CS4514 HELP Session 3

Concurrent Server Using Selective Repeat

Data Link Layer

Mingzhe Li
lmz@cs.wpi.edu
12/05/2005

Description

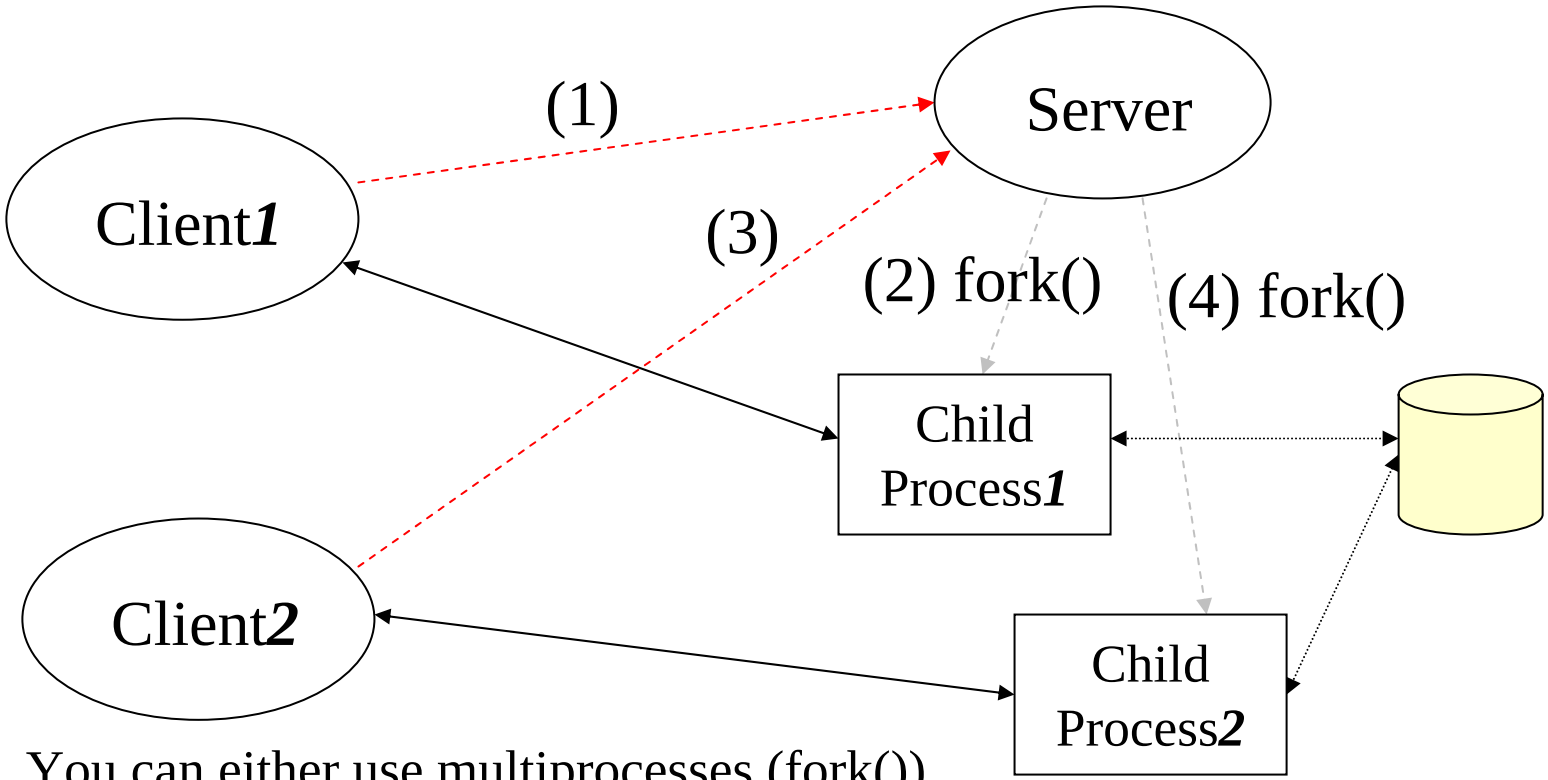
- **Objective:**

  To implement a simple concurrent server and clients having four emulated network protocol stacks.

  - **Application layer: Customized Applications**
  - **Network layer: Message←→Packet (send&recv)**
  - **Datalink layer: Packet ←→ Frame and Selective Repeat sliding window protocol**
  - **Physical layer: TCP connection.**

- **To get full credit, you need a sending and receiving windows size >=4.**
- **Your programs should compile and work on any host of ccc.WPI.EDU.**

