6. Transport layer

Transport layer protocol, data processing and aggregation

Wireless sensor networks

Martin Úbl
ublm@kiv.zcu.cz

2023/24

<DCSE> DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Transport layer

Basics

- transport layer – ISO/OSI layer 4
- we assume that:
  - L2 and L3 is designed and implemented
  - L2 and L3 transfers frames and packets by its rules
  - transfers may be lossy, erroneous
- transport layer role:
  - maintains reliable delivery
  - reduces traffic congestions

Transport layer

Basics

- **reliability**
- transfer is reliable, if the data are delivered to target node:
  - whole
  - in the correct order
  - without errors (unaltered)
- in terms of transport layer, we speak of **end-to-end** reliability
  - L2 layer would potentially consider *hop-to-hop* reliability
  - it is, however, possible to implement hop-to-hop reliability to achieve end-to-end reliability

Transport layer

Basics

- **reliability**
- types of reliability recognized in WSN
  - point-to-point
    - when two specific nodes exchange packets
    - e.g., edge node with remote server
  - point-to-multipoint
    - when a specific node targets a group of nodes
    - e.g., sink node scatters packets to whole network
  - multipoint-to-point
    - when a group of nodes targets a single specific node
    - e.g., all network nodes propagates packets to a sink node

Transport layer

Basics

- transport layer protocol
- do we really need one in WSN?
  - we could omit L4 protocol if we don't require reliable transfers
  - or if no congestion can happen at all
- in fact, we should always consider at least minimalistic implementation

Transport layer

Basics

- ordinary "large" networks
- dominant protocols:
  - TCP – reliable, connection-oriented protocol
  - UDP – unreliable, connection-less protocol
  - and others, like RTP, ...
- we cannot use either one in WSN
- even the UDP header has 8 bytes
  - imagine that the data has 4 bytes – the the UDP header would have twice as much
  - after encapsulation in L3 and L2 PDUs, our 4 byte data might end up with more than 80 % of control fields and overhead!

Transport layer

Basics

- we have to design a transport protocol suiting the WSN needs
- basic requirements:
  - minimalistic (header size)
  - scalable
  - independent of underlying protocols
  - control overhead must not exceed a certain value
    - otherwise we would end up transmitting mostly overheads
  - energy efficient
- on the other hand: *we usually do not require 100 % reliability in WSN*
  - if the overheads are too high, do not attempt for reliable delivery
  - "too high" – protocol- and application-specific

Transport layer

Basics

- however...
- some nodes that are not resource-constrained may use TCP
- for example, nodes that forms the backbone of the WSN
- backbone nodes might not be limited in terms of energy
- however, they are still often implemented as embedded devices

Transport layer

PSFQ

- **Pump Slowly Fetch Quickly** (PSFQ)
- transport layer protocol designed for low-power wireless applications
- main goal: point-to-multipoint reliability
- data is *pumped slowly* from root node to the network
- when data loss occurs, nodes *fetch quickly* from immediate neighbors
- do not use positive acknowledgment (ACK)
- use negative acknowledgment (NACK)

Transport layer
PSFQ

- PSFQ is usable for rather specific scenarios
- assumes light traffic
- does not offer any congestion control
- suitable for e.g., updating firmware of WSN nodes
- can be seen as a reliable multicast on L4 layer
- can be used for virtually any communication scheme, but may not be efficient
  - e.g., multipoint-to-point reliability with PSFQ exhibits very poor performance

Transport layer

PSFQ

- PSFQ missing data detection
- every message has an increasing sequence number
- *gap detection* – when a new message arrives with a higher sequence number than expected, missing data are detected
- node, that detected gaps, tries to recover:
  - starts to singnalize NACK to all immediate neighbors, one-by-one
  - not every node might have the required data
  - since the root *pumps slowly*, nodes have enough time to obtain missing data if they *fetch quickly*

Transport layer

PSFQ

- PSFQ upstream nodes must have a buffer to store past packets
- if the downstream node signalizes NACK, we must be able to recover
- the size of the buffer is determined by the application domain

