Chapter 12

Transmission Control Protocol (TCP) – Part Two
## CONTENTS

### Part One
1. Process-to-process Communication
2. TCP Services
3. Numbering Bytes
4. Flow Control
5. Silly Window Syndrome
6. Error Control
7. TCP Timers

### Part Two
7. Retransmission Timer
8. Congestion Control
9. Segment
10. Options
11. Checksum
12. Connection
13. State Transition Diagram
14. TCP Operation
15. TCP Package
12.7 TCP Timers

- **Timers**
  - **Retransmission**
    - Missing ACK
  - **Persistence**
    - 0-Sized Window
  - **Keepalive**
    - Detect Idle clients
  - **Time-waited**
    - Connection Termination
12.7.1 Retransmission Timer

- How long to wait for an ACK of a previously sent segment before retransmission.
- Depends on distance and network traffic density.
  - Retransmission time should be *dynamic*.
  - Retransmission time = \(2 \times RTT\)
- Dynamic Calculation of RTT:
  - Use a timestamp TCP option (discussed later), or
    a) Actually measure RTT of first two segments of a connection
    b) \(RTT_{\text{new estimate}} = \alpha \times RTT_{\text{old estimate}} + (1 - \alpha) \times RTT_{\text{actual}}\)
    c) Typically, \(\alpha = 0.90\)
    d) Do NOT consider retransmitted segments into the above calculation of RTT
12.8 Congestion Control

- Congestion occurs when some routers along the path run out of buffers and drop packets.
- Sending TCP assumes that a missing ACK is due to congestion.
  - Retransmission will only aggravate the congestion
- A Congestion Window (CongWin) is used to control number of segments transmitted simultaneously (i.e. before waiting for an ACK)

Sender Window = Min (RecvWin, CongWin)

- During connection establishment, each party specifies its maximum Receiving Window (RecvWinMax), and the Maximum Segment Size (MSS) it can receive.
The (Slow Start, Additive Increase, Multiplicative Decrease) Cycle

Initially,
- CongWin = Receiver’s MSS = N
- Threshold = \( \frac{1}{2} \) RecevWinMax

Slow:
- CongWin = 2 \times CongWin

Additive:
- CongWin = 1 + CongWin

\( \text{CongWin} \leq \text{RecvWin} \)

If time-out,
- Threshold = \( \frac{1}{2} \) CongWin
- CongWin = N
12.9 TCP Segment

- Source port address: 16 bits
- Destination port address: 16 bits
- Sequence number: 32 bits
- Acknowledgment number: 32 bits
- 6 Control Bits
- 20 to 60 Bytes
- Sliding Window size in Bytes
- Offset of last byte of urgent data
- # of 4-Byte words
- Options & padding
Control Fields

- Enable flow control, connection establishment & termination, and mode of data transfer.
- One or more bits may be set at the same time
- Discussed in detail later

<table>
<thead>
<tr>
<th>URG: Urgent pointer is valid</th>
<th>RST: Reset the connection</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK: Acknowledgment is valid</td>
<td>SYN: Synchronize sequence numbers</td>
</tr>
<tr>
<td>PSH: Request for push</td>
<td>FIN: Terminate the connection</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URG</th>
<th>ACK</th>
<th>PSH</th>
<th>RST</th>
<th>SYN</th>
<th>FIN</th>
</tr>
</thead>
</table>
12.10 Options

- Up to 40 bytes of optional information to the destination

```
Options
  Single-byte
    - End of option
    - No operation
  Multiple-byte
    - Maximum segment size
    - Window scale factor
    - Timestamp
```
No-operation Option

- A Filler option used for alignment purposes only.

- **Code:** 1 00000001
  - No operation option

- **An 11-byte option**
  - Used to align beginning of an option

- **An 8-byte option**
  - Used to align the next option

- **A 7-byte option**
  - NO-OP
End-of-option Option

- Used as a padding to terminate options. Only ONE may be present.
- The remainder of this 32-bit word is garbage
  - Payload data starts at the next 32-bit word.

```
Code: 0 00000000
     a. End of option

Options
END-OP

Data

b. Used for padding
```
# Maximum segment size (MSS) Option

<table>
<thead>
<tr>
<th>Code: 2 00000010</th>
<th>Length: 4 00000100</th>
<th>Maximum segment size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>1 byte</td>
<td>2 bytes</td>
</tr>
</tbody>
</table>

- The maximum number of data bytes I can receive in one segment
- Each party determines its MSS during connection establishment
  - Cannot be changed later.
- Defaults to 536 bytes
**Window scale factor Option**

