Internet Engineering

241-461

Robert Elz

kre@munnari.OZ.AU
kre@coe.psu.ac.th

http://fivedots.coe.psu.ac.th/~kre

Contents

Fragmentation Algorithm

- Remove header from packet
  - Save it for later
- Divide payload of packet into sections
  - each small enough for MTU after header is re-attached
- Put modified header on each fragment
  - Total Length
    - length of fragment (plus its header)
  - M
    - set to 1 in all but last fragment generated
    - In last, copied from original packet
  - Fragment Offset
    - in first fragment generated copied from original packet
    - in all other fragments
      - previous fragment offset field + length of its payload / 8
  - All other fields
    - copied from original packet
- Done!

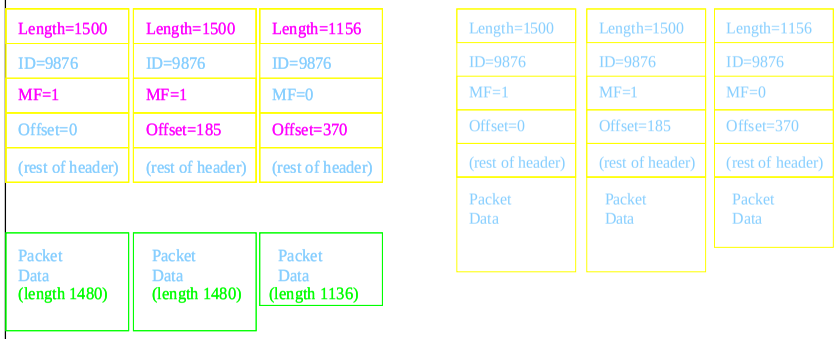
Fragmentation Example

- Take the original packet

  - Separate header from payload
  - Then divide payload into pieces

Fragmentation Example (cont)

- Construct headers for new fragments
  - with copied and modified fields
  - from original packet header



  - Attach headers to packets
    - Then ready to transmit

Contents

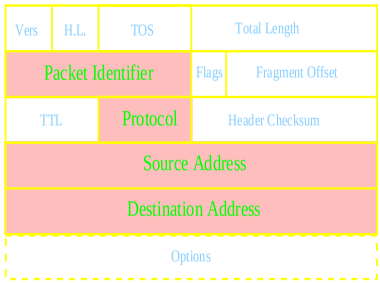
- Why Fragmentation?
- Why Reassembly?
- When to Fragment
- When to Reassemble
- How to Fragment
- How to Reassemble
- Why not fragment
- How to avoid fragmentation (PMTUD)

How to Reassemble

- The Problems
  - Fragments of many different packets
  - Fragments arrive out of order
  - Fragments never arrive
  - Retransmitted packets
    - same packet different fragments
- The Requirement
  - Sort it all out
  - Deliver reassembled packets
- How?

Reassembly

- Must have queue(s)
  - fragments arrive
  - must wait for later fragments
- Want to group fragments
  - fragments of the same packet
- Fragment Classification
  - Packet identifier
  - Protocol
  - Source Address
  - Destination Address
- Classification Fields the same
  - 2 fragments belong to same packet
- Classification Fields differ
    - Any of them
  - 2 fragments belong to different packets



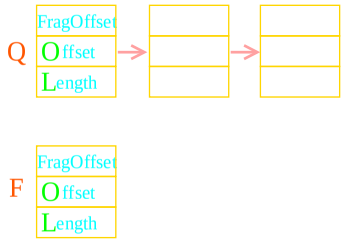
Reassembly Algorithm

- One fragment queue
  - For each different packet
  - Where fragment has arrived
    - deliver complete packets, don't queue them
- Queue ordered
  - by order of fragment offset
    - smallest first
    - biggest last
  - not arrival order
- Merge fragments on queue
  - whenever possible
  - method coming soon
- When queue contains complete packet
  - Deliver packet
    - no more need for this queue

Fragment Merge Decision

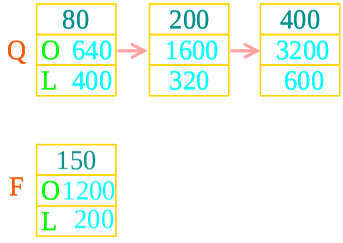
- Fragment F arrives
  - Fragment offset O (in bytes)
  - Fragment Length L (excluding header)
- Find the appropriate queue
  - If none
    - create one -- nothing to merge, put F in new queue
- Scan queue in order (increasing fragment offsets)
  - For each packet Q on the queue in order
    - if O(F) < O(Q) && O(F) + L(F) >= O(Q)
      - F merges into Q (details later)
      - done
    - if O(F) < O(Q)
      - F inserted in queue before Q
      - done
    - if O(Q) + L(Q) >= O(F)
      - F merges into Q (details later)
      - use merged packet as F
      - continue
  - if reach end of queue, append F

