Computer Networks

COMP 3670 Application Layer

Sherif Saad

June 4, 2021

Preface

These are lecture notes for the computer network course taught at the school of computer science, University of Windsor in summer 2021. I used several sources to create these lecture notes, but the main references are [For17],[TvS07] and [KR16]

Please email me if you notice any mistakes or typos.

Table of Contents

Contents

1 Delay in Packet-Switched Networks

Packet-Switched Networks

- Packets sent between two hosts pass through a series of routers.
- Packets sent between two processes over the network could take different paths (routes).
- Packets on the network could get lost, corrupted, arrive out of order, or encounter different delays.

This means we can not assume that we could move as much data as we wish between any two end systems.

Types of Delays

1. Processing Delay
2. Queuing Delay
3. Transmission Delay
4. propagation Delay

The sum of these delays gives us the **total** nodal delay.

Types of Delays

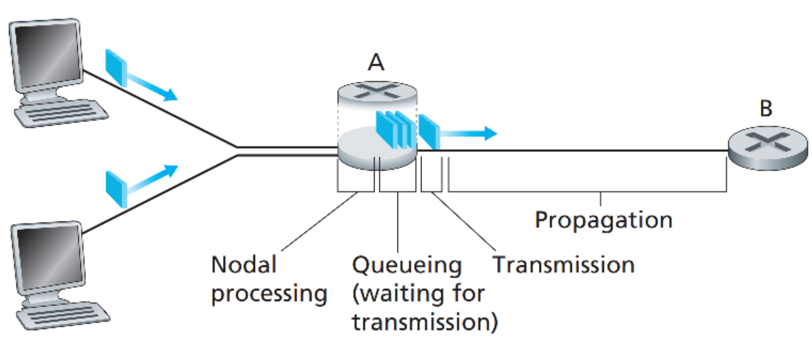

Figure 1: Types of Network Delays

Processing Delay

- The route needs to check the packet header information to determine the appropriate outbound link (next router).
- The time required to examine the packet's header and determine where to direct the packet is part of the processing time.
- The time to check the packet to detect any bit-level errors is also part of the processing time.
- Processing delays in high-speed routers are typically on the order of microseconds.

Queuing Delay

- A packet can be transmitted if no other packets are transmitted on the link.
- If the link is busy, the packet is stored on a local buffer at the router.
- The local buffer is simply a queue (first packet in, first packet out)
- The queuing delay is unbounded; it depends on the current traffic load at the router.
- Queuing delay can be on the order of microseconds to milliseconds.

Transmission Delay

- The packets are transmitted in a first-come-first-served (queue).
- Each communication link between two adjacent nodes has a transmission rate.
- The transmission rate indicates the number of bits the link can transmit per unit of time, mostly per second.
- The transmission delay is the amount of time required to push (transmit) all of the packet's bits from the buffer into the link.
- Transmission delay can be on the order of microseconds to milliseconds.
- For a packet of $\boldsymbol{L}$ number of bits and a transmission rate of $\boldsymbol{R\ bit/sec}$, the transmission delay is $\boldsymbol{L/R}$.

Propagation Delay

- The bits need to propagate to the router at the other end of the link.
- The time required for the bit to propagate from the beginning of the link to the end of the link is the propagation delay.
- The bits need to travel the physical distance between the two routers.
- Each link has a propagation speed; this speed depends on the physical medium of the link (fiber optics, twisted-pair copper, etc.)
- The propagation speed is almost equal or a little less than the speed of the light.
- The propagation delay is the physical distance between the two routers divided by the propagation speed.

Queuing Delay and Packet Loss

- If the packet arrives on an empty queue, the packet will be transmitted immediately (queuing delay is zero)
- Queuing delay is not fixed; it can vary from packet to packet.

When the queuing delay significant, and when is it insignificant?

What happen if the queue become full (overflow)?

