T3. Performance and Reliability Analysis of Communication Networks

# Submit to srp@es.aau.dk with subject [NetPerf24][T3]

# [Emil L. H Jens-Ulrik L-M]

## Task 1. Discuss the operation of a connection-oriented packet-switched network.

In connection-oriented service the entire operation can be split into three manor phases: Setup, Data transfer and Teardown. In the setup phase a virtual connection is established through the network from source to destination and once this connection has been acknowledged data transfer can begin. The data transfer function by sending messages one at a time containing data as well as redundancy for the different protocols ensuring the arrival of the message. Once all messages have been sent, the connection can be terminated by both parts and the network can remove the virtual connection.

Let’s look in a bit more detail by using an example.

A source wants to send a message to a destination. The message contains data as well as extra information such as data length and destination address. The message also contains other things such as protocol specific information such as sequence number for TCP but let’s focus on the destination logical address.

For the packet to arrive at its destination the message is sent to a router in the network. The router consults the rounding table, to find the address of the next hop using the destination logical address. If the message cannot be delivered from this router the result would be the next hop logical address. This address will be converted to the MAC address of the device using the ARP protocol, could be the receiver or another router if the message cannot be delivered by the current router. If the diagram is bigger than the MTU (Maximum Transmission Unit) The diagram is split into smaller pieces before delivered to the link layer. This continues until the message arrives at its destination.

Now the idea is for two receivers to become connected like they were connected in a circuit by establishing this connection from A to B. So, the router creates a ‘virtual circuit’ based on the route of the request message. The above process is used for the routers to create their routing tables and establish the virtual connection. Now with the connection the network layer can send packets one after another. When this is done, and the message has been delivered, the source user sends a teardown message. The receiver sends confirmation. All routers clear the established path from their routing tables.

## Task 2 (Revisit). differentiate between circuit-switched and packet-switched networks.

Circuit-switched delivery systems, create ‘a’ route, and keeps it through the entire message correspondent. This ensures that packets are delivered in order and there is a minimal delay. This is very inefficient as no other data can be sent on this path. The connection is thus occupied by only a single user. Even worse the path is still occupied even when no data is sent. Another downside is if just one router breaks down the entire path breaks down and no error can be made saying the message was not received.

In packet-switched networks, the packet can arrive using multiple paths, allowing for redundancy and safety even if parts of the network break down or get congested. This is better for always having more data on the network more efficiently. In packet networks the connection is not occupied by a single user since each packet is encoded with its source and destination allowing multiple users to share the same channel. However, it might be slower and does not guaranty a ordered delivery of packets, this can however be solved by clever use of transfer protocols.

## Task 3. How does TCP work?

The transport layer receives a byte stream from the application layer. Using TCP the byte stream is segmented into TCP packets each with a TCP header. The goal of TCP is control the congestion of the network, retransmit lost or damaged packets and guarantee delivery in a orderly fashion.

During operation it can happen that the receiving party is slower than the delivering, meaning the reading of the packets is slower than the sending. For this reason, the segments are loaded into a buffer and the send one after another. The buffer allows TCP to provide seemingly uninterrupted service to the application layer, while still ensuring the receiver can keep up. The flow from the application layer is only stopped in the buffer backs up and cannot send. It allows the receiver to receive a lot while the sender is continuing work on constructing the next message. This allows the entire system to be faster (Figure 1).

A diagram of a diagram

Description automatically generated with medium confidence

Figure 1 This figure is from the slides. The idea is to illustrate what we have described.

Besides this TCP ensures everything was sent and received. Every single TCP packet needs to be confirmed with an ACK reply, thus ensuring the transmission. If a timeout happens or a retransmit is requested the same data will be sent again until it arrives successfully. Thus, TCP guarantees the arrival of the packet. TCP also does this in an orderly fashion, by assigning a sequence number to each packet, so if packets arrive out of sequence, the missing packet will have to be retransmitted. TCP is used pretty much anywhere on the internet, due to its robustness and guarantee of delivery.

## Task 4. Differentiate TCP and UDP. Which of these protocols is better, and for what applications?

All the benefits of TCP are not for free, as they come at the cost of latency. TCP, as discussed above, is a guarantee for delivery and receiving and adds a lot of safety when transmitting. UDP, unlike TCP does not guarantee delivery or the order of the packets, it just sends the message no control, no limit transmission rate or no order of delivery guaranteed just send the message. Which sounds very rough compared to the careful TCP, however it does have its uses. It is thus easy to understand that you would like your mail TCP, your money transaction TCP or important documents too. But your facetime call you would like real-time or as close as possible. UDP is fast and thus great for facetime or other streaming applications, as if you lose a frame most people won’t even notice. Additionally, most of these lost frames can be “smoothed” out using interpolation between frames, however if e.g. TCP were used in an extreme case one packet could get stuck for 30 s. Here TCP will continue to deliver the packet, while the stream is frozen, not very desirable in a real time application.

