is no longer valid
    - All of its (N)ACKs are no longer valid

Transport layer

Reliability and Ordering: TCP

TCP's approach to reliability and ordering

- TCP over IP situation
  - Provide connection-oriented, reliable, ordered transport layer service over
    connectionless, unreliable, unordered network layer service

- TCP's approach
  - Limiting packet lifetime using TTL at IP level
  - Choosing maximum packet lifetime *T*
  - Unique connection identifier
    - (client addr, client port, server address, server port)
    - Reuse limited by packet lifetime
  - Individual sequential numbers per connection

TCP's approach to reliability and ordering

- TCP/IP term: Maximum Segment Lifetime (MSL)
  - 2MSL: two MSLs to wait in TIME_WAIT
  - Solaris 2MSL: 4 minutes default
  - Linux 2MSL: 1 minute
  - Windows 2MSL: 30 seconds default
- Problem with 2MSL for uniquely identifying connections
  - Packets from connections that can not be distinguished
  - Especially: fast restart after crash
- Solution: none

TCP's approach to reliability and ordering

- Problem with sequence numbers per connection
  - 32 bit sequence numbers with technology considered as sufficient when designing TCP/IP
  - Sequence number range exploitation
    - today at 1 Gbps
    - in 17 sec
- Solution: verified sequence number
  - PAWS – RFC1323
    - Use TCP 32-bit timestamp option in each packet
    - "Protect Against Wrapped Sequence Numbers"
    - "TCP extensions for highspeed paths"
    - Reject packet
      - If timestamp is lower than last recorded and sequence number is higher