Merge Decision Examples



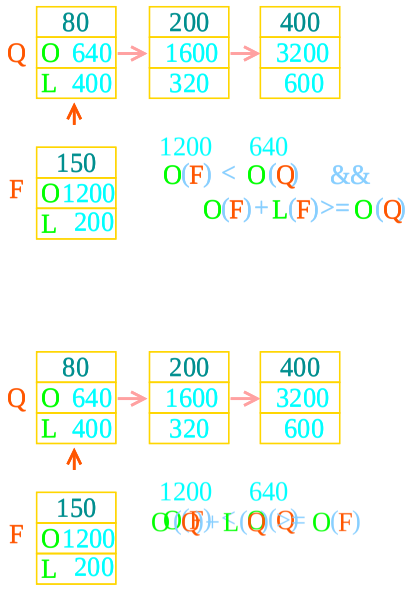
- Queue of waiting Fragments
- Fragment arriving
- Already tested, and know
  - all fragments from same packet
- i.e.:
  - same source & destination addresses
  - same protocol
  - ~~same packet identifier~~

Merge Example 1

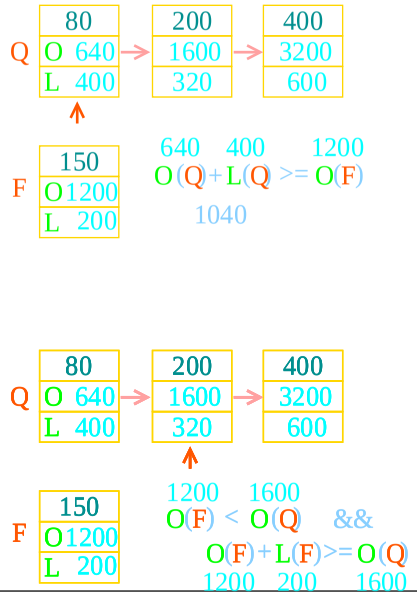


- Queue of waiting fragments
  - with particular Offsets and Lengths
- Fragment arrives
  - It also has Offset and Length
- We run the merge algorithm

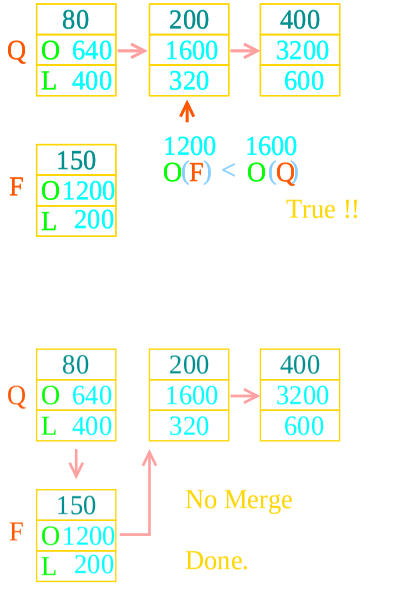
Merge Example 1 (2)



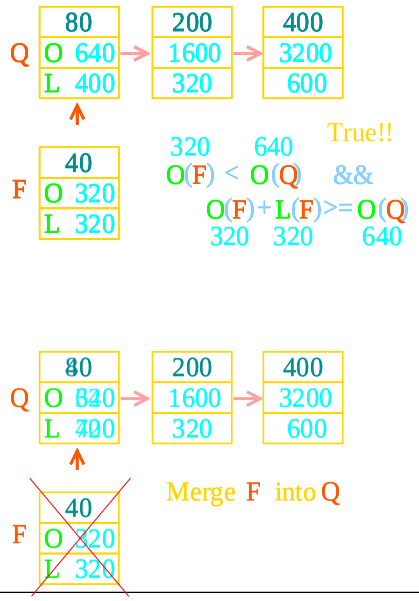
Merge Example 1 (3)



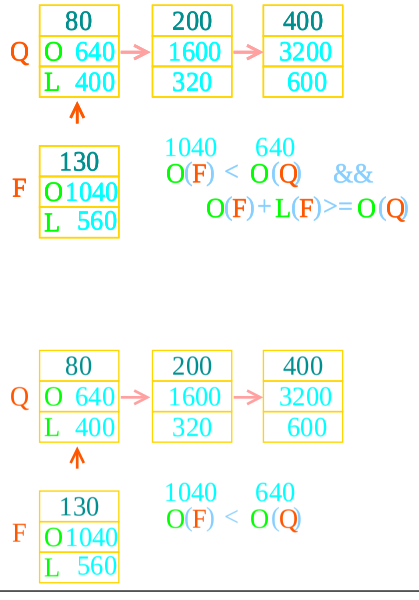
Merge Example 1 (4)