System Overview

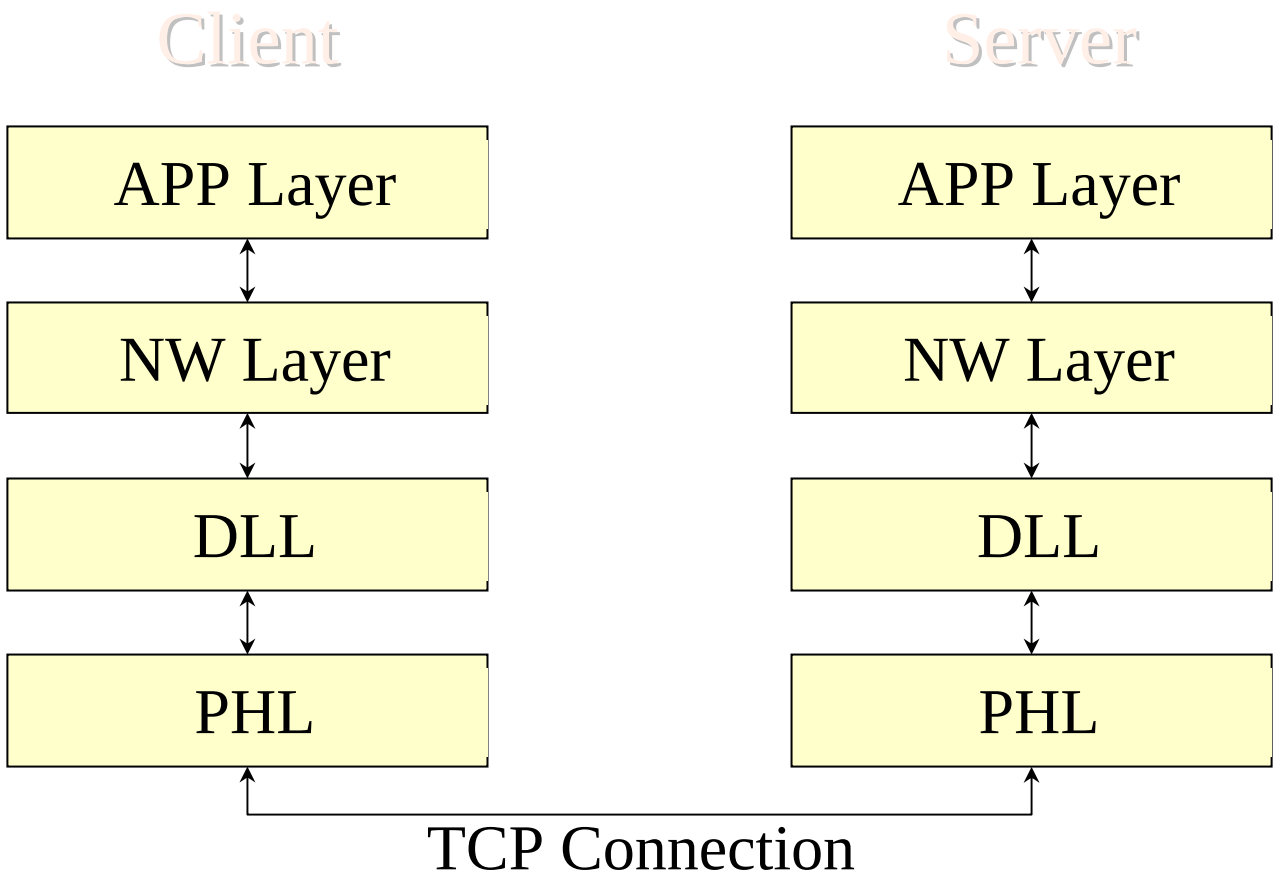


You can either use multiprocesses (fork())

or multithreading (pthread)

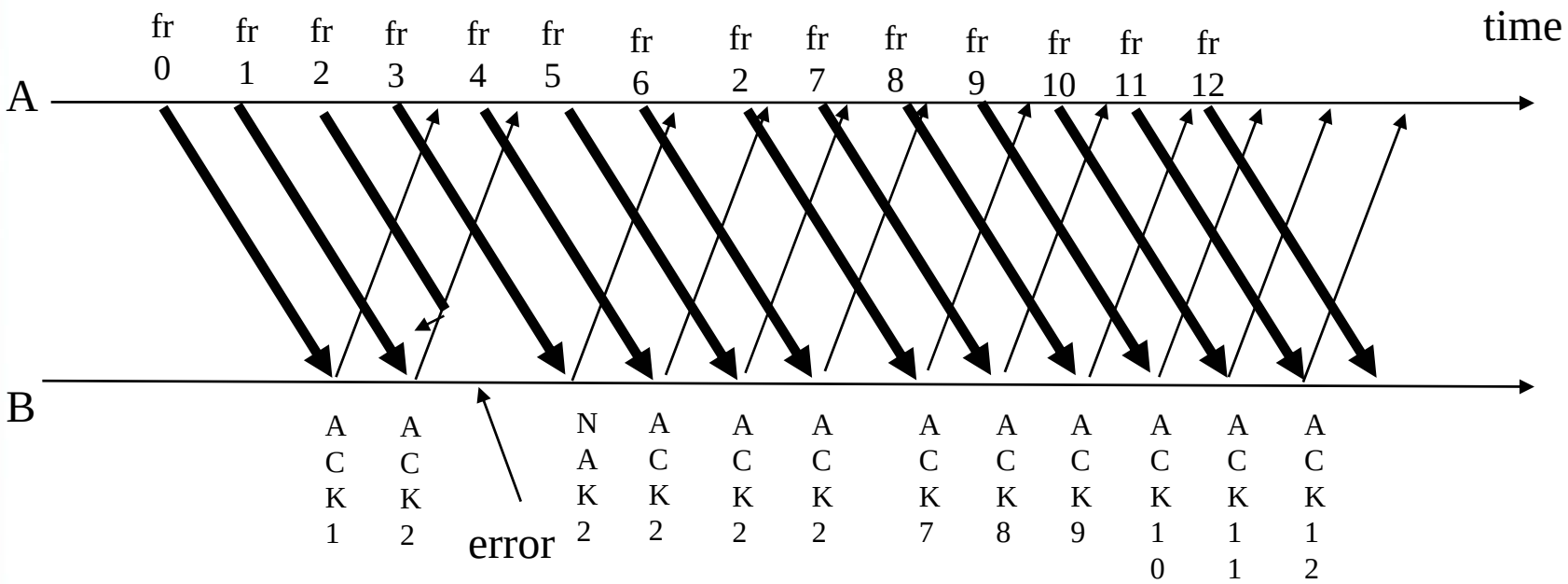
You need to implement concurrent access to the database.

