**T07: Transport Layer I**

**Q1:** Explain the difference between end-to-end communication and node-node communication with a diagram. What does transport layer facilitate (end-to-end communication or node-node communication)?

Node-to-note. A physical (local) connection between 2 hosts (data link) with flow/error control

End-to-end. A connection of any distance (not local) uses the network layer to find/identify the target host/machine. Transport layer targets exactly which service/process it is to be established between 2 remote hosts.

**Q2:** What are the similarities and dissimilarities in transport and data link layer protocols? Is flow control & buffering in transport layer different from data link layer? Briefly explain.

|  |  |  |
| --- | --- | --- |
| Basis | Data Link | Transport |
| Addressing | Local address (LAN) MAC | Port number |
| Connection establishment | No (always connected) | 3-way handshake |
| Connection release | No (always connected) | 3 or 4 way handshake |
| Flow control, error control | Similar (stop-and-wait ARQ etc.) | Similar (dynamic sliding window) |
| Multiplexing | No | Yes, upwards and downwards |
| Crash recovery | No | Yes |

**Q3:** Both UDP and TCP use port number to identify the destination entity when delivering a message. Give two reasons for why these protocols invented a new abstract ID (port number), instead of using process IDs, which already existed when these protocols were designed?

1. Port number is more universally friendly between different OS systems. No OS target is required because port numbers are universal. Process ID’s are OS specific.

2. A single process may establish multiple channels of communications. A single process ID (per process) as the destination identifier cannot be used to distinguish between these channels.

3. having processes listen on well0known ports is easy but well known process ID’s are impossible.

**Q4:** Why does the maximum packet lifetime, T, have to be large enough to ensure that not only the packet but also its acknowledgements have vanished?

Packets could be infinitely cycled in the network/s if a TTL is not assigned. The packet sequence must be big enough so that after the time is up the sequence number can be reused so that the previous packet using the same sequence number has already expired. 3-way handshake used to acknowledge which sequence number is being used by each host.

If timer before sending using the same sequence is not big enough, receiver will acknowledge old duplicate when sender thinks it is acknowledging its newest packet. Before a given sequence can be reused, the timer must be big enough so that any acknowledgements would have been received for older packets before the same sequence is reused.

**Q5:** Why does the symmetric connection release is not completely foolproof? Explain with a diagram.

Two armies’ problem. A sends DR to B. B sends Ack to A. B sends DR to A. (or B sends Ack and DR at same time) A then sends Ack to B before dropping connection. (4-way handshake). Symmetrical is better than Asymmetrical.

2 armies problem. If A is not sure B got the message A will not fight and vice versa.

**Q6:** How does transport layer recover from a router or network crash? Explain your answer with respect to unreliable datagram service (packet switching), connection-oriented network service (virtual-circuit packet switching).



For datagram service (PS) network: Transport entities expect lost TPDU’s all the time and know how to cope

For connection-oriented (VC) network: Loss of virtual circuit is handled by establishing a new one and probing the remote transport entity for TPDU’s received and not received.

**Q7:** When hosts/servers crash, recovery becomes an issue. There are always situations where the protocol fails to recover properly.

For the strategies (sending host and receiving host) shown in the table below, indicate whether it’s **OK**, **DUP** (packet duplicate), **LOST** (packet loss) appropriately.

Assume each client (sender) can be in one of the two states (S0, S1) when the Sever (receiver) announces that it had just rebooted and request that its clients (senders) to inform the status of all open connections.

* No TPDUs outstanding, S0
* One TPDU outstanding, S1

*Hint: first one is done for your reference*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Strategy used by Client** | **Strategy used by Server** | | | | | |
| **First ACK, then write** | | | **First write, then ACK** | | |
| AC(W) | AWC | C(AW) | C(WA) | WAC | WC(A) |
| **Always retransmit the last TPDU** | OK | DUP | OK | OK | DUP | DUP |
| **Never retransmit the last TPDU** | LOST | OK | LOST |  |  |  |

* 1. | P a g e

OK OK DUP



**Retransmit only in state S0**

**(**

**No TPDU outstanding)**

**Retransmit only in state S1**

**TPDU outstanding)**

**(**

Router crash can be hidden from host. Host crash cannot be hidden

* 1. | P a g e