15-640 Distributed Systems **Varun Saravagi (vsaravag)**

Protocol Description

<u>Assumptions:</u>

1. Both the client and the server have the same endian-ness.
2. There is no packet loss between client and server.

Both the points were mentioned on Piazza and taken from there.

<u>Description:</u>

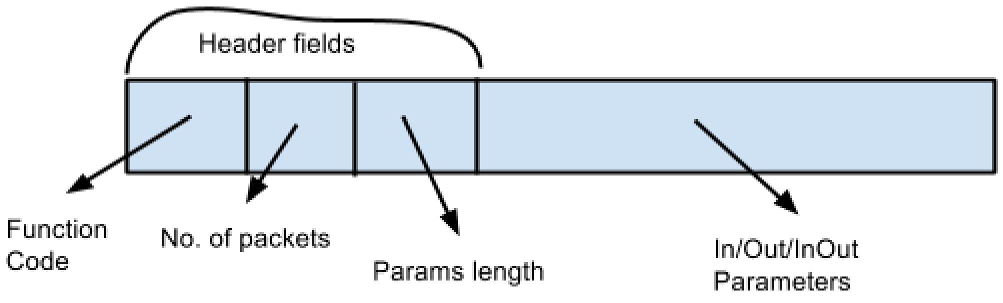

Fig 1: Packet Description

This is the standard which both the client and server follow while communicating with each other. The description of various fields are as follows:

1. *Function Code*: contains the function which needs to communicate (OPEN, CLOSE etc.)
2. *No. of packets*: The number of packets the other entity should expect to receive. The maximum size of a packet is 1500.
3. *Params length*: The size of the params which are being sent to the other entity. The size would vary as per the function. Please note that if the params have a void*/char*, the length of the pointer would also be sent along with.

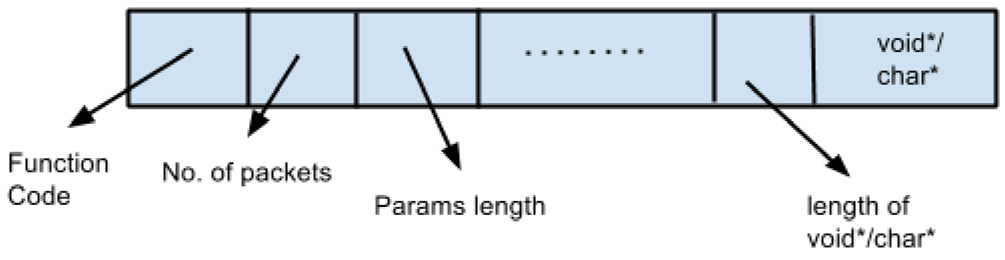

Fig 2: How a void*/char* is sent

4. *In/Out/In-Out Parameters*: Contains the parameters which are required by the called function.

A packet is prepared in the format mentioned before sending over the network. When received, the parameters are separated from the header fields and extracted.

Notes:
If the total size of the packet exceeds 1500, then multiple packets are sent. The subsequent packets, after the first one as shown in Fig 1, have only the parameter contents and do not contain function code, no of packets and params length.

The serialization is done using memcpy. One single packet is a char* in which all the required fields are memcpy(ied). Similarly, while deserializing, all the fields sent are acquired using memcpy(ied).

Limitations:
The limitation of this design is that it would not be able to communicate between machines having different architectures. This is because I am using sizeof(type) to populate the packet and sizeof(type) varies as per the architecture.