System Framework

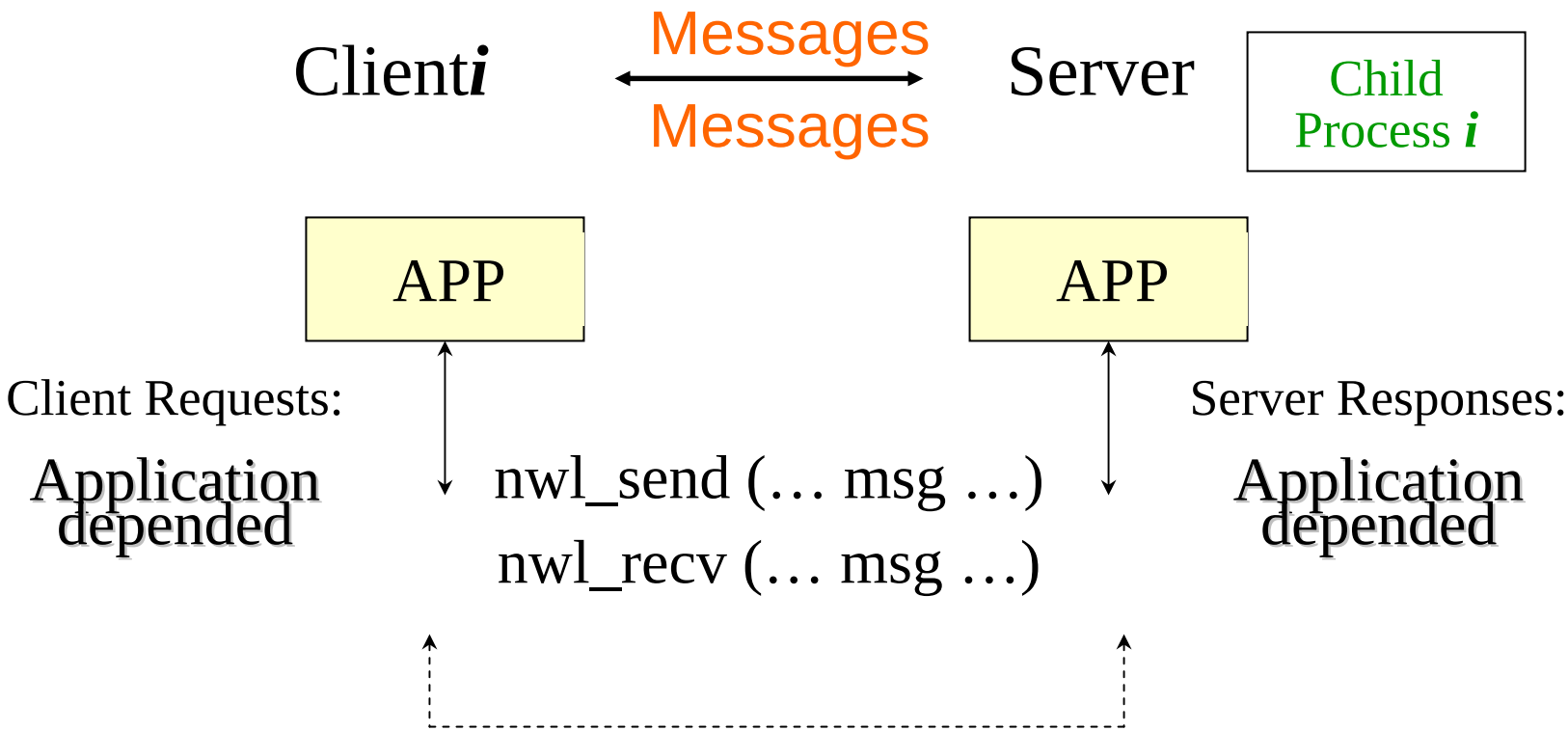


Four Layer stacks

Selective Repeat

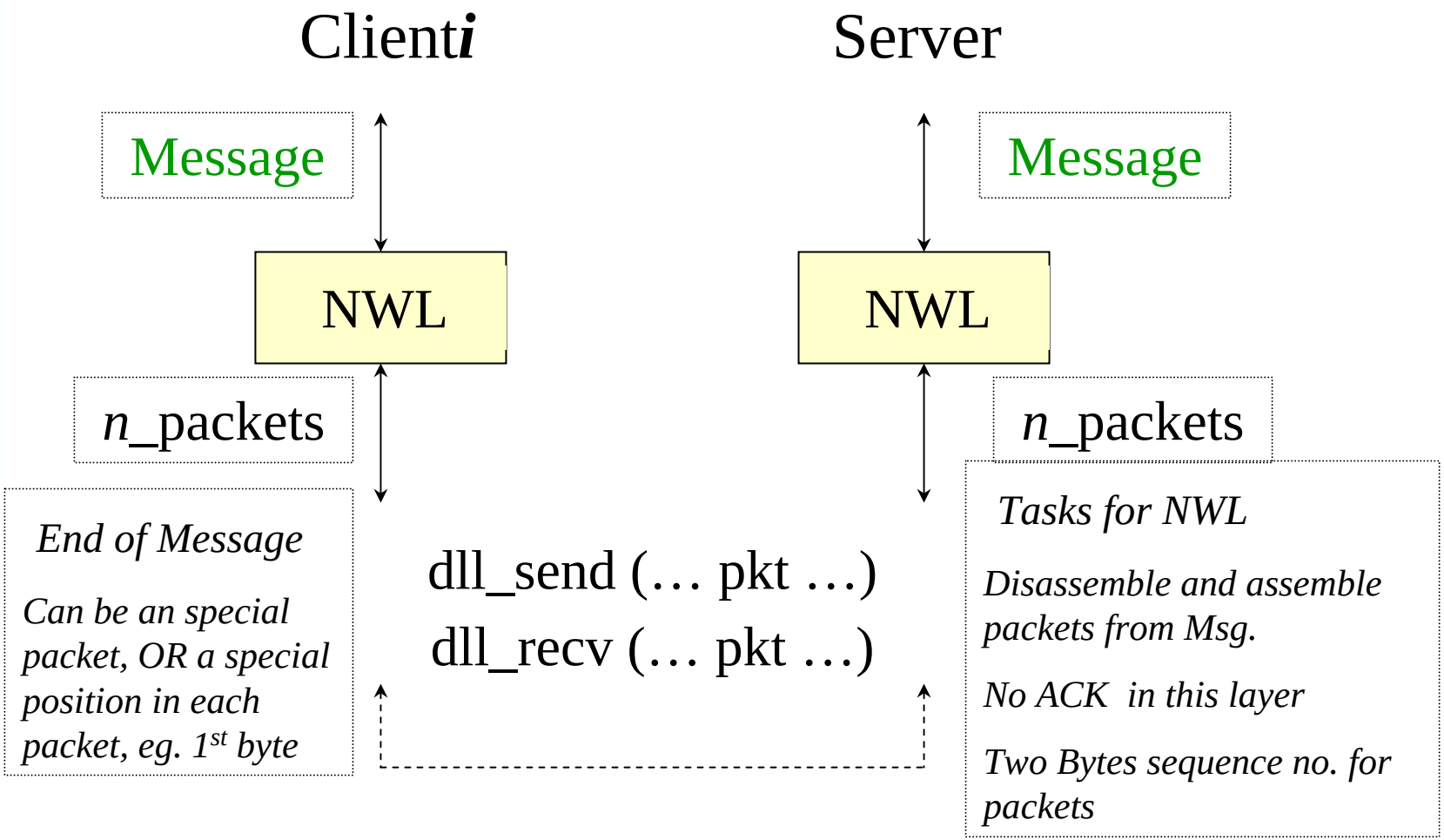


How the System Works: Layer by Layer