Transport layer

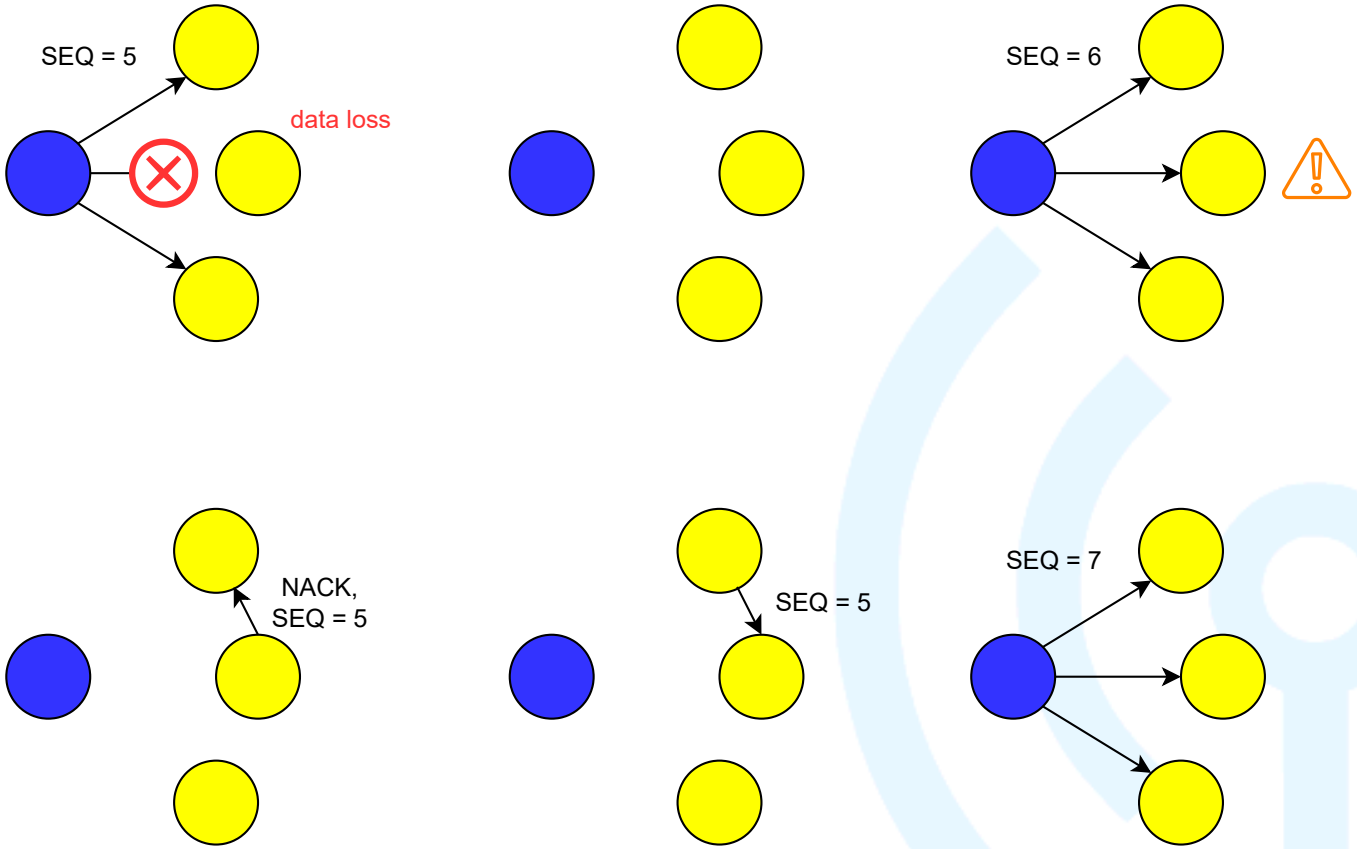
PSFQ

- PSFQ cannot detect missing first and last parts of message
- *proactive fetch* – node sets timer to detect missing data based on timing
  - if no message arrives until timeout, node pro-actively broadcasts NACK
- in case of file transfers (e.g., firmware image update), packets contain total size
  - by checking the total amount of data received against the total image size, we may detect missing first and last parts

Transport layer

PSQF

- example of PSFQ transfer with data loss (left to right, top to bottom)



Transport layer

RMST

- **Reliable Multi-Segment Transport** (RMST)
- designed as a L4 protocol over directed diffusion L3 protocol
- enhances diffrected diffusion by:
  - segmentation and segment reassembly
  - reliable data transfer
- two modes:
  - hop-by-hop recovery (cached)
  - end-to-end recovery (non-cached)
- oriented towards both point-to-multipoint and multipoint-to-point reliability