Significant and Insignificant Queuing Delay

- The packet arrival rate at a given router and the transmission rate of the link will determine the significance of the queuing delay.
- Now, let $\boldsymbol{a}$ denote the average rate at which packets arrive at the queue, where $\boldsymbol{a}$ is in units of *packets/sec*. Let all the packets have the same length $\boldsymbol{L}$ number of bits. Let the transmission rate of the link $\boldsymbol{R}$ $\boldsymbol{bits/sec}$, then $\boldsymbol{a = La\ bits/sec}$

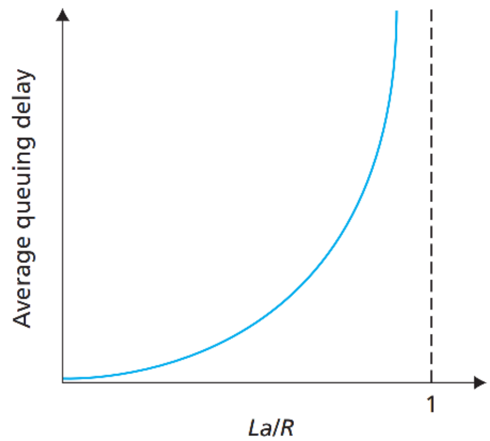

Figure 2: Dependence of average queuing delay on traffic intensity

- If $\boldsymbol{La/R \leq 1}$ and close to zero then the queuing delay is insignificant, For example, packet will arrive at an empty queue.

Do packets arrive at the router periodically or at random time?

Average Queuing Delay

End-to-End Delay

What is the end-to-end delay?
The total delay a packet encounter from from source end-system to destination end-system.

When there is $N-1$ routers between the source and destination, we can express the end-to-end delay using the following:

$$d_{end-end} = N(d_{proc} + d_{trans} + d_{prop})$$

Assuming that the delay at each router is identical, and there is no queuing delay.

Throughput, Bandwidth and Latency

Throughput

- The term throughout refer to the amount of data (bits or packets) that can be sent and received within specific time frame.
- In other words, throughput measures the rate at which packets arrive at the destination successfully.
- It could be measured in **bits/sec** or **packets/sec**.

Let us say sending a file of size $\boldsymbol{F}$ $\boldsymbol{bits}$ from host $\boldsymbol{A}$ to host $\boldsymbol{B}$ took $\boldsymbol{T}$ $\boldsymbol{seconds}$, then we can say that the average throughput for file transfer is $\boldsymbol{F/T}$ $\boldsymbol{bits/sec}$

Throughput, Bandwidth and Latency

Bandwidth

- Bandwidth is a measure of how much data can be sent and received at a time.
- Bandwidth indicate the capacity of the communication link.
- The higher the bandwidth a network has, the more data it can send back and forth.
- Bandwidth is **not** an accurate term to measure the speed.
- Bandwidth could be measured or expressed in bits/megabits/gigabits per second.

Throughput, Bandwidth and Latency

Latency

- Latency is the amount of time that it takes for bits/packets to travel from a source to the destination.
- Latency determines how fast these packets actually reach their destination.
- The different types of delays will impact the latency and the throughput.

End-to-end Throughput

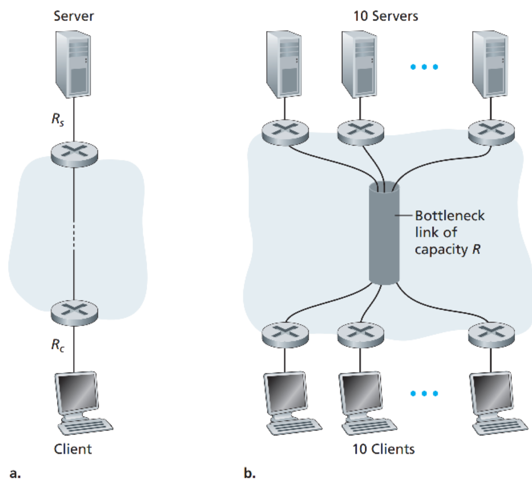

Figure 3: (a) Client downloads a file from server; (b) 10 clients downloading with 10 servers

2 Transport Layer Services

Transport-Layer

- The transport layer enables multiple processes on one host to communicate with multiple processes on another remote host over the network.
- The transport layer extends the network layer services from enabling two end systems to communicate to allowing processes on the end systems to communicate.
- One key challenge for transport-layer protocols is how to enable reliable message delivery between remote processes when the underlying infrastructure may lose and corrupt data.

Transport-Layer Protocols

- Transport-layer protocols provide a logical communication link between the processes running on different end systems.

- This logical link hides the complex communication paths (physical infrastructure) between the two end systems.
- Transport-layer protocols are implemented in the end systems, not in the router or switches.
- The transport layer protocol will break each message received from the application layer into small chunks (segments) and send these chucks over the network by handing them to the network layer.
- On the destination end system, the transport layer receives the segments from the network layer. It reconstructs the application layer message from the sender and hands it over to the application layer at the destination.