## Task 5. Discuss TCP features and why they are important.

### TCP Header

The header of a TCP segment is large 20 or more bytes and contains many of the features:

#### Acknowledgments and sequence numbers

Sequence number: the sequence number is used to keep track of every byte sent by the source, if a packet contains 1000 bytes of data the sequence number will increase with a 1000 after the packet is sent. This keeps track of the packets and allows for detection of out of order packets.

Acknowledgement: the acknowledgement keeps track of every byte received, 1000 bytes is received by some receiver that receiver will add a 1000 to the acknowledgement number when it send out a packet in response. This again keeps track of the packets and allows for detection of out of order packets.

#### Loss detection

TCP uses ACK in its header to send acknowledgement for the packet received say we send packet

A, B, C, D, E🡪 receiver

What we get back is

A, B, C, D, E

Great all the packets were received. How ever if we received

A, B, \_, D, E

C is missing and the TCP will have to retransmit the package. This would have been noticed already at D, when the sender sends D and the receiver would respond with C not received, send again.

If we receive

A, B, C…long time

Here TCP would declare the packet lost due to timeout and request a retransmission.

#### Delayed ACK & The SACK option

To save some bandwidth on the one return channel we can deploy delayed-ACK or SACK in our TCP protocol.

Delayed-ACK: we only respond with ACK every two packets, containing a response for both. And a response is required to be sent 200 ms.

SACK: allows the receiver to describe which exactly packet they haven’t got ACK[A,\*,C,D,E] then we only need to send B again instead of starting from B again.

#### TCP 3-way handshake

The client sends a TCP message with SYN flag raised to a server

Flag = SYN sequence number = x 🡪 server

The server responds by sending an ACK acknowledgement for x +1 back. The server also sends SYN flag raised whit its sequence number

Flag = ACK acknowledgement = x +1 🡨 server

Flag= SYN sequence number = y 🡨 server

The client response by sending ACK acknowledgement for y + 1

ACK acknowledgement = y+1 🡪 server

Thus we get the common SYN SYN-ACK SYN also known as the TCP 3-way handshake. With this both parties have the sequence number and have sent acknowledgement of what they expect to receive next thus ensuring the sender that all was received correct and there is connection established.

#### TCP connection termination

One user sends a message with the FIN flag raised, when the message has been received acknowledgement that user don’t send any more data (this is done when you have no more to send). The other user sends data until they have no more to send then they too send FIN flag raised, they receive acknowledgement and the connection is closed.

#### Flow control

The window size is used and sent in a ACK, thus the sender knows how many bytes the receiver can process and don’t send to much. This is important because it ensures that the TCP connection do not loose any data.

## Task 6. What is HoL blocking problem? Discuss potential solutions.

HoL (Head-Of-Line Blocking) occurs when a single channel is shared between multiple users. Each user wants to send packets and thus must use a queue system. If the packet in head of the queue is blocked, then HoL occurs if the packets behind could be sent instead. Since the medium is shared the packets which could be sent are blocked by the head of the line. To circumvent this issue, multiple solutions can be implemented.

Solution splitting streams

If the packet stream is split into several streams, then each packet would no longer be blocked by the head of the line. Thus, any communication from A going to B does not hinder communication from A to C or D since these each have their own stream. This splitting creates a problem of in order transmission since the queue is no longer systematically emptied from the head. However, each stream will arrive in order.

Solution QUIC protocol

The QUIC protocol is designed by Google to support streams. QUIC builds on top of UDP but adds many of the same features as TCP, however it also adds its own part of the header, being connection ID to keep track of the streams. In QUIC multiple streams can receive data from the same packet and it also employs SACK to further push performance. Thus, with QUIC the HoL problem is alleviated.

Solution FEC

By using Forward Error Correction one can send redundant information along with each packet and send it along different paths such as 4G, 5G, LAN and WIFI. With this method all packets contain information of the entire packet, meaning no matter if an error occurs in the original packet in can be instantaneously recovered and thus the problem of HoL is alleviated since we can recover the delayed packets from the error correction.