<table>
<thead>
<tr>
<th>Code: 3 00000011</th>
<th>Length: 3 00000011</th>
<th>Scale factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>1 byte</td>
<td>1 byte</td>
</tr>
</tbody>
</table>

- The 16-bit Window Size in the header is too small for high-speed links (e.g. OC-24 @ 1.2Gbps)
- Actual Sender Window size = WinSize in header \( \times 2^{\text{WinScaleFactor}} \)
- WinScaleFactor \( \leq 16 \)
- This option may only appear during connection establishment.
  - Header.WinSize may change, but WinScaleFactor is constant throughout the entire connection
**Timestamp Option**

<table>
<thead>
<tr>
<th>Code: 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001000</td>
</tr>
<tr>
<td>Length: 10</td>
</tr>
<tr>
<td>00001010</td>
</tr>
</tbody>
</table>

- Set by the sender just before the segment leaves (to IP layer)
- Echoed by the receiver just before it ACKs the segment
- The sender measure RTT as:
  
  the arrival time of the ACK - the timestamp echo reply.
12.11 Checksum

- Same as in UDP, but mandatory

<table>
<thead>
<tr>
<th>Pseudoheader</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>32-bit source IP address</td>
<td></td>
</tr>
<tr>
<td>32-bit destination IP address</td>
<td></td>
</tr>
<tr>
<td>All 0s</td>
<td>8-bit protocol (6)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Header</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Source port</td>
<td>Destination port</td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td>Acknowledgment number</td>
<td></td>
</tr>
<tr>
<td>HLEN</td>
<td>Reserved</td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
</tbody>
</table>

Data and Option
(Padding must be added to make the data a multiple of 16-bits)
12.12 Connection

- Connections must be *established* to facilitate acknowledgements, and retransmissions.
- Connections are *terminated* at the end.
- A connection may be *reset* at any time in one of these 3 cases:
  1. A destination rest a connection requested for a nonexistent port
  2. One party encounters abnormal situation, so it resets the connection
  3. One party detects that the other party has been idle for too long.
Connection Establishment: A 3-way Handshaking

**Client**

- **ISN**: Possibly MSS and/or WinScaleFactor
- **WinSize**, **ISN**: Possibly MSS and/or WinScaleFactor

**Server**

- **LISTENing State**
- **WinSize**, **Possibly real data**

**Interactions**

1. **Segment 1**: SYN
   - seq: 1200, ack: –
2. **Segment 2**: SYN + ACK
   - seq: 4800, ack: 1201
3. **Segment 3**: ACK
   - seq: 1201, ack: 4801
Connection Termination: A 4-way handshaking

 Usually, Client

 Active Close

 Segment 1: FIN
 seq: 2500, ack: –

 Segment 2: ACK
 seq: 7000, ack: 2501

 Passive Close

 Segment 3: FIN
 seq: 7001, ack: 2501

 Segment 4: ACK
 seq: 2501, ack: 7002

 Usually, Server

 Time

 Time
12.13 TCP FSM: State Transition Diagram

**Client states**

- **CLOSED**
  - Transition: **TIME-WAIT**
  - Transition: **FIN / ACK**
  - Transition: **FIN-WAIT-2**
    - Transition: **FIN / ACK**
    - Transition: **ACK / -**
  - Transition: **FIN-WAIT-1**
    - Transition: **FIN / FIN**
    - Transition: **SYN-SENT**
      - Transition: **SYN1 ACK / ACK**
      - Transition: **ESTABLISHED**
        - Transition: **FIN / ACK**
        - Transition: **ACK / -**

**Server states**

- **CLOSED**
  - Transition: **(Time-out)**
  - Transition: **Active open / SYN**
  - Transition: **LAST ACK**
    - Transition: **Close / FIN**
    - Transition: **CLOSE WAIT**
      - Transition: **FIN / ACK**
      - Transition: **ESTABLISHED**
        - Transition: **ACK / -**
  - Transition: **PASSIVE OPEN / -**
  - Transition: **LISTEN**
    - Transition: **SYN / SYN + ACK**
    - Transition: **SYN-RCVD**
12.14 TCP Operation

- Encapsulation and decapsulation
- Buffering
- Multiplexing and demultiplexing
- Pushing Data
  - Send NOW, deliver to server application ASAP
- Urgent Data
  - Delivered to other application out-of-order
Encapsulation and decapsulation

a. Encapsulation

b. Decapsulation
Multiplexing and demultiplexing
12.15 TCP Package

Application layer
Messages to and from application

TCP

Timers

Input processing module

Main module

TCBs

Output processing module

TCP segment

TCP segment

IP layer
Transmission Control Blocks

<table>
<thead>
<tr>
<th>State</th>
<th>Process</th>
<th>Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Buffers

of the Connection