Transport Layer in TCP/IP Networks

- The UDP (User Datagram Protocol) and the TCP (Transmission Control Protocol) are the most common transport-layer protocols in TCP/IP networks.
- The UDP provides **unreliable**, **connectionless** transport services to the network application.
- The TCP provides **reliable**, **connection-oriented** services to the network application.
- The IP (Internet Protocol) is the most common and important network layer protocol in TCP/IP networks.
- The IP protocol provides logical communication between hosts.
- The IP service model is a **best-effort** delivery service (unreliable service)

Transport-Layer Services/Protocols

What are the main functions any transport layer protocol needs to provide?

- Extending host-to-host delivery to process-to-process delivery (aka transport-layer multiplexing and demultiplexing)

- Provide data integrity validation and error detection (not error correction or recovery)

Providing error detection and integrity checking does not mean reliable data transfer.

The network application could mix/use more than one transport layer protocol.

Multiplexing and Demultiplexing

- A network process could have one or more network sockets.
- The transport layer protocol does not deliver the messages directly to the process; instead, it provides the message to a socket.
- Each network socket must have a unique identifier (Why?)
- Getting the message from the socket, dividing it into segments, and passing the segments to the network layer is multiplexing.
- Getting the segments from the network layer, identifying the socket that should receive the message, and passing it to the socket is demultiplexing.

Note: The multiplexing and demultiplexing functions occur in any communication layered architecture whenever multiple protocols use a single protocol at the adjacent lower layer.

Multiplexing and Demultiplexing

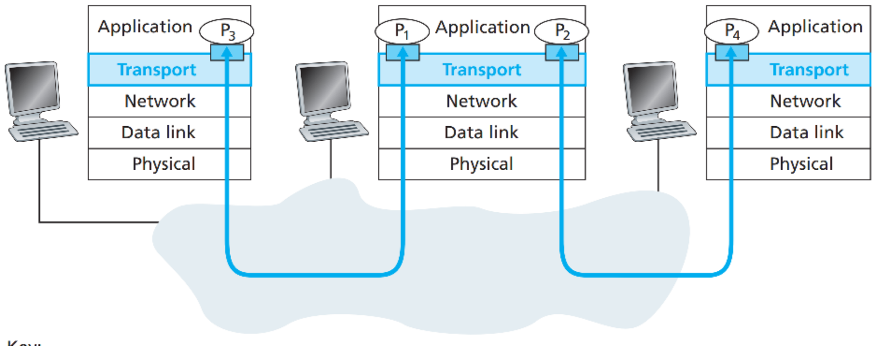

Figure 4: Transport-layer multiplexing and demultiplexing

Transport Layer Sockets

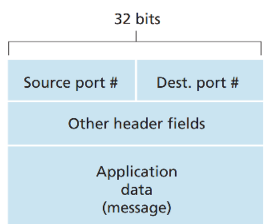

Figure 5: Source and destination port-number fields in a transport-layer segment

- Any transport layer protocols must be able to distinguish between network sockets.
- Therefore, each socket must have a unique identifier.
- The transport layer segment or message must carry this identifier.

UDP and TCP Sockets

- The unique identifier of the socket is the combination of port numbers and IP addresses.
- A UDP socket is identified by the destination IP address and the destination port number.

- If two UDP segments have different source IP addresses and/or source port numbers but have the same destination IP address and destination port number. The two segments will be passed to the same destination process via the same destination socket.

- A TCP socket is identified by the destination IP address, the destination port number, the source IP address, and the source port number.