Application Layer



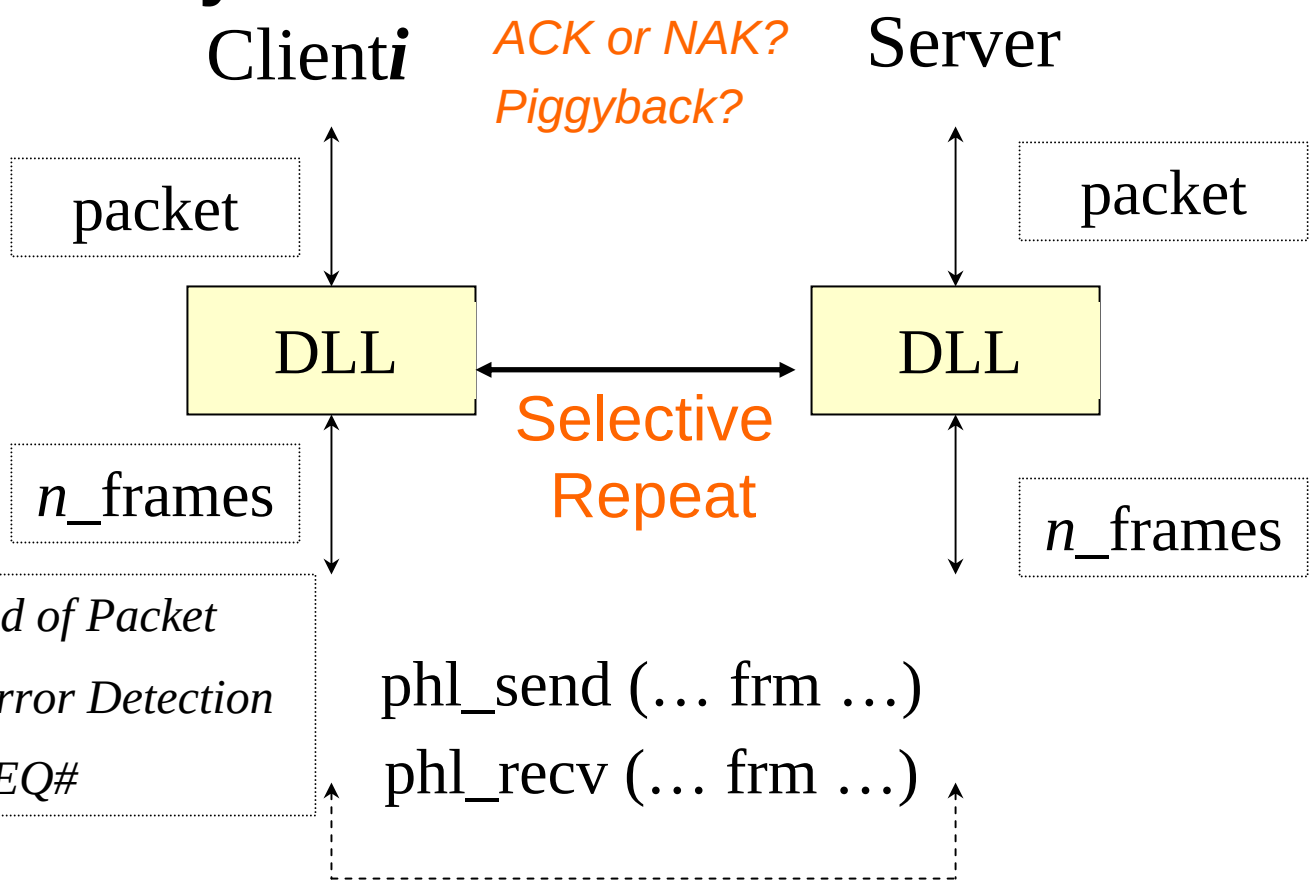
At least 5 operations, there is at least one long operation in each direction that will easily test your sliding window

