# Communication Systems
# Transport System - Performance Issues

Prof. Dr.-Ing. **Lars Wolf**

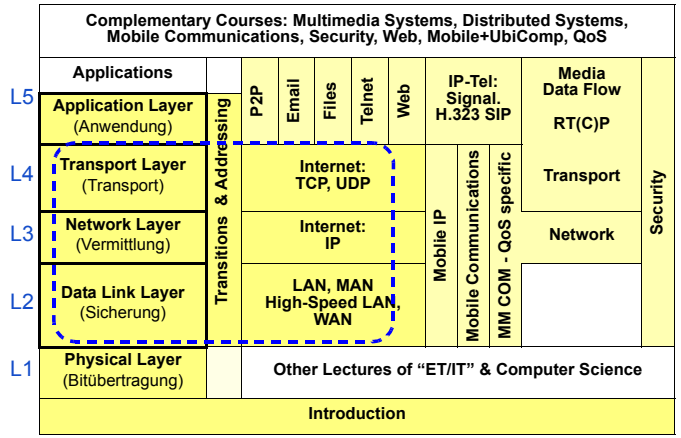
*TU Braunschweig*
*Institut für Betriebssysteme und Rechnerverbund*

*Mühlenpfordtstraße 23, 38106 Braunschweig, Germany*
*Email: wolf@ibr.cs.tu-bs.de*





## Scope




**Performance issues**

- **must be treated at many (all) layers**
- **some common aspects, sometimes even interwoven, across layers**
- **discussion here at L4 since broader issues tend to be more transport related**





## Overview

1. **Performance Issues**
2. **Protocols for High-Performance Networks**





## 1. Performance Issues

**Many aspects can contribute to performance problems**

**Reasons typically difficult to understand**

**Issues from several layers might be intertwined / play a role**

**Some aspects will be studied here**

## Reasons for Performance Problems in Computer Networks

**Some problems, such as congestion, caused by temporary resource overloads**
- **too much traffic at a router**

**Others performance problems due to insufficient resources**
- **e.g., low-end receiver is too slow to process incoming packets fast enough**
  - packets must be retransmitted
  - hence, more delay, waste of bandwidth → worse performance
- **resource allocation not properly balanced (e.g., too few buffers)**

**Synchronous overloads**
- **reply to a malformed TPDU**
  - unfortunately to a broadcast address ... → broadcast storm
- **(re-) boot of machines after power failure**

**Not well-chosen timer values, e.g.**
- **retransmissions too early or too late**
- **wait for piggybacking opportunity**

**Etc.**



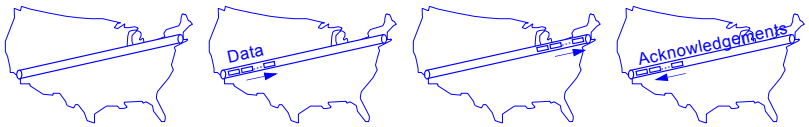


## Bandwidth-Delay Product

***bandwidth [bits/sec] * round-trip delay [sec]***

**Useful quantity for network performance analysis**
- **Capacity of pipe from sender to receiver and back (in bits)**

**Example:**
- **Transmission from San Diego to Boston**
  - sending 64 KB burst (receiver buffer 64 KB)
  - link: 1 Gbps
  - one-way propagation delay (speed-of-light in fiber): 20 msec




  - Bandwidth-delay product: 40 million bit
  - sender would have to transmit burst of 40 million bits to keep pipe busy till ACK

**Receiver window must be >= bandwidth-delay product for good performance**





## Measuring Network Performance

**Basic loop for improving network performance:**
- **measure relevant network parameters and performance**
- **try to understand what is going on**
- **change one parameter**

**repeat until performance is good enough / all improvements have been made**

**Measurements must be made with care:**
- **sample sizes must be large enough**
  - e.g., do not measure time needed to send one TPDU
- **samples must be representative**
  - e.g., measure at different times (during a day, week, ...)
- **clock granularity must be appropriate**
  - e.g., do not try to measure events shorter than 1 msec when clock granularity is 1 msec
- **control the environment and avoid external influences**
  - e.g., use an idle system and create workload yourself
- **...**