UDP Socket

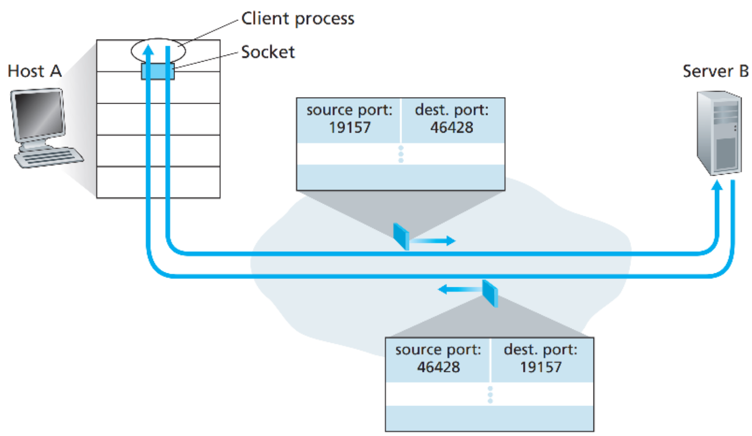

Figure 6: UDP Socket Identifier

TCP Socket

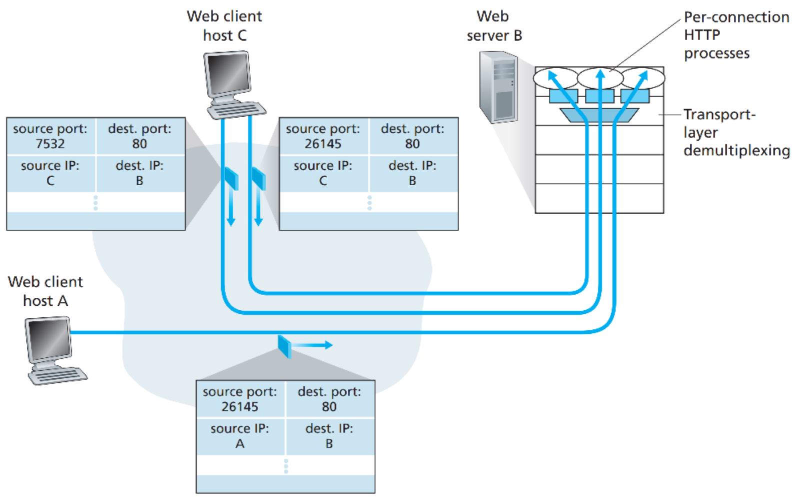

Figure 7: TCP Socket Identifier

3 Connection Less Transport Protocol

Connection Less Transport Protocol: UDP

- UDP is a connectionless protocol
- UDP is an unreliable communication protocol, does not guarantee message delivery.

Why using UDP if UDP is doing nothing? Why not pass the message directly from the application layer to the network layer?

When Should We Use UDP?

- UDP is suitable for real-time applications that could tolerate packet loss. Minimum packet header overhead (only 8 bytes comparing to TCP, which has 20 bytes)
- There is no connection state or and no need to establish the connection.

- If we are moving data and data integrity is critical, then use TCP. For command and control (instructions), then use UDP.
- For data stream, if partial data loss is accepted, use UDP and then use UDP if real-time application.

TCP Socket

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
Network management	SNMP	Typically UDP
Name translation	DNS	Typically UDP

Figure 8: Popular Internet applications and their underlying transport protocols

UDP Segment Structure

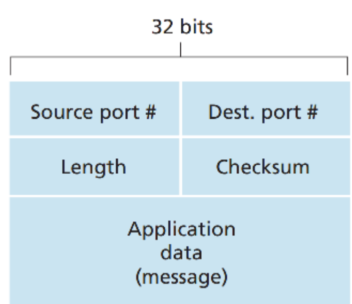

Figure 9: UDP segment structure

UDP Segment Structure

- The UDP segment is divided into header and msg payload.

- The UDP header consists of 88 bytes. That is divided into four fields, each consisting of two bytes (16 bits).
- The four fields of the UDP header are(source port number, destination port number, length, and checksum)
- The length field specifies the number of bytes in the UDP segment, including the header and data (payload). This is because the size of the data field may differ from one UDP segment to the next.
- The checksum filed is provided for error detection. But UDP does nothing to recover from error.

4 Principles of Reliable Data Transfer

Reliable Data Transfer

What do we mean by reliable data transfer?

- Data are guaranteed to arrive at the destination (no data loss)
- Data integrity is guaranteed (no transferred data bits are corrupted like flipped from 0 to 1, or vice versa)
- All messages (segments) are delivered in the order in which they were sent.

Why reliable data transfer is a challenge?
The main challenge is the underlying infrastructure is unreliable (zero guarantees)