Transport layer

RMST

- also based on gap detection
- RMST does not guarantee in-order transmission
  - multiple sources, multiple paths, ...
- there is no way to differentiate between out-of-order arrival and missing data at the time of gap detection
- nodes maintain "gap bitmap"
- nodes with missing data sets a timer for each missing piece of data
  - if the gap is not filled within a time period, NACK is generated
  - just like in the PSFQ, node asks its closest neighbors (hop-to-hop recovery)
    - alternatively, if no node has the data, it asks the source (end-to-end recovery)

Transport layer

RMST

- *NACK combining*
- both PSFQ and RMST may combine multiple NACKs into a single message
- reduces traffic
- reduces overheads

Transport layer

ESRT

- **Event to Sink Reliable Transport** (ESRT)
- focuses on multipoint-to-point reliability and *reliable event detection*
- does not guarantee 100 % reliability "rate"
- instead, it chooses a different metric of reliability
- instead of propagating a fine detail of event description, it focuses on reliably transmitting a coarse information
- cannot be used for situations that require deliver of all segments
  - e.g., firmware update, security-related information, ...

Transport layer

ESRT

- essential properties:
  - self-configuration
  - energy awareness
  - congestion control
  - collective identification
  - biased implementation

Transport layer

ESRT

- *self-configuration*
- events must be detected in adverse network conditions
- ESRT is able to self-adjust
  - ideal operation interval
  - thresholds

Transport layer

ESRT

- *energy awareness*
- ESRT poses most of reliability-related tasks to the sink
- sinks usually have more energy, or they are not constrained at all
- ESRT also may reconfigure to reduce thresholds of detected events to reduce traffic

Transport layer

ESRT

- how is ESRT reliable?
- the main paradigm is to find an optimal frequency of transmissions
- this must take into account:
  - payload size
  - average traffic
  - current reliability
  - available energy

Transport layer

ESRT

- finding the optimal region of operation
- observed reliability: r
- desired reliability: R
  - R can be interpreted as a threshold of reliable event detection
- measure of reliability: $\eta = \frac{r}{R}$
- toleration threshold: ϵ (adjusted at WSN deployment)
- the actual problem: find a frequency f to optimize r to be as close to R as possible
  - i.e., η to be as close to 1 as possible

Transport layer

ESRT

- operating regions

Operation interval	Frequency	Reliability
no congestion, low reliability (NC, LR)	$f < f_{max}$	$\eta < 1 - \epsilon$
no congestion, high reliability (NC, HR)	$f \leq f_{max}$	$\eta > 1 + \epsilon$
congestion, low reliability (C, LR)	$f > f_{max}$	$\eta > 1$
congestion, high reliability (C, HR)	$f > f_{max}$	$\eta \leq 1$
optimal operating region (OOR)	$f < f_{max}$	$1 - \epsilon \leq \eta \leq 1 + \epsilon$

Transport layer

ESRT

- if $R = 1$, fully reliable transfer is required
- practically, we set R around 0.8 (80%)
- really depends on events we try to detect

Transport layer

Choice

- choice of transport protocol depends on many aspects
  - degree of reliability
  - type of reliability
  - data transmitted
    - event detection
    - firmware update
    - node retasking
    - security-related information
  - type of transfer (point-to-multipoint, ...)
  - capabilities
    - cache size (for NACK-based protocols)
    - recovery type

Transport layer

Comparison

- comparison by recovery type

Protocol	End-to-end	Hop-by-hop
PSFQ		✓
RMST	✓	✓
ESRT	✓	

Transport layer

Comparison

- comparison by congestion control

Protocol	Congestion detection	Congestion control
PSFQ	✗	✗
RMST	✗	✗
ESRT	✓(buffer inflation)	✓(throttling)

Transport layer

Comparison

- *energy efficiency* and signaling
- PSFQ
  - low throughput
  - NACK-based signaling
  - in-order delivery
- RMST
  - low to moderate throughput
  - NACK-based signaling
  - out-of-order delivery
- ESRT
  - high throughput
  - sink node aggregates reliability info
  - out-of-order delivery by design

Transport layer

Conclusion

- there are more transport layer protocols
- e.g.,
  - E^2SRT
    - stabilization mechanism for *f*
  - GARUDA
    - framework for reliability and security
  - CODA
    - congestion control protocol

Transport layer

Conclusion

- for a normal sensor network operation, we allow some data loss
- if a single temperature value gets lost, it does not matter
  - in a few minutes, a new value will arrive
  - if not, there are N more sensors that will provide a sufficient approximation
- there are scenarios, in which we must consider reliability, e.g.:
  - event detection (to some degree)
  - firmware update (100 %)
  - node retasking (100 %)
  - security data distribution (100 %)