## System Design for Better Performance

**Instead of tuning poorly-designed network**
**it is better to have a good design in first place**

**Some rules**
**(partly more related to system design):**

1. CPU speed is more important than network speed
   - mostly: OS and protocol overhead dominate transmission time
2. Reduce packet count to reduce software overhead
   - TPDU processing consists of **FIXED PER-TPDU** plus **VARIABLE PER-BYTE** part
   - interrupts from network card (cache misses, etc.)
3. Minimize context switches
   - cache misses, generally change of context
4. Minimize copying
   - copying kills performance
5. You can buy more bandwidth but no lower delay
   - second fiber doubles bandwidth, but doesn't help with delay
6. Avoiding congestion is better than recovering from it
   - because latter takes time, effort, ...
7. Avoid timeouts
   - better too conservative than too tight timer settings

## Fast TPDU Processing

**Main obstacle for fast networking is protocol software**

**Fast TPDU processing via special treatment of normal case**

- **perform necessary checks at beginning, than use fast path**
- **e.g., when sending data**
  - check that connection is established, enough window-space available, etc.
  - use prototype header (headers of consecutive TPDUs are almost identical)
  - copy as block and set only changed fields

| Source port | Destination port |
|---|---|
| Sequence number | |
| Acknowledgement number | |
| Len / Unused | Window size |
| Checksum | Urgent pointer |

TCP prototype header

| VER. | IHL | TOS | Total length |
|---|---|---|---|
| Identification | | | Fragment offset |
| TTL | Protocol | Header checksum | |
| Source address | | | |
| Destination address | | | |

IP prototype header

**Buffer management**

- **avoid unnecessary copy operations**

**Timer management**

- **many timers must be maintained and most never expire**





## Example: Reduce Packet Count

**Processing effort: per message (packet) number and size dependent**

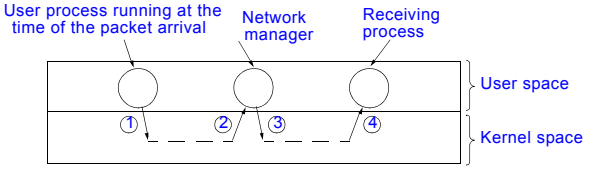
- **per message/packet/frame**
  - interrupt at network controller
  - header (and trailer) processing
- **a lot of processing in SW**
- **usually better if less data units**
  - if necessary at the expense of the delay
  - but e.g. ATM small cells

⇒ REDUCTION OF THE AMOUNT OF DATA UNITS





## Example: Avoid Context Switches




**Protocols and services**

- **realized as parallel processes**
- **structural possibilities**
  - 1 process realizes everything
  - 1 sending process and 1 receiving process per ES
  - 1 sending process and 1 receiving process per transport connection
  - service specific processes
  - 1 or more processes per layer
  - ....

**Always:**

**Context switching consumes considerable processing capacity**

⇒ CAN BE MINIMIZED BY APPROPRIATE PROCESSING STRUCTURE

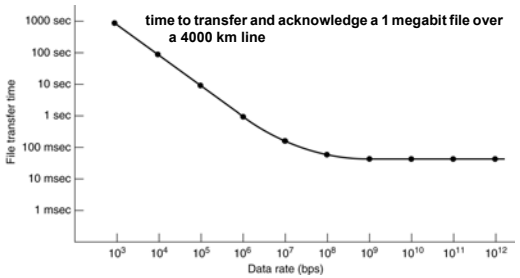




## 2. Protocols for High-Performance Networks

**Several problems came up when old protocols were used for Gigabit networks**

1. **too small (e.g., 32-bit) sequence numbers**
   - with 56 kbps: > 1 week to use up sequence number space
   - with 10 Mbps: 57 minutes to use up sequence number space
   - with 1 Gbps: 34 seconds to use up sequence number space
2. **communication speed has improved faster than computing speed**
3. **'go back n' performs poorly on lines with large bandwidth-delay product**
   - too much time before sender hears about error, retransmit of much data
4. **long gigabit lines are delay limited rather than bandwidth limited**




5. **new (multimedia) appl. are sensible to packet arrival time variance (jitter)**