Reliable Data Transfer

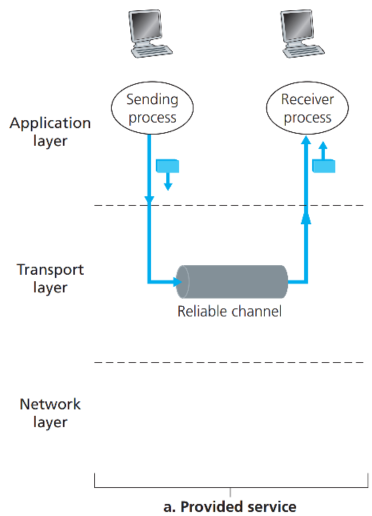

Figure 10: Reliable data transfer: Service model

Reliable Data Transfer

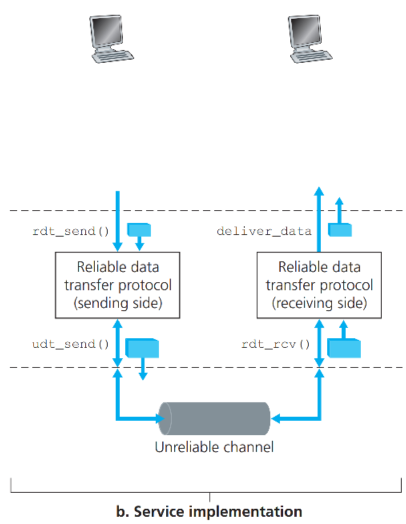

Figure 11: Reliable data transfer: Service Implementation

Building Reliable Data Transfer

Design Strategy

To explain how we could design a reliable data transfer protocol over unreliable infrastructure, we will start with a simple design and unrealistic assumptions and evolve our design by adding complexity and realistic assumptions a few at a time.

RDT v1.0

Reliable Data Transfer (RDT) over a Perfectly Reliable Channel Assumptions

1. Reliable channel
2. Data is sent in one direction (unidirectional data transfer)
3. No need for feedback
4. We can send as many messages as we wish as fast as we can.

RDT v1.0

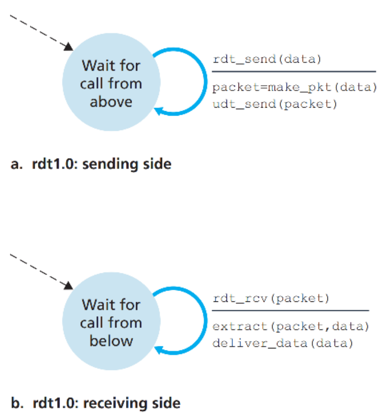

Figure 12: A protocol for a completely reliable channel

RDT v2.0

Reliable Data Transfer over a Channel with Bit Errors Assumptions

1. Message corruption is possible (bits are corrupted like flipped from 0 to 1, or vice versa)

2. No packet or messages lose
3. No out-of-order packets or messages.

RDT v2.0

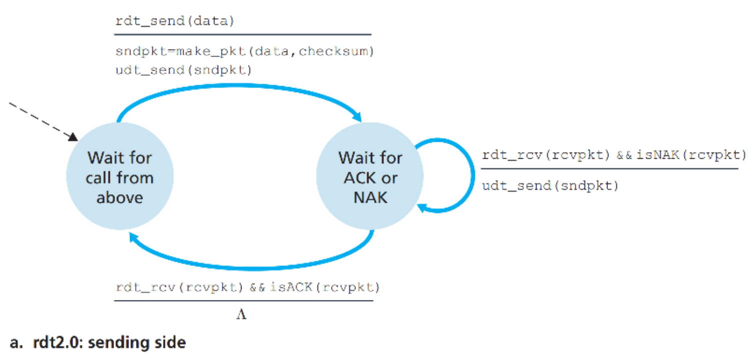

Figure 13: rdt2.0: sending side

RDT v2.0

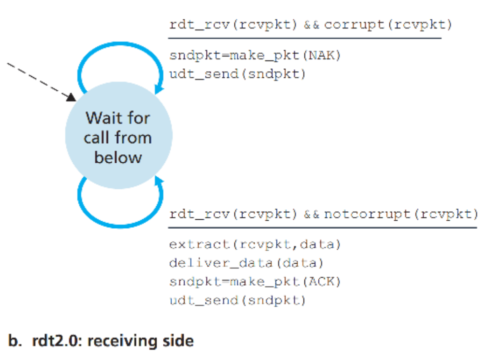

Figure 14: rdt2.0: receiving side

RDT v2.0 Behaviours

- The protocol, in this case, require feedback from the receiver side
- Positive feedback from the receiver indicates that the message's integrity is preserved.

- Negative feedback from the receiver indicates receiving a corrupted message. On negative acknowledgment, the sender will resend the message (repeat until) and waiting for the positive feedback.
- The sender needs a local buffer to store a copy of the message until it receives positive feedback from the receiver.
- The sender can not send any new message until it receives the positive acknowledgment of the last message it sent.
- The RDT2.0 belongs to the stop-and-wait protocols family.

RDT v2.0 Design Goals

- The protocol provides reliable data transfer against bit errors using the retransmission approach, ARQ (Automatic Repeat reQuest) protocol.
- The sender can not send any new message until it receives the positive acknowledgment of the last message it sent.
- The protocol can handle bit errors by implementing an error detection mechanism, uses feedback, and enable retransmission on failure.

