Best Effort Delivery - User Datagram Protocol Part 2

Exercise 1: UDP Datagram Buffering

In this exercise, you will gain an understanding of the concept of datagrams as well as message buffering and how it applies to UDP datagrams.

1. Start one copy of the Networking Workbench on computer A and one copy on computer B. Computer A will serve as sender and computer B will be the receiver
2. Prepare the sender (Computer A):
   - i) On the Computer A workbench menu, click on UDP workbench.
   - ii) Set the **destination IP addresses**to the address of Computer B.
   - iii) Select the “unicast” mode for this exercise.
   - iv) Set the **send port** of Computer A to 8000
   - v) **DO NOT** turn on the receiver
3. Prepare the receiver (Computer B):
   - i) On the Computer B workbench menu, click on UDP □ Non-blocking receiver.
   - ii) Set the **receive port** to 8000
   - iii) Click on the **BIND** button.

Follow the steps below in EXACT sequence:

4. Using the unicast addressing mode, use the UDP Workbench of Computer A to send a message containing “A” to the receiver Computer B

Did anything happen at the receiver?

5. Click “Receive”— what happens?

> Computer B receives the message containing “A”.

6. Send a message containing “B” to the receiver then send a message containing “C” to the receiver.
7. Click “Receive” at the receiver— what happens?

> Computer B receives the message containing “B”.

8. Click “Receive” again—what happens?

> Computer B receives the message containing “C”.

9. Try this again by using the UDP Workbench on Computer A to send a message many times to the receiver in quick succession.
10. Now press “Receive” on Computer B a few times

Describe what is happening here in terms of packet queuing and buffering.

> It takes a while to receive the messages from Computer A to Computer B.

11. Using the sender, send a few messages again then close the application.
12. Click on receive on the receiver computer.

What happens if the sender is shut down after it sends a message but before the message has been displayed by the receiver?

> It still receives the message sent by the sender

Based on these results, where do you think the buffer is held, at the sender or receiver end?

> Based on these results, I think the buffer is held at the receiver end.

Are the UDP segment boundaries maintained throughout the entire transmission process (including any buffering that might occur), or can they be further divided or concatenated? In other words, are the messages kept separate, or are they merged when retrieved from the buffer?

> The messages are kept separate when they are retrieved from the buffer.

Exercise 2: Understanding Binding

In this exercise, you will gain an understanding of the need for binding processes to ports and why the same port number can be reused on different computers.

1. Use two copies of the "UDP Workbench" application, both running on the same computer.
   i) Set up the IP addresses of both application instances. Both IP addresses should be set to the address of the local computer.
   ii) Set the send port of the first copy of the UDP Workbench to 8000 and its receive port to 8001. Set the send port of the second copy of the UDP Workbench to 8001 and its receive port to 8000.
   iii) Do not turn on the receivers yet
2. BEFORE enabling the receiver at either copy, send a message from the first copy to the second copy of the workbench.

What happens (does the message arrive)?

> The message does not arrive.

3. Now, enable the receiver at the second copy of the UDP Workbench.

Does the message arrive if you now enable the receiver (i.e., has it been buffered)?

> The message still does not arrive.

4. Send a second message from the first copy to the second copy.

What happens (does the message arrive)?

> The message does arrive but does not display the text but rather the second copy receives it as a packet.

What does this behavior tell you about the significance of binding?

> The significance of binding based on the behavior of the previous result shows that binding the receiver to the sender would buffer the message and still be able to receive the message in a form of a packet.

5. Click on DONE on both workbenches then open the menu for the "Non-Blocking Receiver" on both copies
6. On the first copy of the Non-Blocking Receiver, leave its receive port at the default value then click on **Bind**.
7. On the second copy of the Non-Blocking Receiver at the same computer, ensure that the receive port value is the same as for the first copy then click on **Bind**.

What happens?

An error occurs.

Why is this computer behavior correct and also necessary?

This computer behavoir is correct and also necessary because there cannot be two receivers on the same computer or local machine.

What do you think would have happened if the second receiver had been at a different computer?

The second receiver would be able to BIND.

8. Confirm your hypothesis empirically (check your answer by trying it out).

What happens?

The second receiver was able to BIND.