How the System Works: Layer by Layer
Network Layer



Note: The max_size of a packet is 256 bytes, The network layer will send packets until blocked by the Data Link Layer. But HOW?

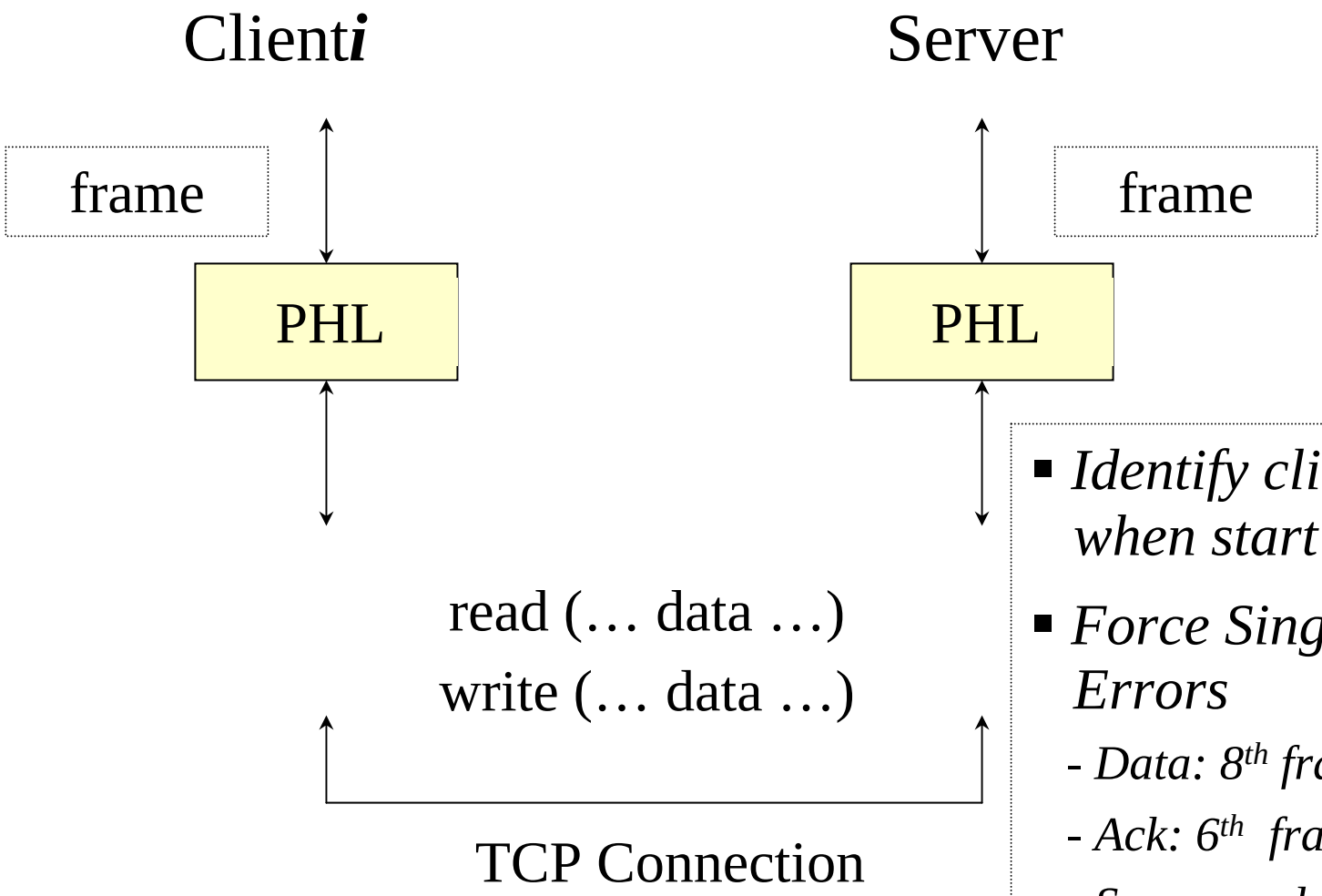
How the System Works: Layer by Layer DataLink Layer