What is the primary design flaw of this protocol?

RDT v2.0 Design Flow

What is the primary design flaw of this protocol?
If the feedback (positive or negative acknowledgment) was corrupted during the transmission. The sender will not be able to know if the receiver received the packet or not.

Possible Solutions

- The sender asks the receiver to resend the feedback
- Adding complex checksum to enable the recovery from the bit errors
- The sender on corrupted feedback could resend the packet (e.g. assume it received negative feedback)

RDT v2.1

RDT with Sequence Number over a Channel with Bit Errors

- When the sender resents the packet after receiving corrupted feedback, we introduce a new problem: duplicate packets at the receiver side.
- To solve the duplicate packets into the sender-to-receiver channel, we need to enable the receiver to distinguish between old and new packets.
- The most common solution is to add a sequence number filed to the transport protocol header section.

RDT v2.1

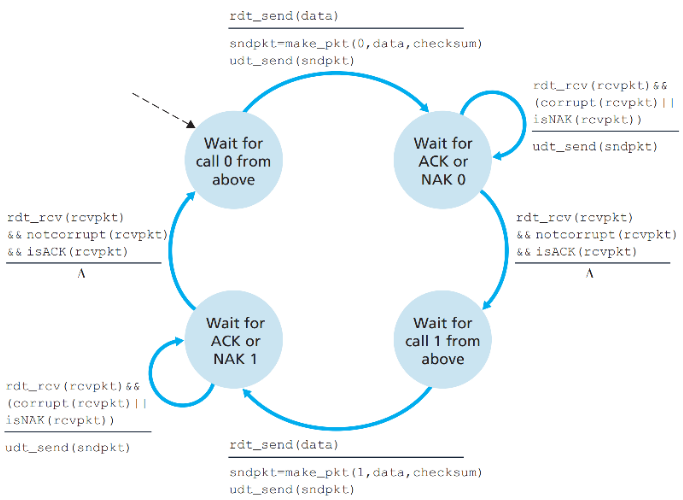

Figure 15: rdt2.1 sender

RDT v2.1

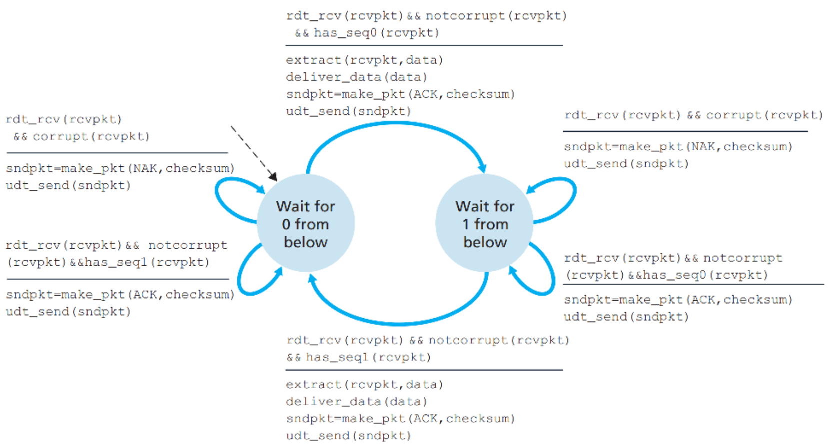

Figure 16: rdt2.1 receiver

RDT v2.2

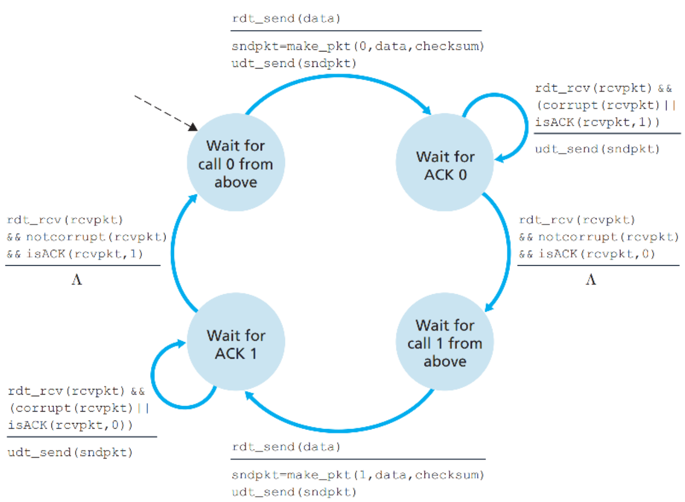

Figure 17: rdt2.2 sender

RDT v2.2

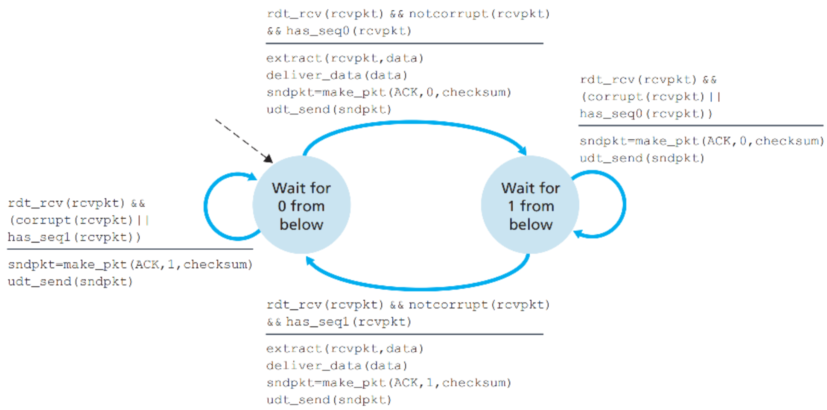

Figure 18: rdt2.2 receiver

RDT v3.0

Reliable Data Transfer over a Lossy Channel with Bit Errors Assumptions

1. Assumptions for RDT 2.0
2. The massages|packets or the acknowledgment might get lost
3. The sender is responsible for detecting lost packets and resolve them using retransmission.

Detecting Packet Loss

How could the sender detect packet loss?

- There is no way the sender could be sure that the packet is lost and will never arrive at the receiver.
- There is no way the sender could be sure that wither the packet or the acknowledgment is lost.

The solution is to implement a time-based retransmission mechanism using a count-down timer.

RDT v3.0 with Timer

How to use the count-down timer?

1. Choose a value for the count down timer
2. Start the timer each time a packet is sent.
3. Respond to a timer interrupt.
4. Stop the timer.

How to choose the value for the count-down timer?

What could be the effect of premature timeout?