## 2.1 Approaches

**Basic principle:**
- **Design for speed, not for bandwidth optimization**

**Do not pack small fields in bytes together**

**Hardware-based network interface implementation helps**
- **but only if protocols are simple enough to be put into fast (custom) hardware**

**Avoid feedback loops**

**Packet layout is important**
- **use few and well-aligned fields**

**Checksum header and data separately**

**Use large maximum data size**

**Note: as often, not everybody agrees ...**





## 2.2 Further Transport Protocols

**In 1980s,**
- **'high-speed' networks came up**
- **many people expected that TCP (and similar transport protocols such as TP4) would not be able to perform sufficiently fast**
- **need for new transport protocols was claimed**

⇒ **Various high-speed transport protocols were developed in 80s, early 90s**
- simpler structure
- some designed to better support specific purposes (e.g., request/reply)
- etc.

**Others analyzed and improved TCP implementations**
- **TCP still in use and at high speed**

**Following gives some examples of such developed transport protocols**





## Delta-t

**History**
- **Lawrence Livermore Laboratories, starting at the end of the 70s until '89**

**Objectives**
- **to support the client/server principle**

**Transactions**
- short requests, short responses
- minimizing the number of packets per transaction
- **basic network service**
  - connectionless
  - unreliable

**Media**
- **pure data service**

**Timer-based connection management**
- **previously not contained in OSI TP 4 or TCP**
- **implicit connect**
  - actual first data transmitted with connect
  - avoids delays of the 2-way/3-way-handshakes





## NETBLT

**Network Block Transfer Protocol: history**
- **MIT**
- **since appr. 1978**

**Objectives**
- **transfer high quantities of data**
  - over broadband network connections
- **connections with high end-to-end delays**
  - for example, via satellites
- **implementation in software**

**Previously**
- **here flow control would require**
  - very large windows
  - but then issues critical with regard to congestions

## NETBLT (2)

**Combined rate control and window-based flow control**
- **explicitly permits bursts (i.e. burst rate, burst time)**
  - active period
    - actual data, burst
  - inactive period
    - in this period data are transmitted to regulate the overall traffic

**Error treatment**
- **selective acknowledgements**
  - are negative ACKs
- **cumulative acknowledgements**
  - positive ACKs for amount of data units
- **buffer**
  - defines amount of data units
  - as agreed on between communication entities
  - this is what cumulative acknowledgements refer to





## VMTP

**Versatile Message Transaction Protocol: history**
- **since 1986**

**Objectives**
- **enable efficient RPC (and client/server) paradigm**
- **real-time**
  - can be realized using more than 3 priorities

**Improvements**
- **addressing**
- **multicasting**
- **rate control**





## XTP

**Express Transfer Protocol: History**
- **Greg Chesson later Alfred Weaver**

**Objectives**
- **(software and) VLSI implementation**
- **real-time support**

**Method**
- **combine transport and network layer**

**Some characteristics**
- **control information and data clearly separated**
- **parallel automatons**
- **permits addressing**
  - Internet addresses
  - OSI: NSAPs
- **supports groups**

**Usage**
- **USA: military area, D: early Telekom Berkom MMT (XTP-Lite)**





## SNR

**History**
- **developers: S(abnani), N(etravali) and R(oome)**
- **AT&T Bell Laboratories**

**Objectives**
- **parallel implementation on multiprocessor**

**Low functionality**
- **periodic exchange of complete information on status**
  - for error control and flow management
- **acknowledgements, etc.**
  - sending not based on event
  - periodically transmitted

**Choice of 3 modi**

| Modus | Error Control | Flow Control | Example |
|---|---|---|---|
| 1 | - | - | terminal on host |
| 2 | yes | - | real-time application |
| 3 | yes | yes | data transfer |

# TP++

**History**
- **Bell Communications Research (Bellcore)**
- **1990**
- **among others E. Biersack**

**Objectives**
- **multimedia data**
- **hardware realization (also software)**

**Characteristics**
- **if possible no flow control**
- **back pressure of network layer controls flow**
  - 1 network connection notifies accordingly
  - no multiplexing (several T connections onto 1 V connection)
- **connection control with synchronized clocks**
- **avoidance of retransmissions**
  - forward error correction