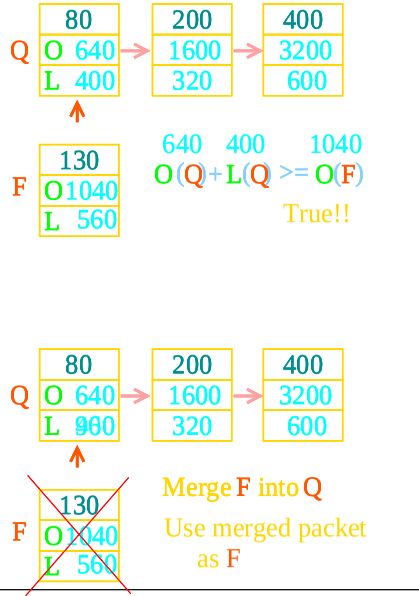
Merge Example 2



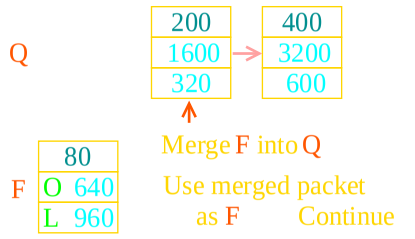
Merge Example 3

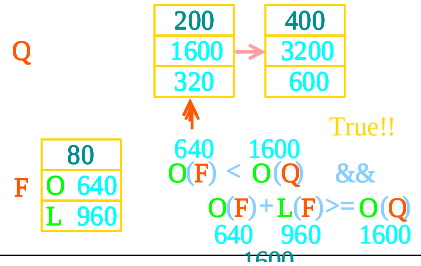


Merge Example 3 (2)

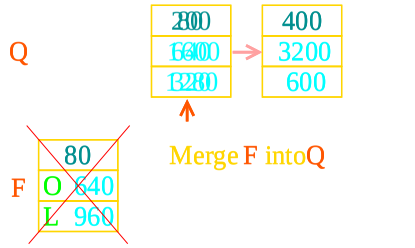


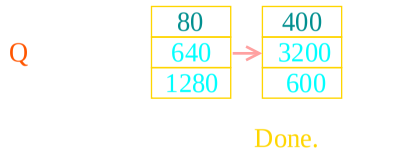
Merge Example 3 (3)





Merge Example 3 (4)





Merge Example Summary

- Example 1
  - Fragment inserted in queue
  - In correct location
    - So offsets in queue remain sorted
  - No merge possible
- Example 2
  - Fragment merged into fragment in queue
    - Data fit before queue fragment
- Example 3
  - Fragment merged into fragment in queue
    - Data fit after queue fragment
  - Resulting merged fragment
    - Merged into following queue fragment
  - Original fragment filled the gap
    - between two fragments on the queue
- Many other variations possible

Merging Fragments

- Now know when to merge fragments
  - But still need details for how to merge
- Assume two fragments F1 and F2 to merge
  - Order so Offset of F1 <= Offset of F2
    - That is, fragment with lower offset is F1
    - If offsets are equal
      - fragment with bigger length is F1
    - The other is F2
- If O(F1) + L(F1) >= O(F2) + L(F2)
  - Discard F2
  - Result is F1
- So, we know
  - O(F1) < O(F2) &&
  - O(F1) + L(F1) < O(F2) + L(F2)

Merging Fragments (2)

- We know
  - O(F1) < O(F2) &&
  - O(F1) + L(F1) < O(F2) + L(F2)
- Build new fragment NF
  - O(NF) = O(F1)
  - L(NF) = O(F2) + L(F2) - O(F1)
  - M(NF) = M(F2)
    - The More Fragments flag
  - TTL(NF) = min(TTL(F1),TTL(F2))
  - All other header fields in F1 and F2 the same
    - Copy values into NF
  - except the checksum
    - calculate appropriate value
  - Copy the data from F1 and F2 into NF
    - Initial data from F1
    - Trailing data from F2
      - Any overlapping data, from either

Merging Fragments (3)

- After Merging
  - If O(NF) == 0 &&
  - M(NF) == 0
  - We have a complete packet
    - Deliver it to transport protocol
- Otherwise
  - NF remains on reassembly queue
    - As in earlier explanation
- What happens if a fragment is lost ??
  - That is, never arrives at destination