RDT v3.0

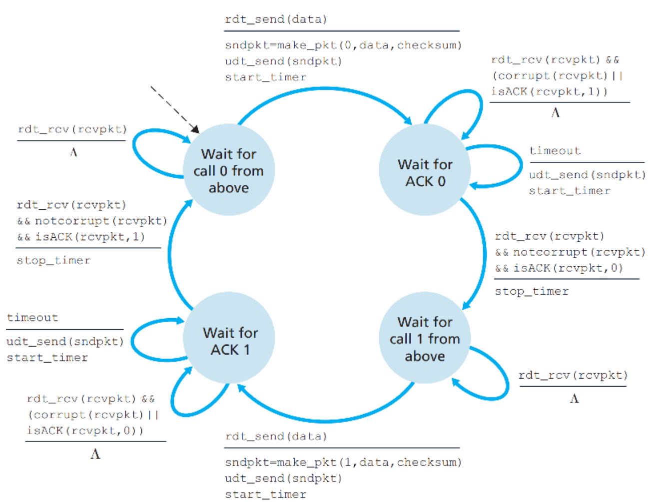

Figure 19: rdt3.0 sender

RDT v3.0

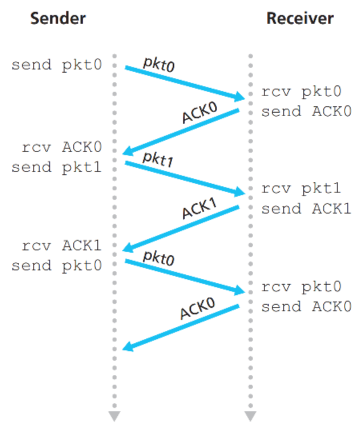

Figure 20: rdt3.0 Normal Operations

RDT v3.0

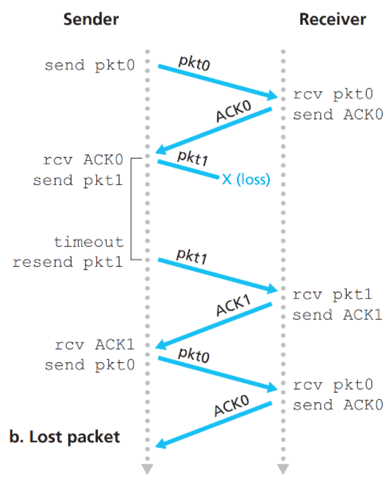

Figure 21: rdt3.0 Packet Loss

RDT v3.0

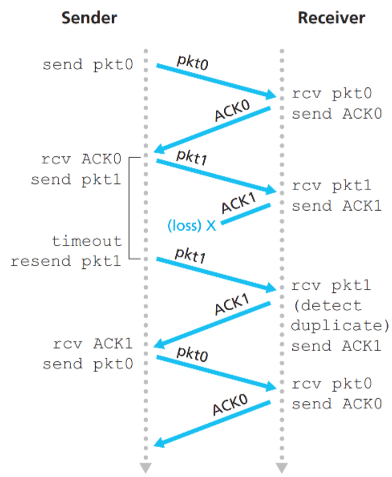

Figure 22: rdt3.0 Lost ACK

RDT v3.0

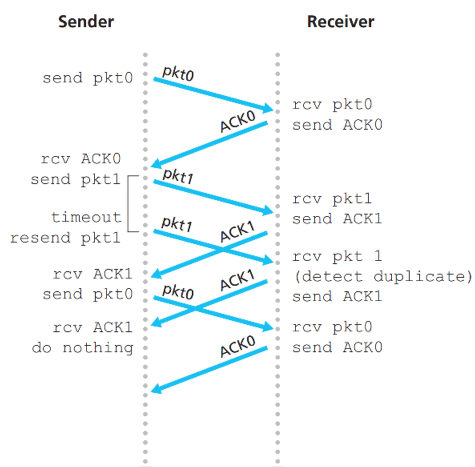

Figure 23: rdt3.0 Premature timeout

Pipelined Reliable Data Transfer Protocols

- RDF v3.0 is a reliable data transfer protocol. However, it is an inefficient protocol. This is because it uses a stop-and-wait pattern.

- RDF 3.0is sometimes known as the alternating-bit protocol. Because packet sequence numbers alternate between 0 and 1.
- A better solution is to use a pipelining pattern instead of the stop-and-wait.
- With pipelining, the sender is allowed to send multiple packets without waiting for acknowledgments.

What is needed to enable pipelining?

Pipelined RDT

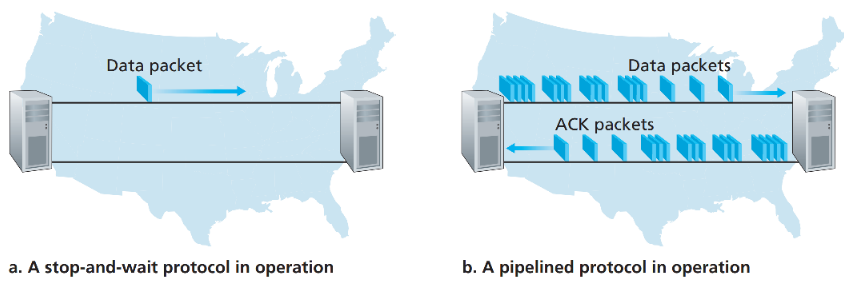

Figure 24: Stop-and-wait versus pipelined protocol

Enabling Pipelining

What is needed to enable pipelining?

- The range of sequence numbers must be increased.