Note: The max_size of a frame payload is 100 bytes

Sliding window size >=4

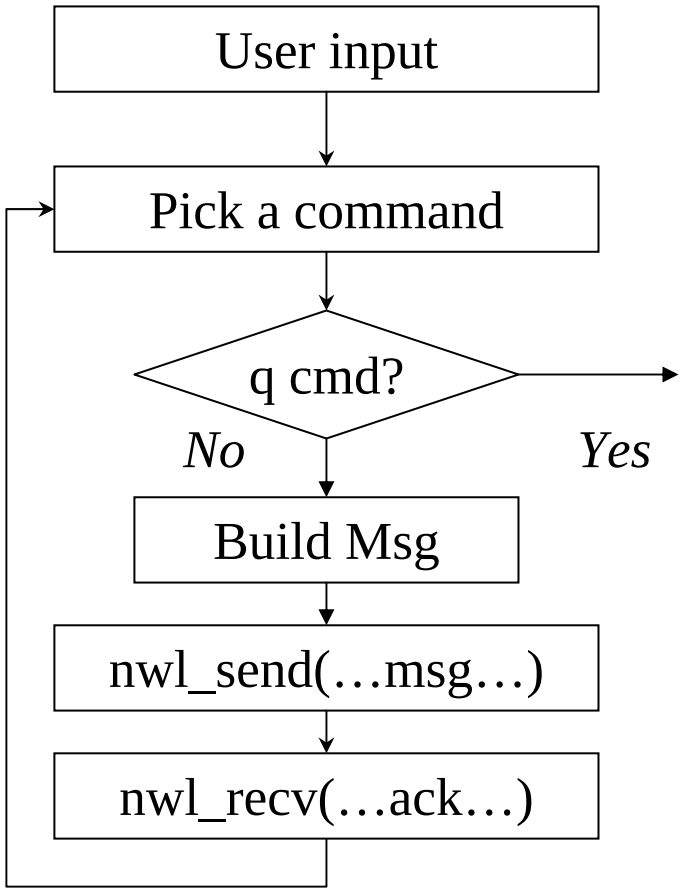
How the System Works: Layer by Layer



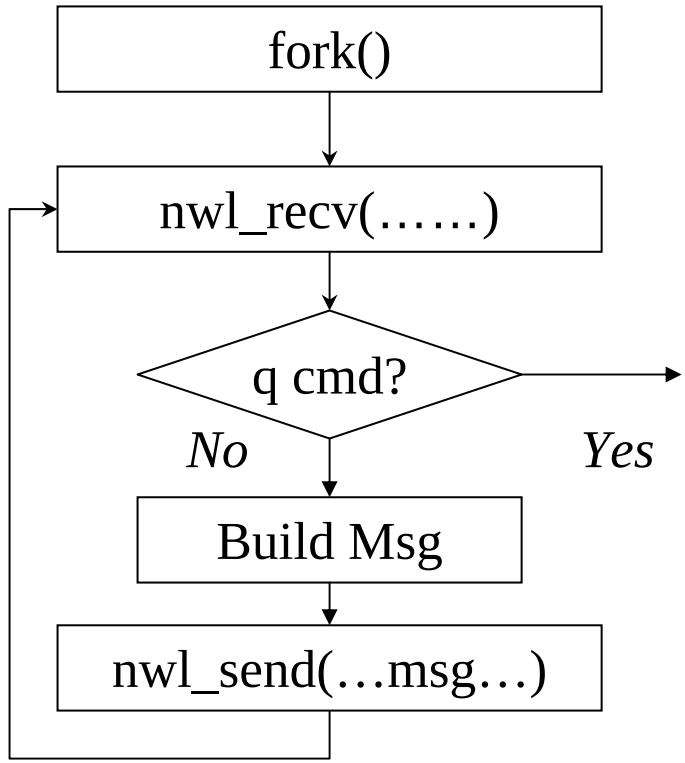
- *Identify client when start*
- *Force Single Bit Errors*
  - *Data: 8th frame*
  - *Ack: 6th frame*
  - *Server and clients*