- The sender and receiver sides of the protocols may have to buffer more than one packet.

What is the range of sequence number and what is the buffering requirements?

What is the primary impact of enabling pipelining on reliability of data transfer?

Next Week

What is the plan for next week?

- Transport Layer (Pipelining, TCP Protocol, Congestion Control)
- Lab 3 (Investigating Transport Layer Protocol Behaviours)
- Quiz 3 (6%) cover this week topic "Transport Layer"

References

References

[For17] James Forshaw, *Attacking network protocols: A hacker's guide to capture, analysis, and exploitation*, 1st ed., No Starch Press, USA, 2017.

[GM98] John S. Gero and Thomas Mc Neill, *An approach to the analysis of design protocols*, Design Studies **19** (1998), no. 1, 21–61.

[KR16] James F. Kurose and Keith W. Ross, *Computer networking: A top-down approach*, 7 ed., Pearson, Boston, MA, 2016.

[M.R01] M.Rose, *Beep: Building blocks for application protocols.*, 2001.

[RM01] Rose and Malamud, *Rfc3117:on the design of application protocols*, 2001.

[TvS07] Andrew S. Tanenbaum and Maarten van Steen, *Distributed systems: Principles and paradigms*, 2 ed., Pearson Prentice Hall, Upper Saddle River, NJ, 2007.