How the Functions Work: Layer by Layer

client APP

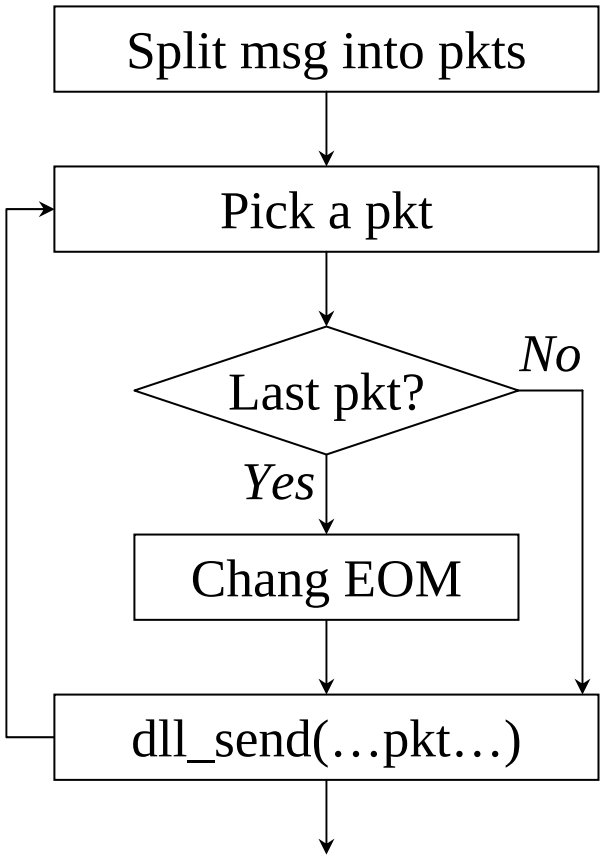


server child process APP

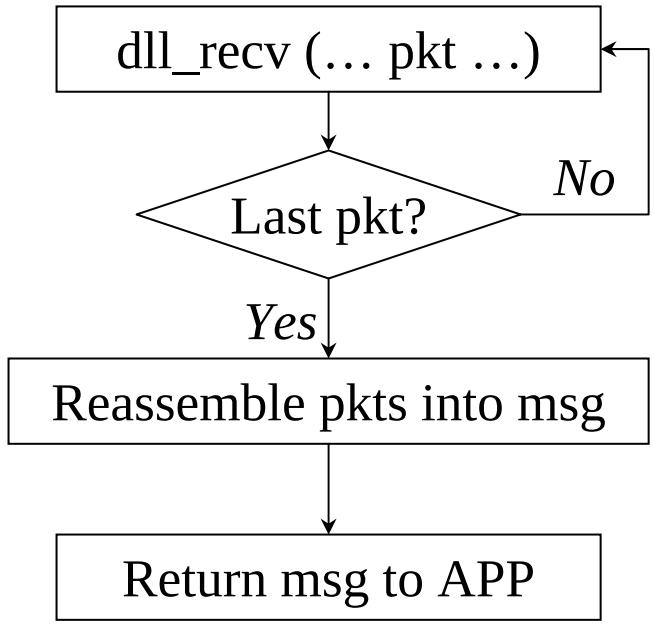


How the Functions Work: Layer by Layer

nwl_send (… msg …)



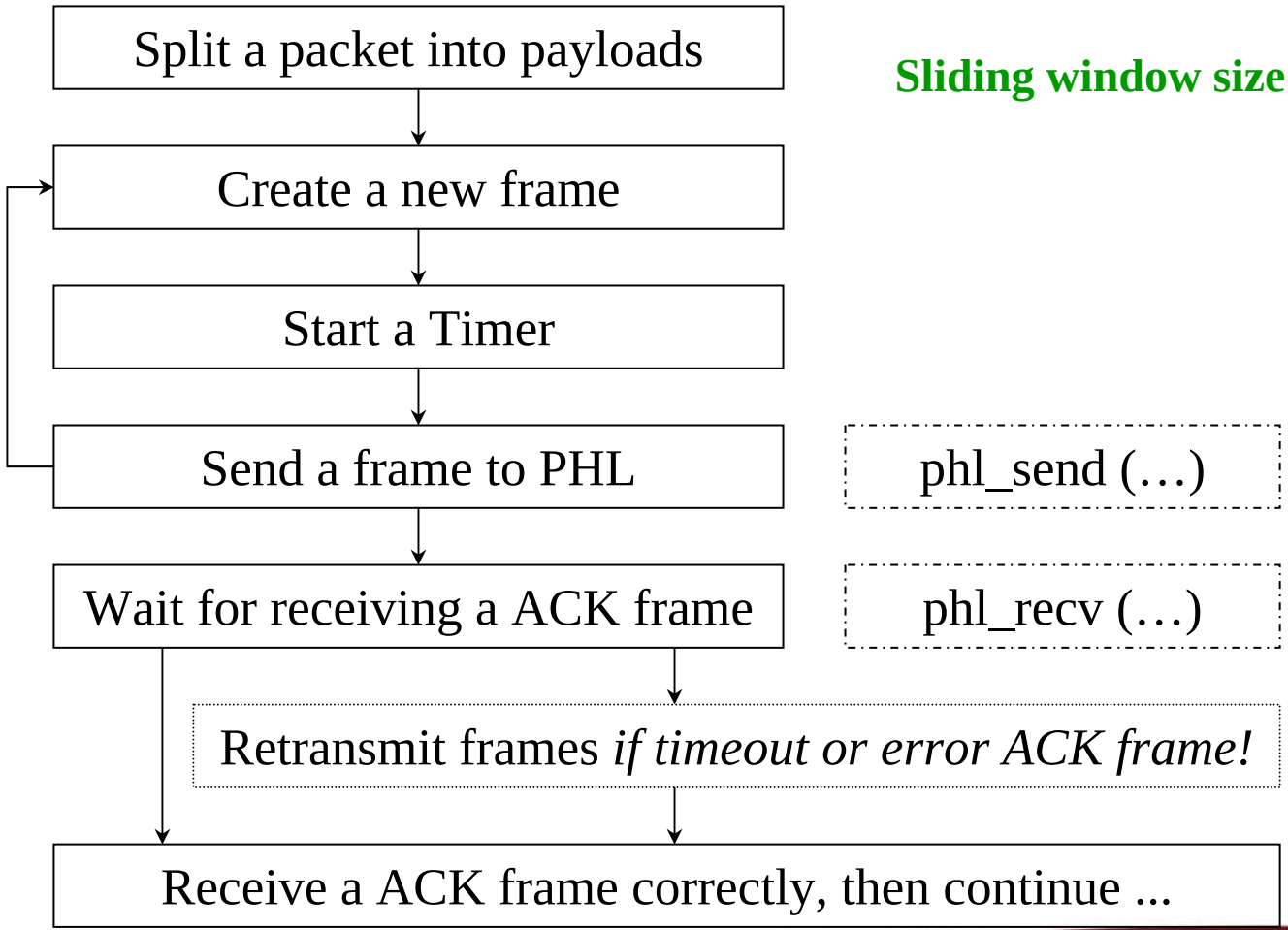
nwl_recv (… msg …)



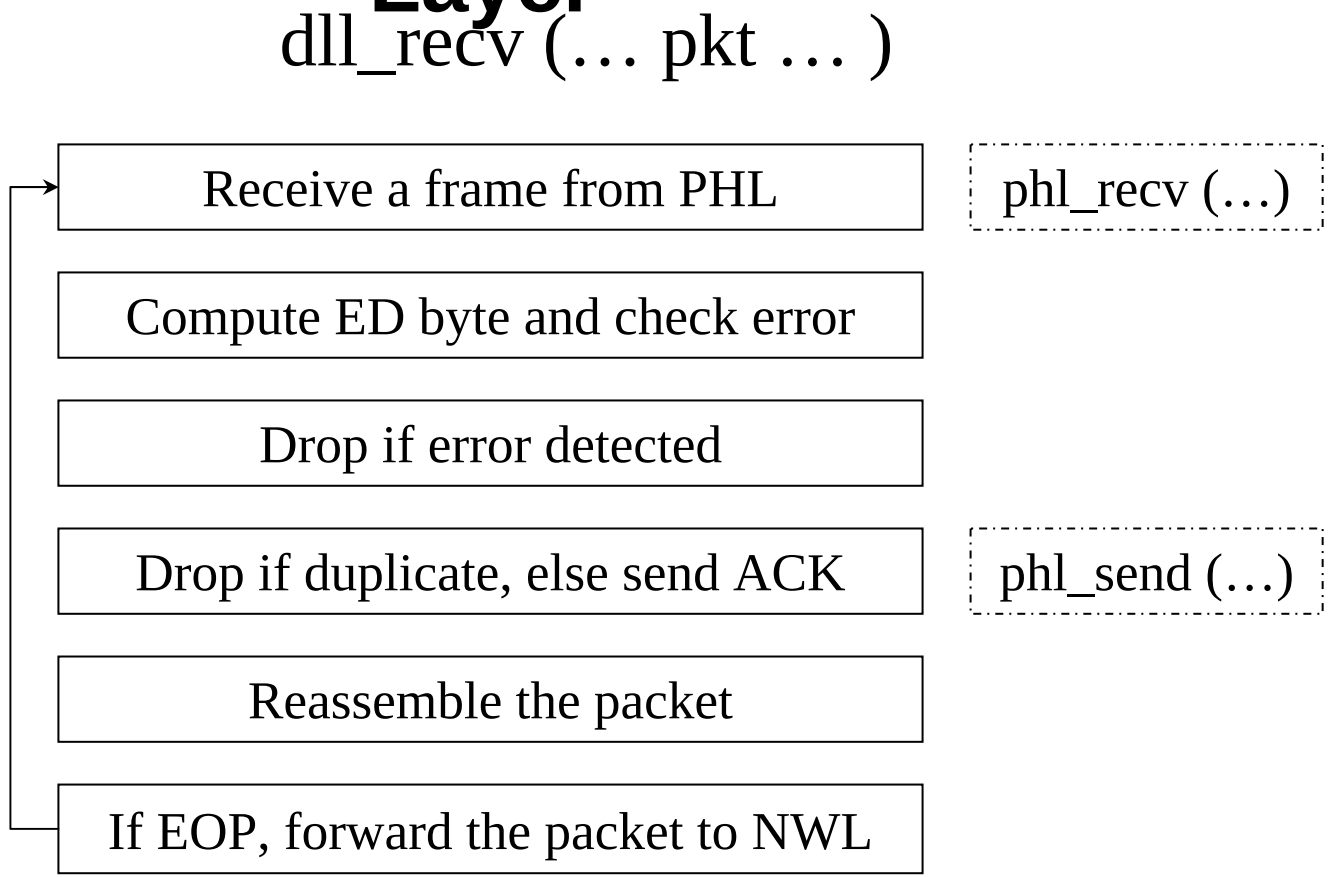
Note: you need have a mechanism to decide the last packet in a message (EOM).

How the Functions Work: Layer by Layer

dll_send (… pkt …)



How the Functions Work: Layer by Layer



Question: When is the correct time to send *NAK or ACK*?

Not after ED drop, but on receiving next frame or dup frame.

Project Tips-1

- **Sliding Window Protocol: Selective repeat (N>=4)**
  - **Try to implement windows size 1 first**
  - **Then implement N (multiple timers)**
- **Follow the example in the book (protocol 6)**
- **How to terminate client process:**
  - **When the client gets the response to the quit message**
  - **A “clean” way to terminate the server child process? Use wait()!**

Project Tips-2

- **Simulate multiple timer in software**
  - **Approach I**
    - **Using link list or array**
    - **pp.223 on textbook()**
    - **Need signal()**
  - **Approach II**
    - **Using link list or array**
    - **Update the *struct timeval* for next select() call**

Project Tip3

- ***How could the NWL Keep sending packets until blocked by the Data Link Layer ?***

  Our suggestion is that you could use pipe to implement it: NWL keeps writing packets to the pipe until the pipe is full.
- **A simple code of pipe could be found at**

  http://web.umr.edu/~ercal/284/PipeExamples/Examples.html
- **Pipe is more like a socket between local processes.**

Concurrent TCP Server Example (fork)

```
pid_t pid;
int listenfd, connfd;


/* 1. create a socket socket() */
if ((listenfd = socket(AF_INET, SOCK_STREAM, 0)) < 0 )
err_quit("build server socket error\n", -1);
/* 2. fill in sockaddr_in{ } with server's well-known port */
…
/* 3. bind socket to a sockaddr_in structure bind() */
bind (listenfd, ...);
/* 4. specify the backlog of incoming connection requests  listen() */
listen (listenfd, LISTENQ);
while(1){
    connfd = accept(listenfd, ... );  /* probably blocks */
    if(( pid = fork()) == 0){
        close(listenfd);  /* child closes listening socket */
        doit(connfd);     /* process the request */
        close(connfd);    /* done with this client */
        exit(0);
    }
    close(connfd);    /* parent closes connected socket */
}
```

Questions?